

JOSÉ MANUEL MOTTA



UNSAM

UNIVERSIDAD
NACIONAL DE
SAN MARTÍN

ESCUELA DE CIENCIA Y TECNOLOGÍA

SISTEMA DE AUDIO COMPATIBLE CON RESONANCIA MAGNÉTICA

San Martín, Buenos Aires

República Argentina

Año 2017



UNSAM
UNIVERSIDAD
NACIONAL DE
SAN MARTÍN

PROYECTO FINAL INTEGRADOR
INGENIERÍA ELECTRÓNICA
"SISTEMA DE AUDIO COMPATIBLE CON
RESONANCIA MAGNÉTICA"

AUTOR:
JOSE MANUEL MOTTA

AÑO: 2017 / REV. 7

SISTEMA DE AUDIO COMPATIBLE CON RESONANCIA MAGNÉTICA

Proyecto Final Integrador presentado en cumplimiento de las exigencias de la Carrera Ingeniería Electrónica de la Escuela de Ciencia y Tecnología (ECyT), perteneciente a la Universidad Nacional de San Martín (UNSAM).

Realizado por el alumno José Manuel Motta (Legajo CYT-6602).

Tutor: Ing. Daniel Brunati

Director de cátedra: Ing. Marcelo Romeo

Director de carrera: Ing. Jorge Sinderman

San Martín, Buenos Aires

República Argentina

Año 2017



UNSAM
UNIVERSIDAD
NACIONAL DE
SAN MARTÍN

PROYECTO FINAL INTEGRADOR
INGENIERÍA ELECTRÓNICA
"SISTEMA DE AUDIO COMPATIBLE CON
RESONANCIA MAGNÉTICA"

AUTOR:
JOSE MANUEL MOTTA

AÑO: 2017 / REV. 7

Dedicatoria

A mi mujer, Andrea, y a mis hijos, Avril y Benicio.

A mis padres.



Agradecimientos

A An, mi mujer y compañera, quien me apoyó incondicionalmente y me dio ánimos durante este largo camino.

A Daniel Brunati, por guiarme y empujarme en la última cuesta.

A Darío Vega, por enseñarme que es posible “hacer ingeniería” en el ámbito del servicio técnico de equipos médicos.

A los profesores de universidades nacionales de ingeniería, en especial a aquellos de la UNSAM y la UTN.

A los desarrolladores y promotores del proyecto CIAA.

A la música y el sonido, motivos fundamentales de mi ingreso a la ingeniería.



UNSAM
UNIVERSIDAD
NACIONAL DE
SAN MARTÍN

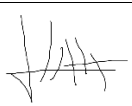
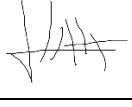
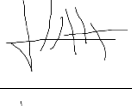
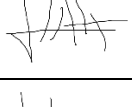
PROYECTO FINAL INTEGRADOR
INGENIERÍA ELECTRÓNICA
"SISTEMA DE AUDIO COMPATIBLE CON
RESONANCIA MAGNÉTICA"

AUTOR:
JOSE MANUEL MOTTA

AÑO: 2017 / REV. 7



Historial de revisiones

Rev.	Fecha	Motivo	Firma
01	24/05/2017	Inicio. Planificación y gestión.	
02	12/07/2017	Etapa analógica y digital.	
03	25/10/2017	Firmware. Diagramas.	
04	14/11/2017	Correcciones generales. Referencias.	
05	18/11/2017	Ensayos y conclusiones.	
06	19/11/2017	Correcciones.	
07	30/11/2017	Correcciones.	

Lista de distribución

Tutor	Ing. Daniel Brunati
Director de cátedra	Ing. Marcelo Romeo
Supporters	Dr. Federico Biafore / Ing. Natalia Requejo
Alumno	José Manuel Motta
Carrera	Ingeniería Electrónica
Director de carrera	Ing. Jorge Sinderman



Introducción

Este proyecto de grado consistió en el diseño, implementación y especificación de un sistema de audio compatible con equipos de resonancia magnética para diagnóstico por imágenes.

El origen del proyecto se fundamentó en la necesidad de contar con un método efectivo para la adquisición de respuestas verbales de pacientes durante la realización de estudios por imágenes mediante Resonancia Magnética Funcional (fMRI).

La motivación se originó en el proyecto de realización de estudios cerebrales de activación cortical que serán realizados en el Resonador Magnético (RM) de 3 Tesla del Centro Universitario de Imágenes Médicas (CEUNIM), sito en Campus Miguelete de la Universidad de San Martín.

Uno de los paradigmas utilizados en fMRI incorpora estímulos auditivos. Dicho paradigma permite evaluar el funcionamiento de la corteza auditiva del paciente ante diversas patologías (por ej. tumor o trauma).

El desarrollo de este paradigma requiere que el paciente escuche nítidamente estímulos sonoros (música, frases, etc.) y pueda responder verbalmente al operador lo requerido. Luego, un especialista correlacionará las imágenes adquiridas por el resonador con las respuestas de voz del paciente.

Dado que los resonadores magnéticos generan un alto nivel de ruido acústico durante la adquisición de las imágenes, es necesario proveer de *auriculares de gran aislación acústica* al paciente para la escucha de los estímulos sonoros y desarrollar un *sistema efectivo y económico para la captura de la señal de voz*, la cual constituye la respuesta al estímulo. Esto último no es factible mediante un sistema de audio ambiente o micrófono convencional.

Existen en el mundo pocos proveedores de sistemas de audio para fMRI y su costo es elevado. Surgió así la propuesta del Dr. Federico Biafore (UNSAM) de diseñar un *método para la captura de la voz del paciente* y realizar un *adecuado procesamiento digital* de la misma con el fin de hacerla inteligible para el profesional que evaluará las imágenes.

Este proyecto incorporó también la función de grabación de la misma en un medio de almacenamiento portátil para su posterior análisis y correlación con las imágenes obtenidas.

Otras limitaciones que plantea una instalación de resonancia magnética son el intenso campo magnético y la alta sensibilidad a las interferencias de radiofrecuencia que posee el resonador. Estas condiciones limitan los materiales y las técnicas de transmisión de señales que se utilizan dentro de la sala del resonador y, en especial, en el agujero (*bore*) del mismo.



Contenido

1.	Planificación y gestión	9
1.1	Objetivos	9
	Objetivo general.....	9
	Objetivos específicos.....	9
1.2	Definición del Alcance y Propósito del proyecto.....	9
1.3	Requerimientos	11
1.4	Supuestos	11
1.5	Definición de los entregables	12
1.6	Desglose de tareas (WBS).....	12
1.7	Gestión del tiempo	15
1.8	Análisis de riesgos	15
1.8.1	Identificación de riesgos	15
1.8.2	Mitigación de riesgos	16
1.9	Gestión de calidad	17
1.10	Estimación de costos	18
2.	Desarrollo e implementación	19
2.1	Diseño del Módulo Principal.....	20
2.1.1	Etapa analógica.....	20
2.1.2	Etapa digital	27
2.2	Firmware	33
2.2.1	Estructura del firmware.....	33
2.2.2	Descripción de funcionamiento	33
2.3	Integración	38
3.	Ensayos y conclusiones	40
3.1	Consideraciones	40
3.2	Ensayo	41
3.3	Conclusiones	46
4.	Glosario	49
5.	Fuentes	51
	Referencias.....	51
	Bibliografía	51
	Publicaciones	51
	Enlaces	52
	ANEXO I	55
	ANEXO II.....	56



1. Planificación y gestión

1.1 Objetivos

Objetivo general

Diseñar un sistema de audio apto para utilización en el equipo de resonancia magnética de alto campo magnético del CEUNIM (UNSAM)

Objetivos específicos

- Diseñar e implementar un prototipo para la adquisición, procesamiento y almacenamiento en memoria extraíble de la respuesta verbal de un paciente de resonancia magnética nuclear para ser usado en el CEUNIM.
- Especificar la integración de un sistema de audio completo, compatible con resonancia magnética, detallando los componentes requeridos (a diseñar y comerciales) y buscando la optimización de la relación calidad/costo/tiempo de desarrollo.

1.2 Definición del Alcance y Propósito del proyecto

El propósito de este proyecto es desarrollar el prototipo de la *sección de respuesta* de un sistema de audio compatible resonancia magnética y especificar técnicamente el sistema completo, con el fin de hacerlo realizable con componentes comerciales. Una vez concluido el proyecto, con la documentación y los entregables correspondientes, se hará la presentación del mismo ante las autoridades pertinentes de la Escuela de Ciencia y Tecnología (ECyT), con el fin de obtener el título de grado en Ingeniería Electrónica.

El Módulo Principal (MP) posee dos funciones primarias, descriptas a continuación:



- Adaptar, digitalizar, procesar y grabar las respuestas verbales del paciente que es escaneado en un Resonador Magnético (RM). Esta señal es procesada dentro del MP, mediante un filtro adaptativo de cancelación de ruido, y luego amplificada para la escucha en vivo por el operador y grabada en una memoria extraíble.
- Amplificar las señales de audio (estímulos) provenientes de una fuente de audio externa y/o del micrófono del operador con el fin de alimentar un transductor electroneumático, que genera el sonido para los auriculares neumáticos del paciente (*).

Otras funciones del MP incluyen el manejo de pulsadores de control y de una pantalla LCD.

El MP forma parte de un sistema mayor, el cual se representa esquemáticamente en la Fig. 1.

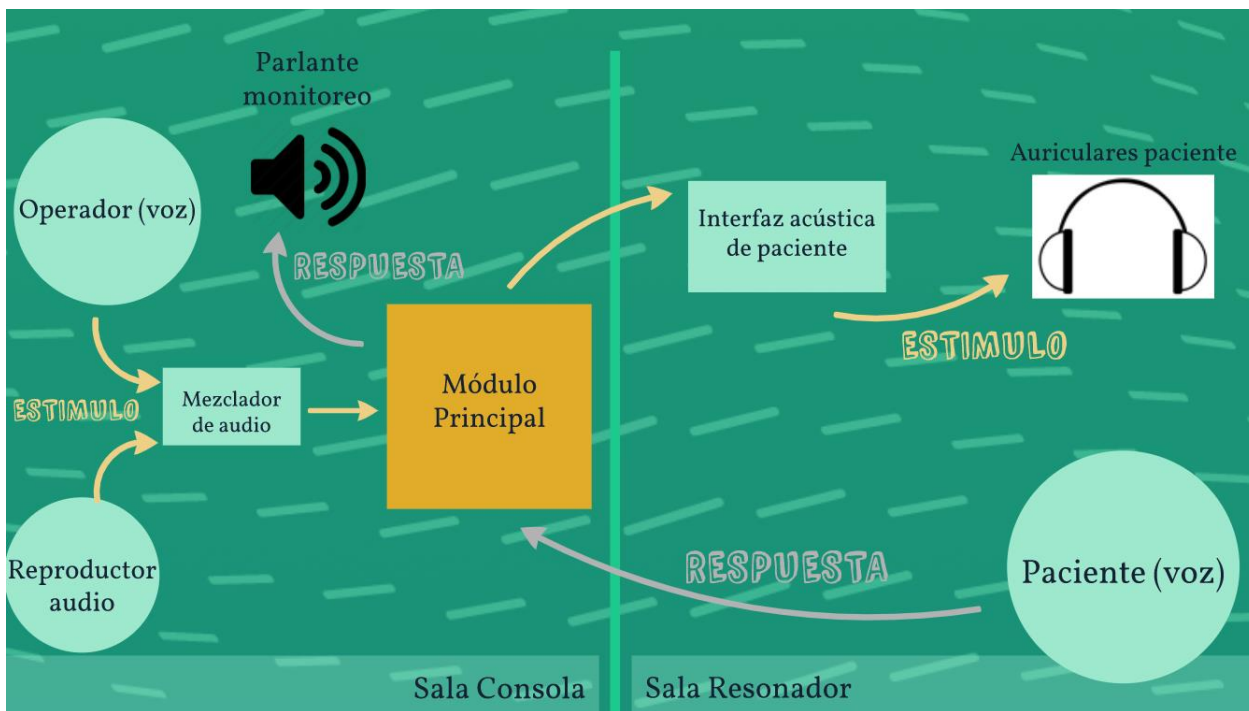


Figura 1: diagrama conceptual del Sistema de Audio compatible con Resonancia Magnética

(*) El diseño de la *interfaz acústica de paciente* y su amplificador de potencia, que constituyen la *sección de estímulo*, exceden al alcance del presente proyecto.



1.3 Requerimientos

Técnicos

- La función digital del MP debe ser llevada a cabo por una tarjeta de desarrollo EDU-CIAA-NXP, operando bajo sistema operativo en tiempo real FreeRTOS.
- La señal de voz debe ser procesada (filtrada) digitalmente por la EDU-CIAA.
- La señal de voz debe
 - ser almacenada en una tarjeta SD y
 - ser reproducida en la sala de consola en tiempo real.
- El archivo de audio debe contener la fecha y hora de inicio de captura de la información, en sincronía con los datos del RM.
- La duración máxima de grabación se especifica entre 15 y 20 minutos.

Administrativos

- El proyecto deberá cumplimentar un mínimo de 200 horas de trabajo, según el requerimiento de la cátedra *Proyecto Final Integrador* de la carrera Ingeniería Electrónica.

1.4 Supuestos

Se supone que

- se realizaron los estudios de factibilidad correspondientes y es posible llevar a cabo el proyecto,
- se cuenta con una tarjeta de desarrollo *EDU-CIAA-NXP* para el desarrollo del prototipo,
- el proyecto cuenta con el aval y aceptación de la Escuela de Ciencia y Tecnología (ECyT) de la UNSAM y el CEUNIM,
- se realizarán, en el RM de alto campo (3T) del CEUNIM, estudios cerebrales por medio de Resonancia Magnética Funcional (fMRI) y
- se tendrá acceso a un RM con el objetivo de realizar las pruebas de campo.



1.5 Definición de los entregables

Documentación:

- Informe final del proyecto
- Manual de uso del Módulo Principal (*sección de estímulo únicamente*)

Hardware:

- Prototipo del Módulo Principal (*sección de estímulo únicamente*)

Firmware:

- Código fuente del firmware desarrollado
- Diagrama en bloques o estados

1.6 Desglose de tareas (WBS)

A continuación, se presenta el desglose en tareas del proyecto presentado. Las tareas desarrolladas se diferenciaron en cinco etapas.

1. Requerimientos

1.1 Investigación de factibilidad

1.1.1 Reuniones con interesados

1.1.2 Investigación en Internet y en biblioteca de la UNSAM

1.2 Supuestos del proyecto

1.3 Desarrollo y redacción del *Project Charter*

1.4 Desglose de tareas (WBS)

1.5 Gestión del tiempo (diagrama de Gantt)

1.6 Análisis de riesgos

1.6.1 Identificación de riesgos y mitigación

1.6.2 Estimación de ocurrencia



2. Diseño del Módulo Principal

2.1 Diseño del firmware (basado en la EDU-CIAA)

2.1.1 Métodos de adquisición/Conversión AD

2.1.1.1 Programación del ADC con DMA

2.1.1.2 Programación del ADC sin DMA

2.1.1.3 Pruebas de transferencia directa ADC-DAC

2.1.1.4 Integración en EDU-CIAA+FreeRTOS

2.1.2 Procesamiento

2.1.2.1 Investigación sobre filtros adaptativos LMS

2.1.2.2 Elección del filtro óptimo

2.1.2.3 Pruebas con filtro en Cortex M3

2.1.2.4 Elección de coeficientes y funciones CMSIS

2.1.2.5 Integración en EDU-CIAA+FreeRTOS

2.1.3 Almacenamiento

2.1.3.1 Investigación sobre implementación de memorias SD en EDU-CIAA

2.1.3.2 Integración en EDU-CIAA+FreeRTOS

2.1.4 Implementación del RTC (*Real Time Clock*) en EDU-CIAA

2.1.5 Visualización

2.1.5.1 Investigación sobre manejo del display LCD 16x2

2.1.5.2 Integración en EDU-CIAA+FreeRTOS

2.1.6 Interfaz de usuario

2.1.6.1 Desarrollo de la experiencia del usuario

2.1.6.2 Integración en EDU-CIAA+FreeRTOS

2.1.7 Integración final en EDU-CIAA+FreeRTOS

2.2 Diseño del hardware

2.2.1 Preamplificador de micrófonos

2.2.1.1 Diseño



- 2.2.1.2 Pruebas en placa
 - 2.2.2 Display LCD 16x2
 - 2.2.2.1 Diseño
 - 2.2.2.2 Pruebas en EDU-CIAA
 - 2.2.3 Amplificador de potencia para parlante de monitoreo
 - 2.2.3.1 Diseño
 - 2.2.3.2 Pruebas en placa
 - 2.2.4 Especificación del amplificador del transductor y el transductor
- 3. Ensayo
 - 3.1 Pruebas en banco
 - 3.1.1 Armado de circuito general en placa de pruebas
 - 3.1.2 Grabación de señales con el prototipo
 - 3.1.3 Mediciones y validación
- 4. Ensamblado del prototipo del Módulo Principal
 - 4.1 Montaje electrónico
 - 4.1.1 Fabricación de placa
 - 4.1.2 Soldado de componentes
 - 4.2 Montaje mecánico
 - 4.2.1 Adaptación del gabinete
 - 4.2.2 Ensamblado en gabinete
- 5. Documentación
 - 5.1 Redacción del informe final
 - 5.1.1 Borrador
 - 5.1.2 Correcciones
 - 5.2 Redacción del manual de usuario
 - 5.2.1 Borrador
 - 5.2.2 Correcciones



1.7 Gestión del tiempo

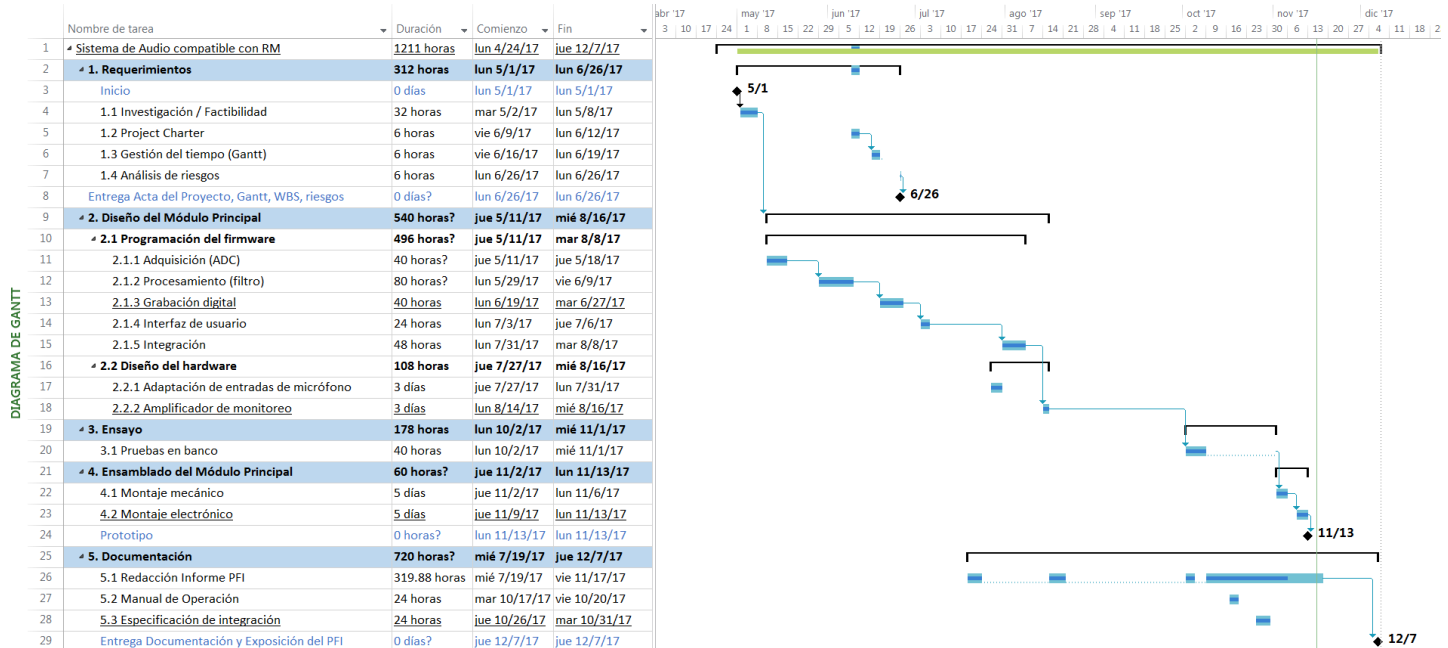


Figura 2: Diagrama de Gantt

1.8 Análisis de riesgos

Se identifican en los puntos 1.8.1 y 1.8.2 los potenciales riesgos del proyecto (R) y la mitigación (M) de dichos riesgos, respectivamente.

En el [ANEXO I](#) se detalla el análisis completo mediante una matriz de riesgos.

1.8.1 Identificación de riesgos

R01: Debido a que se utilizará la tarjeta EDU-CIAA para la función digital, conforme requisito especificado por la Cátedra, podría existir falta de documentación o soporte para la implementación del diseño digital en modo multinúcleo y operando bajo FreeRTOS, siendo ambos



también requisitos de la Cátedra. Si este fuera el caso, los tiempos de desarrollo se verían incrementados, dificultando la entrega en tiempo y forma.

Tipo de riesgo: ALTO

R02: La presencia de alto campo magnético del resonador y el alto nivel de ruido acústico que genera, imponen ciertas limitaciones tecnológicas. Esto podría afectar el funcionamiento del prototipo y consecuentemente el resultado que se desea obtener.

Tipo de riesgo: MODERADO

R03: Demoras en la puesta en marcha del Resonador Magnético (RM). Para las pruebas de campo se necesitará, al momento de los ensayos finales, contar con acceso a la sala del RM del CEUNIM y disponer del equipo en funcionamiento y operado por un técnico durante el lapso aproximado de 3 horas. Esto afectaría básicamente la entrega en tiempo del proyecto.

Tipo de riesgo: ALTO

R04: Cambios en el propósito o alcance del proyecto podrían afectar al tiempo propuesto para la finalización del mismo.

Tipo de riesgo: MODERADO

1.8.2 Mitigación de riesgos

Con el propósito de mitigar los riesgos anteriormente mencionados, se procederá conforme al siguiente plan.

M01: Vinculación e interacción con expertos en sistemas embebidos mediante el [grupo de Google](#) existente para consultas afines y con docentes del proyecto CIAA, la UBA y la UNSAM.

Interacción con los *supporters* de este proyecto, quienes cuentan con experiencia laboral y académica, así como vínculos dentro de áreas afines.

En caso de no poder desarrollarse el proyecto en modo multinúcleo, se evaluará la factibilidad y justificará el diseño utilizando un solo núcleo del procesador LPC4337 de la EDU-CIAA.



M02: Se realizarán pruebas con el software MATLAB usando archivos de audio reales provenientes de sistemas de resonancia similares, con el fin de agilizar la etapa de desarrollo del filtro y lograr la reducción de ruido óptima con herramientas de software.

Luego se investigará acerca de los aspectos o dificultades que puedan presentarse durante la instalación en una sala de resonancia. Se aprovechará el contacto ya existente con ingenieros especialistas en resonancia magnética de la empresa General Electric.

M03: Previo a la redacción del presente informe, se realizaron reuniones con los interesados del proyecto, confirmando plazos de instalación del equipo de resonancia magnética para el mes de octubre de 2017.

En caso de no disponer del equipo del CEUNIM, se realizarán gestiones para acceder a otro resonador de similares características.

En caso de dificultarse la gestión de prueba en un equipo de resonancia magnética, por motivos ajenos al autor de este proyecto y no controlados por él, se buscará evaluar el sistema en banco de pruebas, con la consideración de simular condiciones de uso representativas de la realidad.

M04: Redefinición y reestructuración del alcance de común acuerdo con los interesados. Desdoblamiento del proyecto. Justificación.

1.9 Gestión de calidad

El objetivo del proyecto, en cuanto a la calidad del mismo, es garantizar que tanto el firmware como el hardware sean consistentes en el entorno de trabajo y para las exigencias ante los cuales estarán sometidos.

Mediante los sucesivos ensayos que tuvieron lugar en el banco de pruebas, se buscó mejorar la experiencia de usuario durante la operación del dispositivo. Si bien la navegación a través de los menús y la operación en general resultaron satisfactorias, existe la oportunidad de mejorar la respuesta de los pulsadores de control, mediante la optimización del firmware. Esta mejora deberá llevarse a cabo si el prototipo pasa a fase de producción.

En cuanto al hardware, se buscó integrarlo en el menor espacio y con la menor cantidad posible de cables y conexiones. Esto permitió mejorar la confiabilidad y disminuir eventuales fallas.



La selección de materiales para el gabinete, los cables y los micrófonos fue pensada para que el prototipo pueda probarse en una instalación de resonancia magnética, la cual posee requisitos específicos, como la no utilización de materiales magnéticos dentro de la sala del resonador y el uso de cables coaxiales en el entorno del magneto para evitar interferencias electromagnéticas.

1.10 Estimación de costos

Los siguientes valores se encuentran expresados en pesos argentinos (\$). Dichos montos serán asumidos íntegramente por el autor de este proyecto.

• Costos de componentes (aprox.)	
○ Tarjeta de desarrollo EDU-CIAA-NXP	850
○ Componentes electrónicos (resistencias, capacitores, etc.)	600
○ Display LCD 16x2	150
○ 2 micrófonos de tipo electret	100
○ Tarjeta SD	100
○ Gabinete	300
○ Cables, conectores, fichas, etc.	500
Total	2600



2. Desarrollo e implementación

En este capítulo se detallan el diseño y la teoría de operación del módulo principal (MP) y su integración en un sistema completo de audio compatible con resonancia magnética.

El MP está compuesto por dos secciones independientes denominadas *de estímulo* y *de respuesta*. Esta división hace posible diferenciar las funciones del MP dentro del sistema y organizar el desarrollo del mismo.

Este proyecto consiste en el diseño de la *sección de respuesta*, por ser aquella que demanda mayor tiempo de desarrollo, integra conocimientos adquiridos a lo largo de la carrera y representa un mayor desafío para un proyecto final de Ingeniería Electrónica.

La *sección de estímulo*, excede los contenidos de este trabajo, mas representa un estudio de potencial interés para las disciplinas física y biomédica, ya que incorpora una *interfaz acústica de paciente* (transductor) la cual, debido a su alto costo e inexistencia en el mercado nacional, sería ventajoso desarrollar localmente.

Del punto de vista electrónico, dentro de la *sección de respuesta* existen dos etapas, la *etapa digital* y la *analógica*. Ambas etapas se vinculan mediante una interfaz analógica (IA) que fue diseñada a lo largo de este proyecto.

La IA consiste en una tarjeta con componentes electrónicos que se encarga de acondicionar las señales provenientes de los dos micrófonos localizados dentro del magneto para ingresarlos en la tarjeta de procesamiento EDU-CIAA y, a su vez, amplificar la señal de audio de la EDU-CIAA para alimentar al parlante de monitoreo en la sala de consola.

Respecto de las conexiones externas, el MP posee dos *entradas* encargadas de

- recibir las señales de **estímulo sonoro**, provenientes de un reproductor de audio convencional y/o un micrófono de pie accionado manualmente por el operador, y
- recibir las señales provenientes de dos micrófonos colocados en la proximidad del paciente. Uno de ellos capta la **respuesta** del paciente, sumado al ruido acústico del resonador magnético (RM). El segundo micrófono captura solo el ruido acústico del RM.

Las *salidas* del MP cumplen las funciones de

- amplificar las señales para alimentar la *interfaz acústica de paciente* o transductor, que alimenta a los auriculares neumáticos, y
- amplificar el parlante de monitoreo del operador.



En la Fig. 3 puede observarse un diagrama funcional del sistema completo, con las secciones de *respuesta* y *estímulo* diferenciadas, así como las etapas analógica y digital, la IA y los componentes externos.

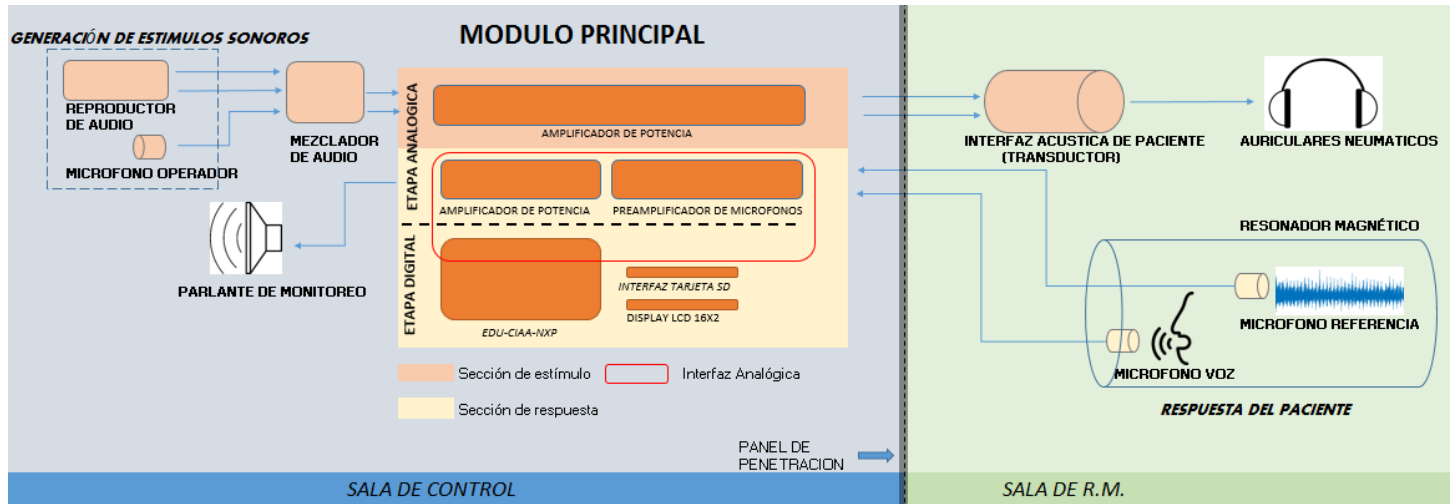


Figura 3: diagrama funcional del Sistema de audio para resonancia magnética

2.1 Diseño del Módulo Principal

2.1.1 Etapa analógica

2.1.1.1 Sección de respuesta

Adquisición de la señal de voz. Preamplificador de micrófonos.

Dos señales son necesarias para el óptimo procesamiento y filtrado de la voz del paciente. Estas señales provienen de sendos micrófonos que capturan la voz del paciente (sumado al ruido de operación del resonador magnético) y la referencia de ruido. Estas se adquieren mediante micrófonos de tipo electrostático o *electret*.

Los micrófonos electret tienen estructura de plástico y aluminio, es decir que son no magnéticos, haciéndose adecuados para colocarse en las proximidades del paciente.



El cableado utilizado para los micrófonos, ubicado entre el resonador y el módulo principal (MP), debe ser de tipo coaxial con el fin de prevenir la interferencia electromagnética hacia las antenas de captura del resonador (bobinas) durante la adquisición.

Las señales provenientes de los micrófonos se adaptan mediante un circuito preamplificador situado en la interfaz analógica (IA), para ingresar en la EDU-CIAA por los canales 1 y 2 del ADC0 (Fig. 4).

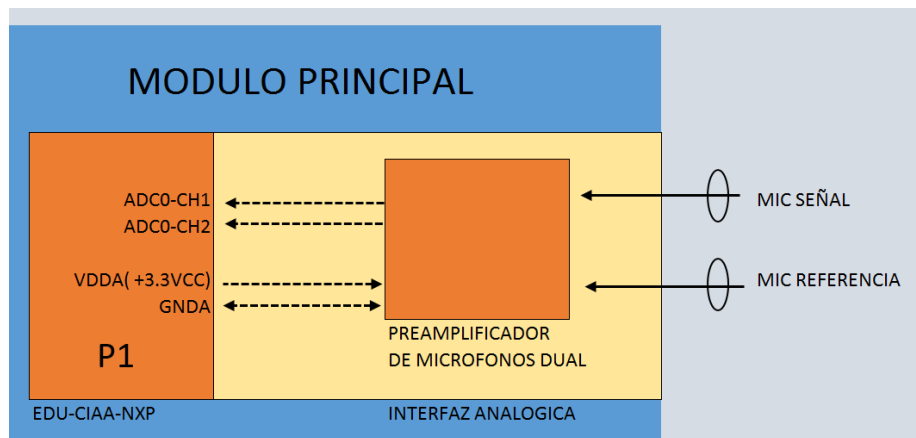


Figura 4: señales del preamplificador de micrófonos

El preamplificador consta de un circuito integrado operacional dual *MCP6002* “rail-to-rail”. Dicho operacional es de fuente única y puede ser alimentado con tensiones bajas, haciéndose ideal para ser alimentado a través de la EDU-CIAA por medio de su salida VDDA, de +3.3V en el conector P1.

El circuito permite adaptar la señal alterna sin continua de 100mVp-p proveniente de los micrófonos, en una señal alterna montada en una continua de +1.65V, y cuya amplitud varía entre 0 y +3.3V, como lo requiere la entrada del ADC.

En la Fig. 5 puede observarse uno de los dos circuitos preamplificadores de micrófonos. El amplificador operacional *MCP6002* es dual, es decir que con un solo circuito integrado es posible manejar las dos señales de los micrófonos.

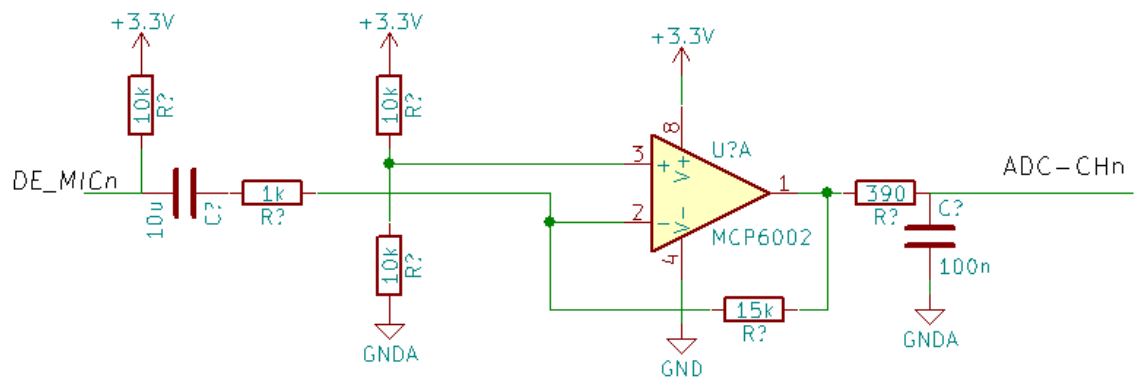


Figura 5: esquemático de medio preamplificador de micrófonos

En la salida del amplificador operacional se colocó un filtro pasabajos de tipo RC (primer orden) con frecuencia de corte de 4kHz a modo de antialiasing.

Amplificación de la señal de voz del paciente. Amplificador de potencia.

El MP incorpora un parlante para que el operador pueda monitorear, a medida que va transcurriendo el estudio, las respuestas del paciente que son procesadas digitalmente y en tiempo real por la *EDU-CIAA*, como se explicará luego en la sección 2.2.

El amplificador del parlante de monitoreo se basa en el circuito utilizado en la placa de extensión de audio del proyecto *CIAA*.

Posee un amplificador operacional LM386 configurado para que su ganancia sea de 20 como lo indica su nota de aplicación de este integrado. [1]

Este amplificador recibe la señal proveniente del DAC, la cual es filtrada mediante un filtro pasabajos con frecuencia de corte de 4kHz y además es desacoplada antes de ingresar por el terminal no inversor, es decir, se elimina su componente continua de 1.65V.

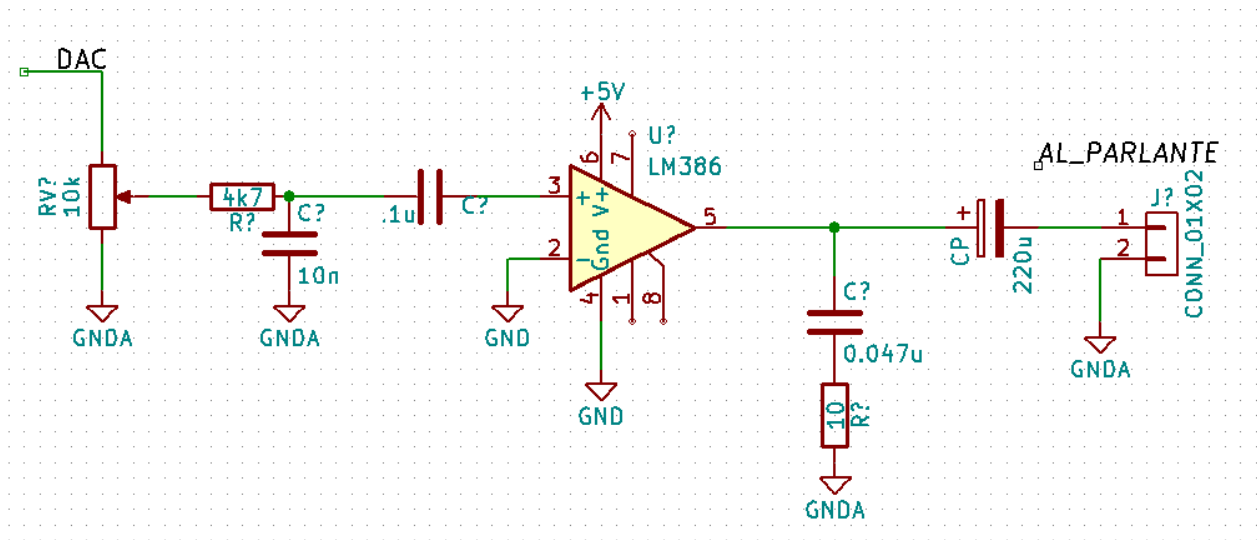


Figura 6: esquemático del amplificador para el parlante de monitoreo

El volumen de monitoreo es ajustable mediante el potenciómetro colocado a la salida del DAC.

La impedancia de salida es de 8 Ohms y la potencia máxima que desarrolla el amplificador es de 275mW. El consumo en reposo es de 4mA.

Interfaz analógica

Los amplificadores de micrófonos y del parlante, así como las conexiones con el potenciómetro de monitoreo, la alimentación de retroiluminación de la pantalla LCD y los pulsadores de comando del MP fueron ubicados en una única tarjeta electrónica, denominada Interfaz Analógica (IA), con el fin de integrar en un pequeño tamaño y darle robustez al hardware.

En la Fig. 7 puede observarse el diagrama de interconexiones de la EDU-CIAA, con la IA, la interfaz para la tarjeta SD y la pantalla LCD 16x2.

La Fig. 8 muestra el circuito esquemático completo de la IA.

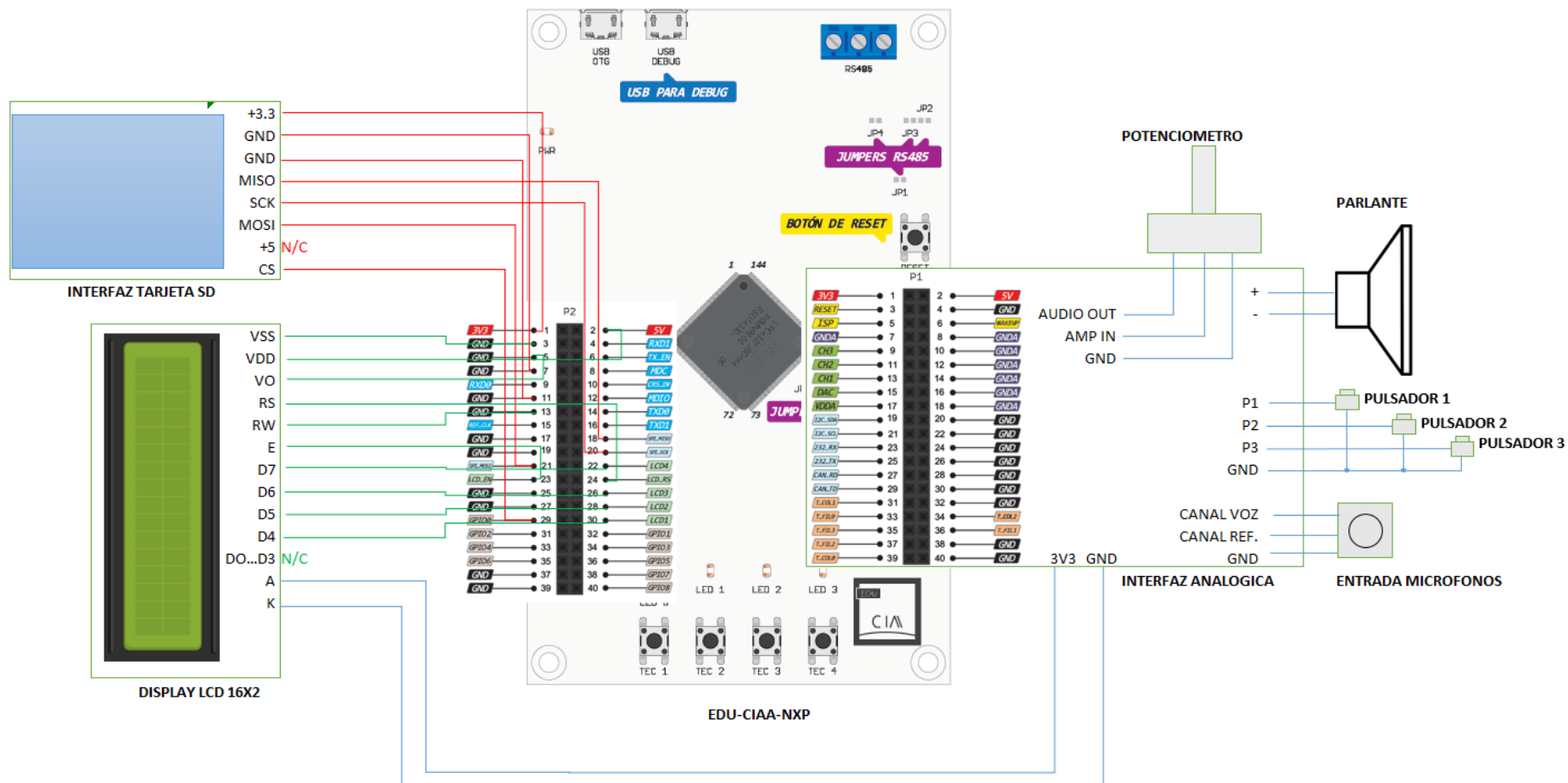


Figura 7: interconexiones de la EDU-CIAA

2.1.1.2 Sección de estímulo

El módulo principal posee una entrada estéreo donde se le inyecta la señal proveniente de un mezclador de audio comercial. Dicha señal puede contener tanto música funcional, estímulos sonoros (ruidos, efectos), como las indicaciones o estímulos de voz proporcionadas por el operador.

La señal estéreo es amplificada en el MP (en la etapa analógica) y luego alimenta a la interfaz acústica del paciente o transductor electroacústico, que a su vez alimenta a los auriculares neumáticos (Fig. 9).

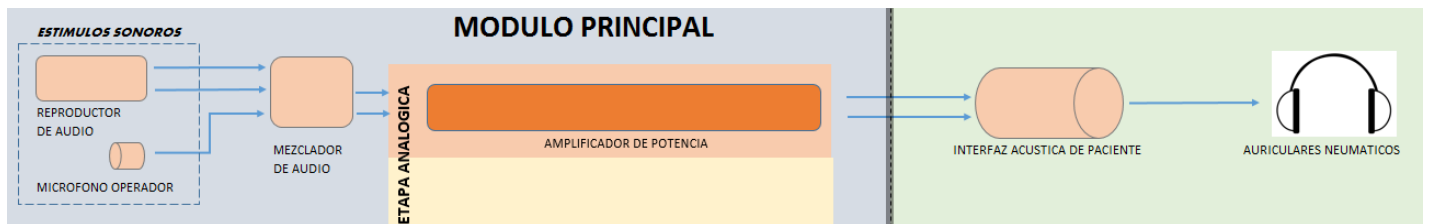


Figura 9: diagrama funcional de la Sección de estímulo

El transductor electroacústico se sitúa en la sala del resonador magnético y es el encargado de convertir las señales eléctricas de los canales de audio izquierdo y derecho, que llevan el estímulo, en ondas acústicas (Fig. 10).

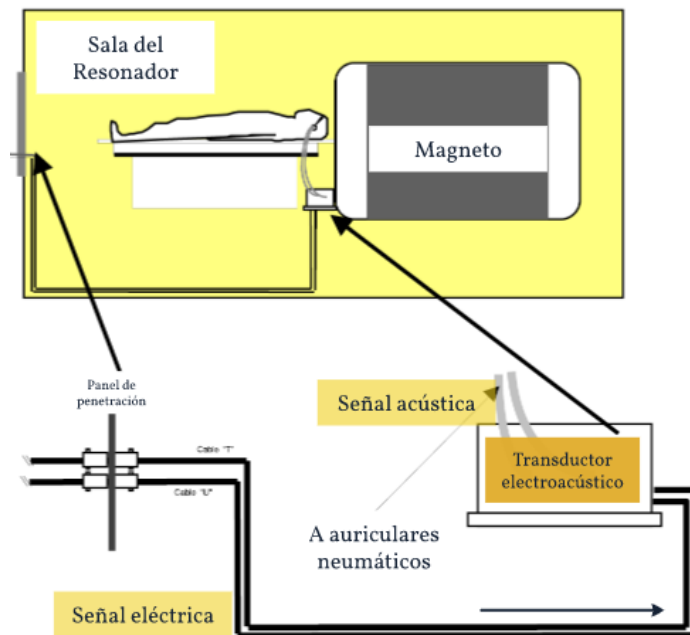


Fig. 10: ubicación del transductor electroacústico



El transductor consiste en un gabinete con dos cavidades aisladas acústicamente, donde se sitúan los parlantes. La aislación acústica entre las cavidades es muy importante para evitar la interferencia entre los canales y conservar la imagen estéreo sin distorsión provocada por el canal adyacente. Es importante destacar esto, ya que los estímulos sonoros de los paradigmas utilizados podrían ser creados para que el paciente distinga entre ambos oídos o para realizar diversos estudios audiométricos.

El gabinete también debe estar aislado respecto del exterior, para evitar la filtración del audio hacia la sala del resonador.

Las salidas de audio del transductor consisten en dos orificios de 0.5cm, en los cuales se conectan los tubos acústicos que forman parte de los auriculares neumáticos.

Para alimentar el transductor se requiere un amplificador de potencia para audio estéreo de 50 a 70W. Este amplificador debe ser estéreo.

Como se comentó al inicio del capítulo, el diseño del amplificador y el transductor excede al alcance del presente proyecto.

2.1.2 Etapa digital

La función digital es realizada por la tarjeta de desarrollo EDU-CIAA-NXP. La EDU-CIAA posee un microcontrolador LPC4337 de doble núcleo (ARM Cortex M0/M4). Este micro está especialmente diseñado para aplicaciones embebidas. El Cortex M4 es un núcleo de 32bits que ofrece bajo consumo, facilidad para el *debugging* y alto nivel de integración para el soporte de bloques. Además, soporta instrucciones para el procesamiento digital de señales o DSP, las cuales son aprovechadas en este trabajo. [2]

El programa es ejecutado íntegramente por el núcleo M4. El núcleo Cortex M0, especialmente diseñado como un procesador de bajo consumo que sirve de reemplazo para los procesadores de 8 y 16 bits existentes y con instrucciones y código reducidos, no fue utilizado con el objetivo de simplificar el diseño del firmware y administrar eficientemente los tiempos de desarrollo para la concreción de este proyecto. En la integración final del firmware, se comprobó que el núcleo M4 cumplía con el objetivo propuesto.

El firmware fue programado en varias etapas hasta llegar a la integración final.

Según requerimiento del proyecto, el firmware fue diseñado para correr bajo el sistema operativo en tiempo real FreeRTOS.



2.1.2.1 Digitalización de las señales-respuesta

Uno de los objetivos del presente proyecto es capturar las respuestas verbales que emite el paciente como respuesta a determinados estímulos, que pueden ser instrucciones, sonidos o imágenes. Estas respuestas resultan enmascaradas por el ruido proveniente del propio resonador magnético (RM) cuando está en operación, el cual no es posible reducir de forma pasiva sin eliminar la señal útil de voz.

Durante un estudio de fMRI y de resonancia en general se producen más de 100dBA de ruido acústico en un resonador de 3T. Existe una clara dependencia entre la intensidad del campo y el nivel de ruido generado, siendo mayor el ruido conforme aumenta la intensidad del campo magnético del equipo. [3] Este intenso ruido se debe a la vibración de las bobinas de conducción cuando son excitadas por pulsos de corriente, es decir, es una característica inherente del RM.

Como se mencionó en el punto 2.1.1.1, la adquisición de la señal de voz del paciente se realiza con dos micrófonos de tipo electret orientados, uno hacia la boca del paciente, el cual captará la voz sumada al ruido, y otro orientado hacia la fuente de ruido.

Ambas señales se transmiten luego al MP, donde son amplificadas en la IA y digitalizadas en la EDU-CIAA.

La digitalización de las señales la realiza el ADC0 de 10 bits de la EDU-CIAA, por medio de los canales 1 y 2 del mismo.

El ADC fue configurado para muestrear a una frecuencia de 8kHz. Esta frecuencia fue elegida adoptando una solución de compromiso luego de un proceso iterativo durante la etapa de diseño, teniendo en cuenta la calidad de audio requerida, el tiempo de procesamiento demandado por el filtro digital y el tiempo de escritura en la SD.

Según el teorema de Nyquist, la frecuencia de muestreo debe ser mayor al doble que la máxima frecuencia de trabajo para luego poder ser reconstruida sin problemas. El rango de frecuencia utilizado en numerosas aplicaciones de transmisión de voz, como la telefonía, es comúnmente entre 300Hz y 3.4kHz, ya que se ha encontrado allí el principal aporte a la inteligibilidad y reconocimiento de la voz humana. [4] Este hecho sugiere adoptar una frecuencia de muestreo de aproximadamente 8kHz o mayor. En este trabajo se optó por 8kHz como solución de compromiso, como se explicará en la sección 2.1.2.3.

Dado que la aplicación no requiere que el audio sea de alta calidad, es aceptable muestrear el audio con la resolución de 10 bits provista por el ADC de la EDU-CIAA.

Para la ejecución precisa y estable de la captura de muestras analógicas de ambos micrófonos se utiliza una interrupción por *timer* (hardware). En la rutina de atención de esta interrupción, se produce el muestreo de los canales 1 y 2 del ADC0. Dicho muestreo puede ser considerado en simultáneo. Según mediciones de tiempos con osciloscopio, la diferencia temporal de toma de muestras entre el canal 1 y el canal 2 es de solo 13,3µs. Este tiempo corresponde a una



frecuencia de $\sim 77\text{kHz}$, es decir, mucho mayor que la frecuencia de muestreo. Esto es muy importante para no alterar la correlación entre las señales, necesaria para lograr un filtrado óptimo.

2.1.2.2 Procesamiento. Reducción de ruido.

El procesador LPC4337 posee funciones optimizadas para procesamiento de señales (DSP) en lenguaje C. Estas funciones pertenecen a la biblioteca CMSIS-DSP. Esta potente característica es aprovechada en este proyecto, incorporando la tarea de filtrado de la señal en el mismo firmware de la EDU-CIAA, sin requerir componentes externos o un DSP dedicado.

Para el filtrado se seleccionó el filtro de tipo adaptativo, ya que se consideró como el óptimo para obtener los resultados esperados.

Los filtros adaptativos son utilizados en situaciones donde las características o propiedades estadísticas de las señales involucradas son desconocidas o varían con el tiempo. En particular, se eligió el filtro LMS para el filtrado de la señal de voz proveniente del paciente.

El algoritmo LMS (del inglés, *Least Mean Square*) se usa en filtros adaptativos para encontrar los coeficientes del filtro que permiten obtener el valor esperado mínimo del cuadrado de la señal de error, definida como la diferencia entre la señal deseada y la señal producida a la salida del filtro.

El desarrollo matemático del filtro LMS excede al contenido del presente proyecto. Sin embargo, es de interés explicar conceptualmente el funcionamiento del mismo.

Considerando que nuestro modelo físico puede representarse como el sistema de la Fig. 11, un sensor o micrófono primario es posicionado para capturar la señal de voz del paciente, s . Sin embargo, esta señal se encuentra afectada por ruido aditivo no correlacionado $n0$. Un sensor primario captura la señal $d(n) = s + n0$ y un segundo sensor "de referencia" es posicionado para capturar ruido proveniente de la misma fuente que $n0$ pero sin tomar la señal s . Este ruido es representado en la Fig. 11 como $n1$.

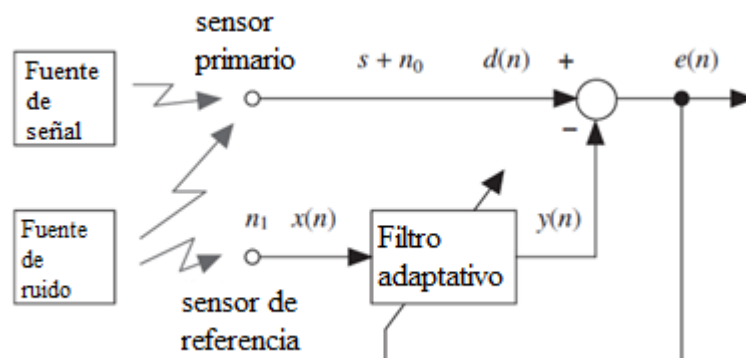


Figura 11: estructura básica de filtro adaptativo configurado para cancelación de ruido

Como se originan en la misma fuente, puede asumirse que las señales de ruido n_0 y n_1 están fuertemente correlacionadas y se asume que el ruido no tiene correlación con la señal s .

En la práctica, el sensor de referencia puede tomar algo de la señal s , con lo cual puede haber correlación entre la señal y el ruido, generando una reducción en el rendimiento del sistema de cancelación de ruido.

El objetivo del sistema de cancelación de ruido es sustraer el ruido aditivo n_0 de la salida del sensor primario ($s + n_0$). El propósito del filtro adaptativo es entonces estimar o derivar n_0 de n_1 .

Finalmente, la salida del sistema será la señal de error cuadrático medio minimizada o $e(n)$, la cual será idealmente igual a s , la señal sin ruido. En la práctica esto no ocurre, pero es posible reducir considerablemente el ruido, lo suficiente para hacer inteligible la respuesta del paciente. [5]

El tiempo total que demanda el procesamiento de los datos es de aproximadamente 10ms y fue medido durante la etapa de desarrollo del firmware.

2.1.2.3 Grabación digital

Uno de los objetivos del proyecto es permitir correlacionar las respuestas verbales del paciente con las imágenes adquiridas durante el estudio fMRI en una etapa posterior a la realización del examen. Para lograr esto, el sistema diseñado debe almacenar estas respuestas en una tarjeta SD.

Un desafío importante consistió en determinar si la escritura en la SD podía ser ejecutada directamente, sin necesidad de un buffer externo, a la frecuencia de muestreo elegida. Para esto, se



midio el tiempo de escritura de datos en la SD, resultando en 100µs por byte. La solución de compromiso adoptada fue transferir a la SD bloques de 500 muestras (de 1 byte c/u) en un tiempo aproximado de 50ms. La digitalización y llenado de un buffer primario (entrada) de 500 elementos de 16 bits, según la frecuencia de muestreo elegida de 8kHz, demora unos 62.5ms. Con lo cual, el tener un tiempo de llenado del buffer de entrada similar al de transferencia a la SD, hizo necesario volcar las muestras en un buffer o vector secundario (salida) una vez finalizada la adquisición de 500 muestras, para evitar el solapamiento de muestras sobre el buffer de entrada.

Esta técnica se aplicó a las dos señales muestreadas por el ADC. De manera que existen dos buffers de entrada y dos de salida.

El procesamiento de los datos (filtrado) es realizado, en un tiempo aproximado de 10ms, con los datos de los dos buffers de salida que contienen las señales digitalizadas de los respectivos micrófonos. Una vez obtenido el vector filtrado resultante (también de 500 elementos), el mismo es convertido a 8 bits y transferido a la tarjeta SD.

De esta manera, mediante la técnica del doble buffer, se evitó la utilización de una memoria externa. La implementación del firmware se detallará luego en la sección 2.2.2.

El ADC de la EDU-CIAA posee una resolución de 10 bits. Con lo cual fue necesario trabajar con datos de 16 bits para el procesamiento y filtrado. Sin embargo, estos datos se convirtieron a 8 bits para ser almacenados en la SD y a su vez sacados por el DAC, para el monitoreo a través del parlante del MP.

Para la transferencia de información entre la EDU-CIAA y la tarjeta SD se utilizaron las salidas y entradas del bus SPI presentes en el conector P2 de la EDU-CIAA (Fig. 12).

El bus SPI es un estándar de transferencia de información en serie entre dispositivos electrónicos que incluye una línea de reloj para sincronización, salida de datos, entrada de datos y selección de chip esclavo/maestro.

Para la grabación de la señal se eligió como medio físico una tarjeta SD-SCHD-Cat.4 de 8Gb de capacidad.

Según requerimiento del proyecto, el tiempo máximo a grabar es de 15 a 20 minutos sin interrupciones.

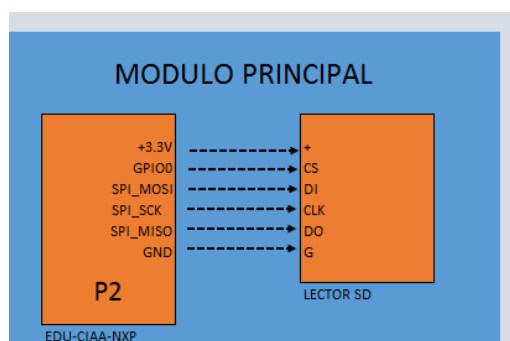


Figura 12: conexión de la interfaz SD a la EDU-CIAA



Para simplificar la programación del firmware, se optó por escribir en la SD un archivo de audio mono de 8 bits y 8 kHz, sin compresión y sin cabecera. Es decir, sin información de codificación embebida en el archivo (frecuencia de muestreo, tasa de bits, cantidad de canales, etc.). La extensión del archivo generado es RAW y la codificación es PCM. El tipo de archivo podría considerarse como un WAV sin cabecera.

Para la conversión del archivo RAW a un formato de audio estándar, se incluyó en la tarjeta SD el programa SOX y un script llamado “crear_wav.bat”, el cual convierte en un solo paso el archivo RAW a WAV. Luego de esta conversión, el archivo WAV con la respuesta del paciente procesada, puede ser abierto en MATLAB/Octave o en un programa de edición de audio.

2.1.2.4 Interfaz del usuario

La interfaz de usuario del MP consiste en

- Pantalla LCD 16x2 caracteres
- Pulsadores de control (P1, P2, P3)

La pantalla o *display* LCD presenta al operador la hora actual al encender el dispositivo. Para este fin se programó el reloj de tiempo real (RTC) de la EDU-CIAA utilizando la SAPI diseñada para tal fin (*sapi_rtc.c*). La hora es conservada aun cuando se interrumpe la alimentación de la tarjeta. Esto es posible ya que la EDU-CIAA incorpora un zócalo para una pila, la cual mantiene energizado el circuito RTC independientemente de la alimentación externa.

Antes de comenzar a grabar, la hora puede ser ajustada por el operador, quien buscará sincronizar la hora del MP con la del resonador, con el fin de poder correlacionar temporalmente audio e imágenes durante la revisión del estudio.

Durante la grabación se observa la hora actual y el mensaje “GRABANDO”.

Una vez dada la orden de terminar la grabación, la pantalla muestra la leyenda “FINALIZADO”, para que el operador sepa que puede extraer de manera segura la tarjeta SD.

El MP posee 3 pulsadores de control. Las funciones de estas teclas son de control y configuración.



2.2 Firmware

El lenguaje elegido para el desarrollo fue C. El firmware fue diseñado en bloques y simulados con la plataforma IDE LPCXpresso individualmente. En la etapa final del diseño se realizó la integración de todos los bloques.

La programación fue realizada bajo sistema operativo en tiempo real FreeRTOS. Se utilizaron cinco tareas más una interrupción por timer.

2.2.1 Estructura del firmware

El programa se estructura de la siguiente forma:

1. Inclusiones de bibliotecas, definiciones, declaraciones de variables y funciones
2. Cuerpo de las funciones y la rutina de atención de interrupción para el *timer* (Prioridad máx.)
3. Implementación de las tareas
 - clock_conf_Task (Prioridad 2)
 - SD_Write_Task (Prioridad 1)
 - diskTask (Prioridad 1)
 - recLEDTask (Prioridad 1)
 - FiltroLMSTask (Prioridad 1)
4. Programa principal (*main*)

2.2.2 Descripción de funcionamiento

Desde el punto de vista del funcional, se describe a continuación el flujo temporal del programa completo.

En la Fig. 13 se detalla el flujo de los datos, desde el muestreo de las señales de los micrófonos hasta las salidas a la tarjeta SD y al DAC.



1. Tarea inicial (*clock_conf_Task*)

Posee la máxima prioridad. Por ende, el microcontrolador solo ejecuta esta tarea al encenderse.

La tarea *clock_conf_Task* posee las instrucciones de configuración de hora (opcional) y la orden de comienzo de grabación.

Una vez finalizada la tarea inicial, se suspende la misma, quedando habilitadas las otras tareas de menor prioridad para su ejecución.

Se habilita la interrupción del timer, para realizar el muestreo/digitalización del audio.

2. Interrupción del timer (*TIMER0_IRQHandler*)

La rutina de atención de interrupción del timer (ISR) se ejecuta 8000 veces por segundo (8kHz) o cada 125µs, y posee la máxima prioridad, es decir, por encima de todas las tareas del sistema operativo.

Las funciones principales son:

- Inicio de conversión y lectura de los canales 1 y 2 del ADC0 (muestreo/digitalización). Un canal contiene la señal de voz del paciente (sumada al ruido del resonador) y el otro contiene sólo el ruido de referencia.

El ADC de la *EDU-CIAA* es de 10 bits. Los valores analógicos son convertidos a digital y guardados en dos *buffers* o vectores de 500 elementos denominados *de entrada*.

- Al llegar a las 500 muestras tomadas cada 125µs, o sea cada 62,5ms, se realiza el volcado de los vectores *de entrada* en dos buffers secundarios o *de salida*.

Luego, los buffers de salida son utilizados por las tareas de procesamiento (*FiltroLMSTask*) y grabación (*SD_Write_Task*), mientras el buffer de entrada comienza a llenarse con los nuevos datos provenientes del ADC. Se buscó una solución de compromiso para que estas tareas (de procesamiento y almacenamiento) puedan finalizar sus respectivas funciones con las 500 muestras dentro de los 62.5ms, como fuera explicado en la sección 2.1.2.3. Este método evita la sobre escritura del buffer primario o *de entrada*, que podría haber ocurrido si se utilizaba un solo buffer.

- Envío al DAC de la señal filtrada por la tarea *FiltroLMSTask* para el monitoreo en tiempo real.



- Verificación de fin de grabación mediante la lectura de estado de uno de los pulsadores de control y habilitación de bandera de cierre de archivo (*cerrar*).

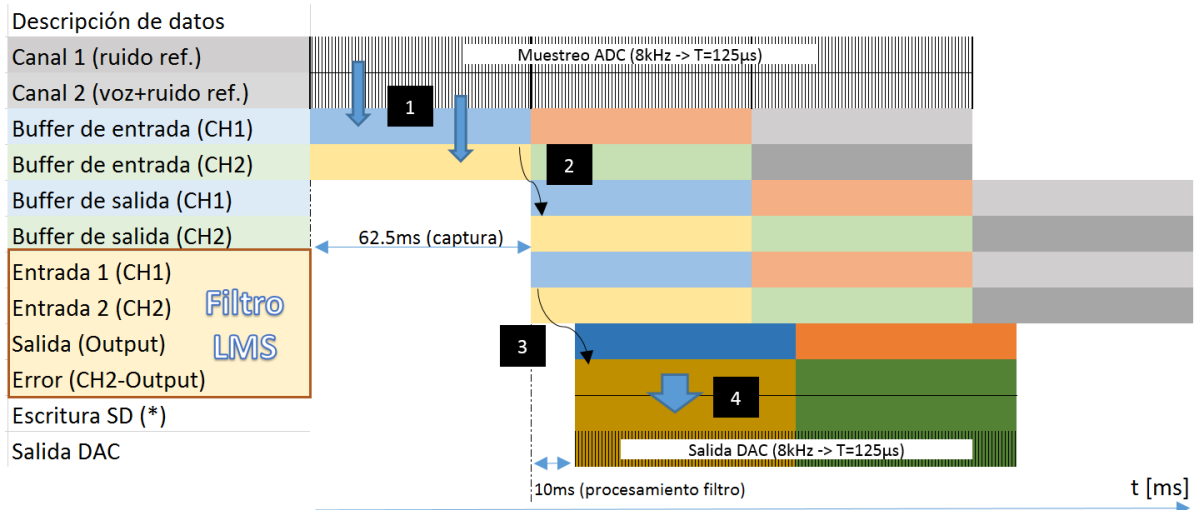


Fig. 13: flujo de los datos

1. Muestreo en el ADC0 a 8kHz, canales 1 (ruido) y 2 (voz+ruido), y llenado de los buffers de entrada.
2. Transferencia de datos de los buffers de entrada a los de salida al llegar a las 500 muestras en 62,5ms.
3. Procesamiento de los buffers de salida en el filtro LMS, en un tiempo aproximado de 10ms, obteniéndose dos vectores de salida: Salida y Error;
4. Vector "Error", resultante de la resta entre la señal del canal 2 (voz+ruido) y el vector Salida (estimación del ruido) del filtro, es transferido a la SD en bloques de 500 elementos y al DAC, dato por dato en la ISR del timer.

(*) Todos los vectores son de 500 elementos de 16 bits, salvo el vector de escritura a la SD, el cual es convertido a 8 bits dentro de la tarea "SD_Write_Task".

3. Tareas de filtrado, grabación y control.

Las siguientes tareas son ejecutadas con igual prioridad por el sistema operativo durante la captura de los datos. El procesador distribuye su tiempo en partes iguales entre ellas según la configuración del tick.

- *FiltroLMSTask*



Se encarga del procesamiento de las señales muestreadas por los amplificadores de la *sección de respuesta*.

Inicializa el filtro (una sola vez), determinando cantidad de coeficientes y otros parámetros, y mientras la bandera *filtrar* esté activa ejecuta la función *arm_lms_q15()* de la biblioteca CMSIS-DSP.

Esta función recibe como parámetros los vectores o buffers de salida (voz+ruido / ruido de referencia) y retorna los vectores salida del filtro (*Output*) y error (*error_aux*). Todos estos son vectores de 500 elementos de 16 bits.

La cantidad de coeficientes fue definida en 8, considerándose como el óptimo valor tras varias iteraciones durante la etapa de prueba del filtro.

La señal filtrada es la señal de error, como se explicó en la sección 2.1.2.2. Esta señal resulta de la resta entre el canal 2, que contiene la voz sumada al ruido y la salida (*Output*) del filtro, que contiene la estimación del ruido.

Activa la bandera *grabar*.

- *SD_Write_Task*

Tarea encargada de convertir el vector *error_aux* de 16 bits, en otro (*error_aux_sd*) de 8 bits, que contiene la señal filtrada.

Además, realiza el volcado de este vector de 500 elementos de 8 bits a la tarjeta SD cuando la bandera *grabar* está activa. Este proceso demora 50ms aproximadamente.

La tarea utiliza funciones de la biblioteca FatFS, especialmente diseñada para sistemas embebidos. Se utiliza el bus SPI de la EDU-CIAA para realizar la transferencia de información entre esta y la tarjeta SD.

El proceso de grabación se repite siempre y cuando la bandera *cerrar* este en 0. Es decir, cuando no hay orden de cierre de archivo.

Cuando la bandera *cerrar* pasa a 1, se produce el cierre del archivo en la SD.

Esta tarea posee una instrucción *vTaskDelay*, la cual hace que la tarea se active cada 10ms. Sin embargo, la escritura en la SD se ejecutará siempre y cuando tenga los datos provenientes del filtro.

- *diskTask*

Ejecuta cada 10ms la función *disk_timerproc()*. Esta función usa 2 timers por software como *timeouts*, para asegurar que el sistema de archivos no se corrompa si



se extrae la SD o falla por otro motivo. También chequea las señales CardPresent y WriteProtect.

Solo corre mientras se está utilizando el sistema de archivos y la SD.

Esta tarea posee una instrucción *vTaskDelay*, la cual hace que la tarea se ejecute cada 10ms.

- *recLEDTask*

Realiza el control del texto que se muestra en pantalla (*GRABANDO/FINALIZADO*), maneja los leds de estado y realiza la suspensión de todas las tareas una vez que la bandera *finalizar* se activa.

Resulta de interés visualizar la ejecución de la interrupción del timer y de las tareas en un diagrama temporal, según son manejadas por el sistema operativo. De esta manera pueden apreciarse las prioridades de cada una y la alternancia entre ellas, desde el arranque del dispositivo hasta la operación del mismo durante el procesamiento y grabación (Fig. 14).

Observar que al principio sólo se ejecuta la tarea *clock_conf_Task* por ser de prioridad 2. Cuando se pulsa el botón para comenzar la grabación, esta tarea se suspende, habilitándose la interrupción del timer (de máxima prioridad) y las tareas 2, 3, 4 y 5, de prioridad 1.

Las tareas 2 y 3 poseen una demora (*vTaskDelay*) configurada en 10ms que las mantiene en el estado bloqueado, ya que no es necesario que permanezcan activas todo el tiempo. Las tareas 4 y 5 se ejecutan todo el tiempo, siendo estas las dedicadas al procesamiento de los datos y a la escritura en la tarjeta SD, respectivamente.

En la Figura 15 se muestra el diagrama de flujo del firmware completo.

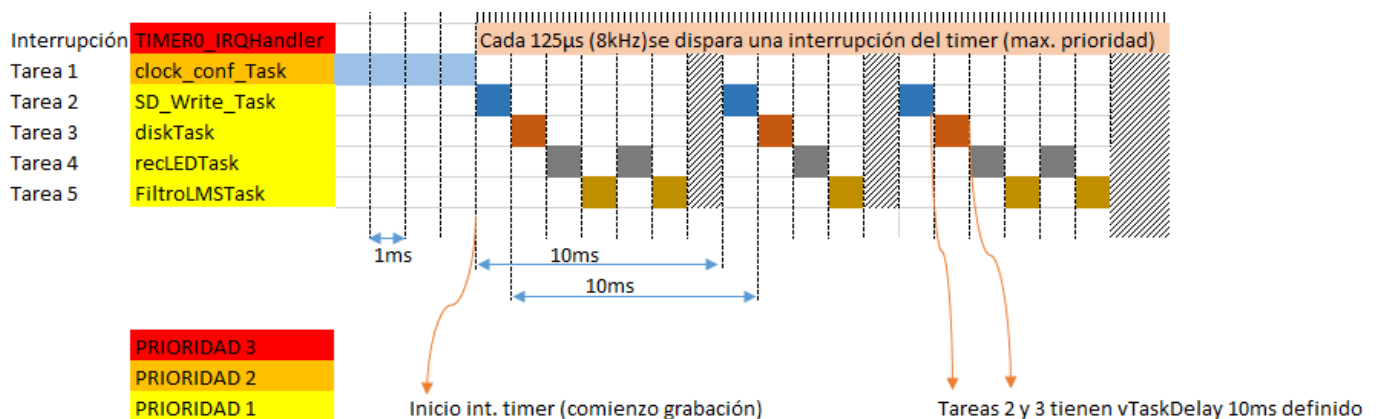


Figura 14: secuencia de ejecución de las tareas e interrupción del timer



2.2.2 Diagrama de flujo

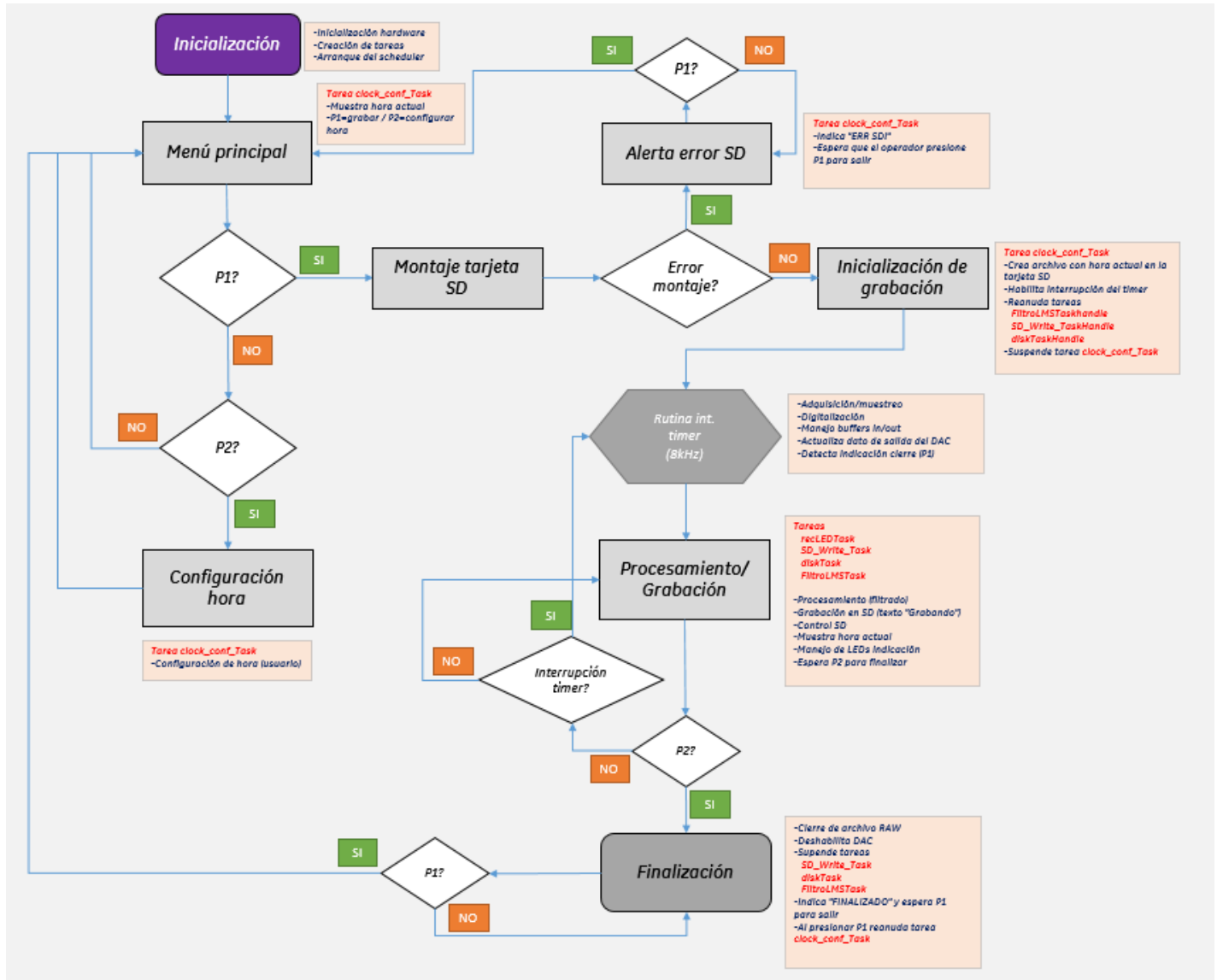


Figura 15: diagrama de flujo del firmware

2.3 Integración

Este sistema de audio especial requiere varios componentes comerciales conectados al módulo principal.



Puede plantearse la integración final del sistema una vez finalizadas las dos secciones que lo componen:

1. *Sección de respuesta:* Diseño y prototipo del bloque de adquisición y procesamiento de respuesta del paciente (este proyecto).
2. *Sección de estímulo:* Diseño y prototipo del bloque amplificador del transductor y del transductor para auriculares neumáticos (futuro proyecto) .

Del punto de vista operativo, es factible incorporar el MP proyectado en este informe (sin la *sección de estímulo*) en una instalación de resonancia magnética que ya posea un sistema de sonido para la escucha del paciente.

A continuación, se detallan los componentes necesarios para conformar el sistema completo.

Componentes comerciales:

- Mezclador de audio de 4 canales, con salida amplificada parlantes de monitoreo y salida maestra estéreo.
- Micrófono de pie
- Reproductor de audio
- Parlantes de monitoreo
- Auriculares neumáticos. Por ej.: *SCANSOUND SKU: P-101*

Componentes a diseñar:

- Módulo Principal
- Interfaz acústica de paciente o transductor.



3. Ensayos y conclusiones

3.1 Consideraciones

En el plan original de este proyecto se consideró realizar pruebas de simulación en banco y pruebas de campo, es decir, estas últimas en una instalación que contará con un Resonador Magnético (RM) de uso clínico. Dado que el RM adquirido por la Escuela de Ciencia y Tecnología (ECyT) todavía no se encuentra instalado y que no pudo conseguirse tiempo de prueba en otra institución que contara con dicho equipamiento, las pruebas de campo resultaron inviables de realizar en el tiempo requerido para la concreción del proyecto.

Pese a esta situación, se presentará en el punto 3.2 un método para la prueba del prototipo que permite fundamentar el correcto funcionamiento del mismo y generar las conclusiones correspondientes en el punto 3.3.

Antes de pasar a la fase de producción, el prototipo deberá evaluarse en el ámbito real. A lo largo del presente proyecto se consultaron distintos especialistas en instalaciones de resonancia magnética.

A continuación, se presentan los aspectos técnicos más importantes a ser considerados para futuras pruebas de campo:

- Los cables de los micrófonos deberán ser de tipo coaxiales para evitar interferencia electromagnética durante la captura de imágenes y a su vez, deberán colocarse dentro de caños flexibles plásticos. Los cables nunca tienen que hacer contacto con el paciente, ya que, debido a la inducción, puede producirse el calentamiento del mismo.
- El pasaje de los cables entre la sala del operador, o sala técnica, y la sala del RM, deberá hacerse a través del panel de penetración o de conexiones del RM. Dicho panel deberá poseer dos conectores tipo BNC libres, para permitir la conexión de los cables de micrófono. Se deberá evaluar la colocación de filtros de RF en dichos conectores para evitar interferencias electromagnéticas filtradas hacia el interior de la sala del resonador.
- Se deberán evaluar distintas posiciones de micrófono y distancias entre los mismos y la boca del paciente, hasta obtener el resultado óptimo de inteligibilidad de la señal de voz.



3.2 Ensayo

Las pruebas del prototipo se realizaron en un banco de pruebas o simulación, utilizando los siguientes componentes, conectados como se indica en la Fig. 16.

Como se indicó en el punto 3.1, la intención de este ensayo es conseguir evaluar el sistema de forma preliminar y obtener resultados representativos del sistema real.

Los elementos utilizados fueron los siguientes:

- Módulo Principal (sección de respuesta)
- Tarjeta SD
- Computadora portátil
- Micrófonos electret omnidireccionales
- Parlante SONY SRS-X5

Potencia: 5W x 2c. + 10W x 1c.

Respuesta en frecuencia: 60Hz – 15kHz

- Decibelímetro digital (aplicación *Sound Meter* de *ABC Apps*)

Nivel de presión sonora máx.: 90dB

Para la prueba del prototipo se ubicaron los dos micrófonos a una distancia conocida “d” del parlante de prueba.

El ruido acústico del resonador magnético fue reproducido por el parlante conectado a la computadora. Se reprodujo el archivo de audio con el ruido de la secuencia EPI (*Echo-Planar Imaging*) de un resonador magnético de 3T. [6] Esta secuencia es la que se utiliza comúnmente para estudios de cerebro mediante fMRI.

Utilizando una aplicación, previamente calibrada, que emula un decibelímetro digital, se ajustó el nivel de volumen del parlante para que en el punto de captación del micrófono, a la distancia “d” de 5 cm. del parlante, sea de aproximadamente 90 dB de presión sonora.



Los micrófonos fueron posicionados de la siguiente manera: el de referencia orientado directo hacia el parlante y el de voz, a la misma distancia del parlante pero orientado en sentido opuesto (Fig. 16).

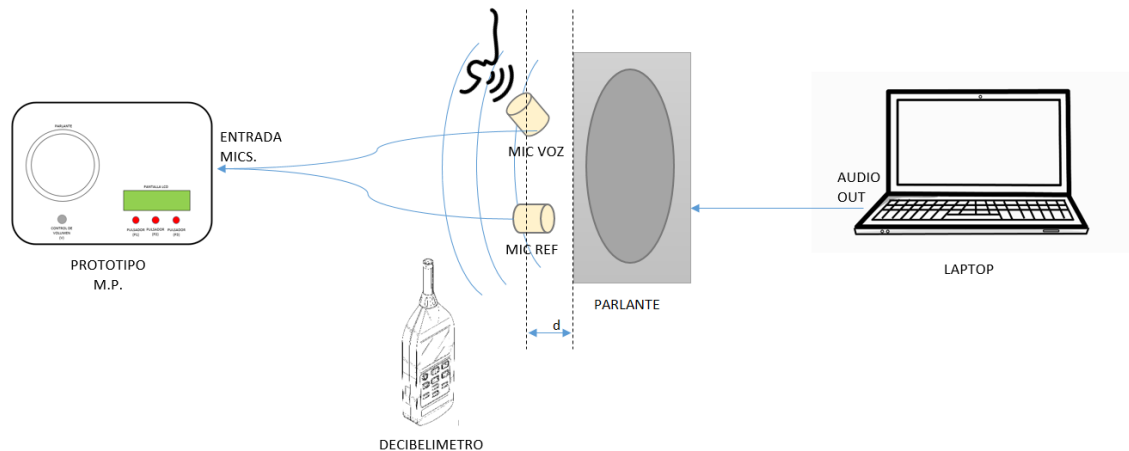


Figura 16: banco de prueba y simulación para la sección de respuesta

Se generaron cuatro archivos de audio WAV para luego ser analizados en MATLAB.

En primer término, se reprodujo el ruido de la secuencia EPI y éste fue grabado por el dispositivo en la tarjeta SD, primero sin filtrar y luego con el filtro activado. Se generaron los archivos *EPI.wav* y *EPI_filtrado.wav* respectivamente.

Luego, se procedió a grabar una frase hablada con el micrófono de voz, mientras el parlante seguía emitiendo el ruido EPI. Bajo esta condición se reprodujo experimentalmente la situación real, donde el micrófono de voz captura la señal de voz sumada al elevado ruido del resonador. El micrófono de referencia y el filtro LMS fueron desactivados. Se generó el archivo *EPI_voz.wav*.

Finalmente, se grabó la misma frase, pero con el micrófono de referencia activado y orientado hacia el parlante, evitando la incidencia directa de la voz, y el filtro LMS activado. Se generó el archivo *EPI_voz_filtrado.wav*.

En la siguiente tabla, se detallan las distintas adquisiciones tomadas:

Distancia “d” (cm)	Nivel de ruido estimado (dBSPL)	Micrófono voz activado	Micrófono ref. activado	Presencia de voz	Filtro LMS activo	Nombre del archivo
5	85-90	SI	NO	NO	NO	EPI.wav
			SI		SI	EPI_filtrado.wav
		SI	NO	SI	NO	EPI_voz.wav
			SI		SI	EPI_voz_filtrado.wav



Antes de cada prueba se aseguró conservar la distancia a los micrófonos, el nivel de ruido y el nivel de la voz hablada.

Los audios resultantes se procesaron gráficamente mediante MATLAB.

La Figura 17 corresponde a la forma de onda (dominio temporal) del ruido de la secuencia EPI, antes (rojo) y después (azul) de aplicar el filtro LMS, graficadas con idéntica escala vertical. En la señal en azul se observa que el filtro ajusta sus coeficientes en los 500ms iniciales, reduciendo la amplitud de la señal, como es esperado.

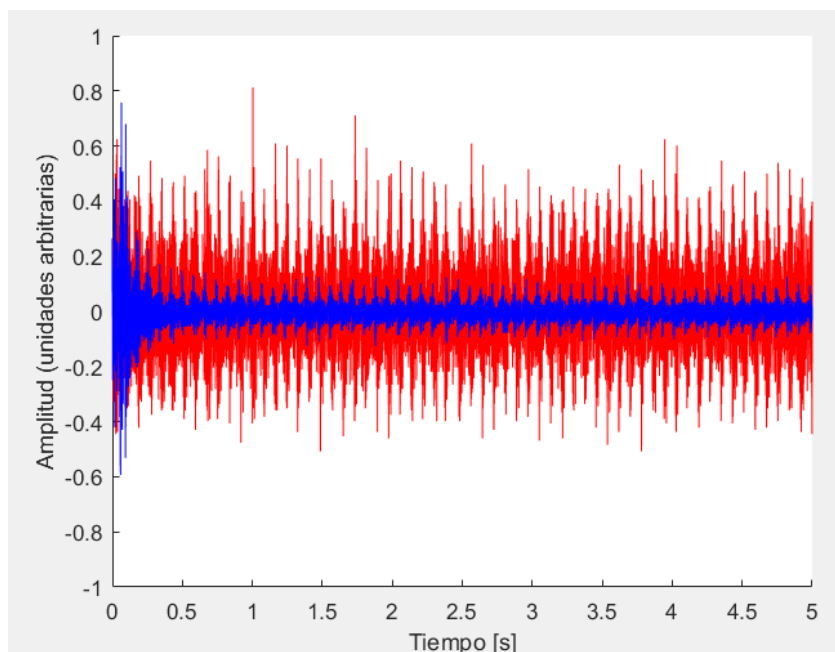


Figura 17: captura del ruido EPI en el dominio temporal, antes (rojo) y después (azul) de la aplicación del filtro LMS (EPI.wav / EPI_filtrado.wav)

Para el caso del espectro de frecuencia, se graficó sobre el eje vertical el módulo de la transformada de Fourier. Los límites de la escala frecuencial tienen en cuenta el teorema de muestreo. Dado que la frecuencia de muestreo es de 8000Hz, la frecuencia máxima de la escala horizontal se delimitó a 4000Hz.

La Figura 18 ilustra en el dominio frecuencial, el ruido de la secuencia EPI con el filtro LMS desactivado y activado. Se observa una reducción de nivel en todo el espectro.

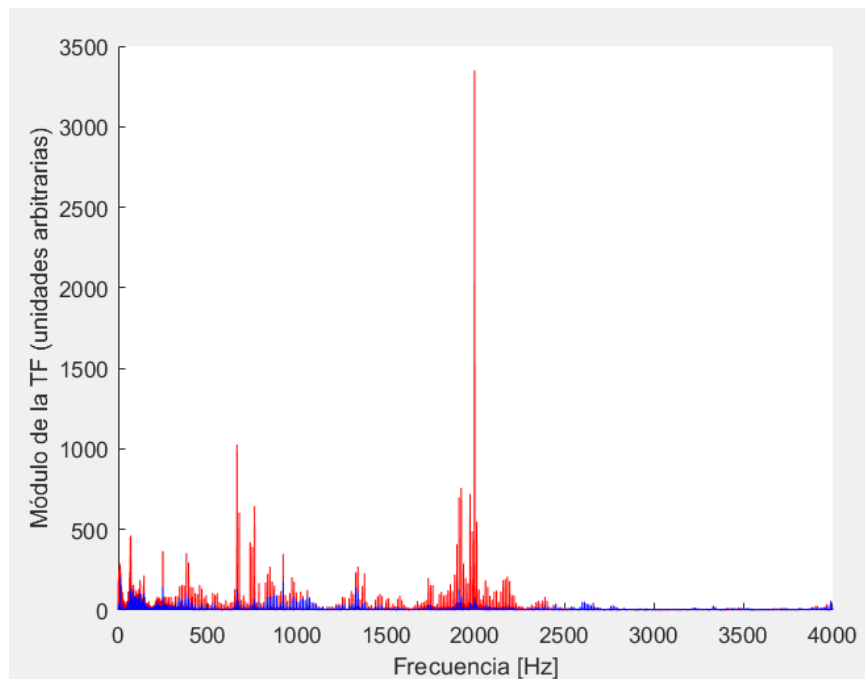


Figura 18: espectro de frecuencia de secuencia EPI antes (rojo) y después (azul) de la aplicación del filtro LMS (EPI.wav / EPI_filtrado.wav)

Para cuantificar el nivel de reducción de ruido bajo las condiciones descriptas anteriormente, se procedió a calcular el valor eficaz o RMS de las señales, antes y después de la aplicación del filtro, en el dominio temporal. Luego se realizó el cociente de dichos valores para obtener la magnitud aproximada de la reducción de ruido, resultando en un valor de 3,6. Es decir, la señal de ruido filtrada sufrió una reducción en valor eficaz de 3,6 veces (más de 11dB) tras la aplicación del filtro LMS.

A continuación, se procedió con la grabación de la voz sumada al ruido, primero sin aplicar el filtro LMS y luego aplicando el mismo. Con el fin de alinear en el tiempo ambas grabaciones, se utilizó un metrónomo para indicar el comienzo de cada palabra.

Los gráficos de la Figura 19 representan las dos señales grabadas en el dominio temporal. Puede evidenciarse, para el caso de la grabación con el filtro LMS aplicado (en color azul), la presencia de las distintas palabras de la frase: "Esto es una prueba". En cambio, en la señal sin el filtro (en rojo) no se distinguen las palabras, ya que se encuentran enmascaradas temporalmente por el ruido.

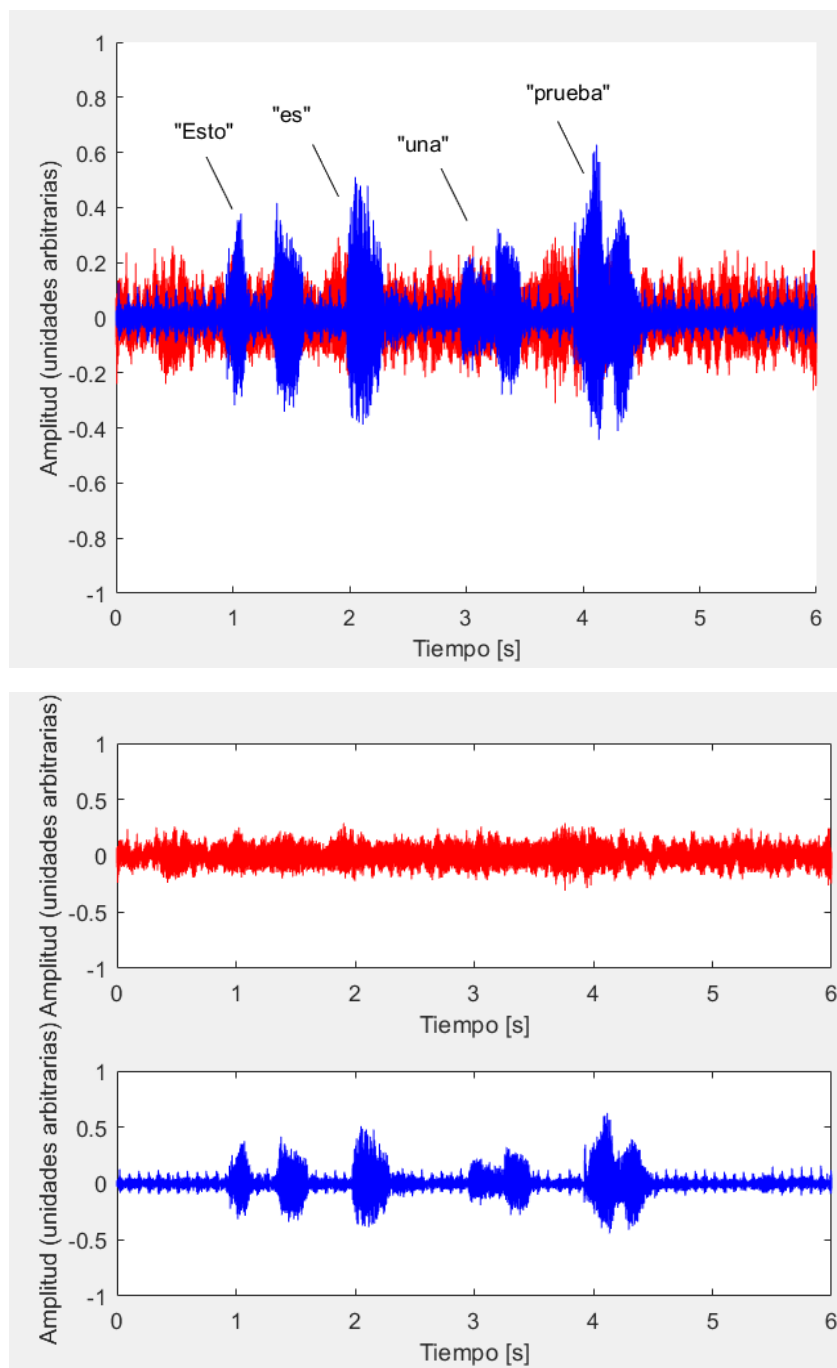


Figura 19: Arriba - formas de onda superpuestas de la prueba de voz inmersa en ruido de secuencia EPI, antes (rojo) y después (azul) de la aplicación del filtro

*Abajo – señales de voz antes (rojo) y después (azul) del filtrado, sin superponer
(EPI_voz.wav / EPI_voz_filtrado.wav)*



Con los resultados obtenidos, se puede concluir que el sistema opera correctamente, recuperando la inteligibilidad de una señal de voz inmersa en ruido generado por un resonador magnético operando con la secuencia EPI.

El método de ensayo propuesto se asume como representativo del caso real. Sin embargo, debido a las limitaciones del ensayo propuesto, es fundamental testear el prototipo en el escenario real. Esto permitirá realizar las pruebas cuantitativas que originarán las especificaciones técnicas del equipo.

3.3 Conclusiones

El objetivo inicial de este trabajo de grado fue desarrollar un sistema de audio compatible con resonancia magnética utilizando componentes diseñados y comerciales. Se consiguió especificar dicho sistema y se logró implementar y evaluar cualitativamente la *sección de respuesta* del mismo, obteniendo resultados satisfactorios en un banco de pruebas.

Una de las condiciones impuestas por la cátedra fue utilizar la EDU-CIAA para realizar la función digital. La EDU-CIAA posee el microcontrolador LPC4337, el cual no está pensado para procesar señales de audio de alta calidad en tiempo real, ya que cuenta con conversores analógico-digitales de 10 bits de resolución. Sin embargo, como la señal a grabar es la voz humana y puede adoptarse una frecuencia máxima de 4kHz, la grabación del audio se implementó usando una frecuencia de muestreo de 8kHz y 8 bits de resolución. Con estas especificaciones, la EDU-CIAA consigue muestrear y procesar en tiempo real las dos señales de los micrófonos exitosamente.

Se implementó el firmware en un sistema operativo en tiempo real, condición también requerida por la cátedra.

Otro requisito fue realizar algún tipo de procesamiento de señales, condición cumplida por el filtro digital adaptativo implementado con funciones de la biblioteca CMSIS-DSP.

Para el procesamiento del audio fue necesario realizar una investigación acerca de sistemas embebidos, en particular de la EDU-CIAA, de sistemas operativos en tiempo real y de filtros adaptativos en general. Estos temas, de considerable complejidad para el autor de este proyecto, demandaron mayor tiempo que el planificado al comienzo del proyecto.

En el inicio del proyecto se había incluido como objetivo el diseño del amplificador del transductor electroacústico. En la medida que el procesamiento de las señales e implementación del firmware requirieron mayor tiempo de estudio y desarrollo, este objetivo fue suspendido, de común acuerdo con los interesados.



Luego de reuniones con los interesados, se decidió separar el sistema en dos secciones (*de estímulo y de respuesta*). De esta forma se redefinieron los objetivos y se logró el enfoque necesario para concretar el proyecto. Se decidió acotar el alcance del proyecto y el desarrollo a la *sección de respuesta* únicamente, que además resulta de mayor interés para el campo de la Ingeniería Electrónica. La *sección de estímulo*, que incluye el amplificador del transductor y el transductor propiamente dicho, es complementaria de este proyecto y de interés para un futuro desarrollo.

Otra desviación del plan original fue la imposibilidad de probar el prototipo en un sistema real. Esto se debió a que el resonador magnético del CEUNIM tuvo demoras en su etapa de instalación. Tampoco fue viable probar el sistema en otro equipo similar. Sin embargo, se desarrolló un método de simulación del prototipo, con el fin de demostrar el correcto funcionamiento cualitativamente y en un entorno representativo del sistema real.

Los resultados obtenidos con el prototipo fabricado, fueron satisfactorios y de valor potencial para fines de investigación y desarrollo de aplicaciones en fMRI. Además, resulta escalable y adaptable a otros usos, como por ejemplo, para adquirir respuestas táctiles del paciente.

En cuanto a la calidad del sistema, existen oportunidades de mejora y optimización, como la calidad del audio de monitoreo y la experiencia del usuario en cuanto a la operación. Sin embargo, estas condiciones no impiden el uso normal del prototipo. Estas mejoras sí deberían ser consideradas a futuro, previo a la fase de producción del dispositivo.

En resumen, este proyecto permitió integrar y enriquecer sustancialmente el conocimiento, tanto teórico como práctico, en diversas ramas de la ingeniería, como las técnicas digitales, la programación, la electrónica analógica, el procesamiento de señales y la acústica.

Se espera que el dispositivo desarrollado sea puesto a prueba en el resonador magnético del CEUNIM, institución pública que prestará servicios médicos a la comunidad y aportará enseñanza e investigación en el campo de las imágenes médicas.

Cabe mencionar que la tarjeta EDU-CIAA, componente principal del prototipo desarrollado, es de diseño nacional. El montaje en gabinete del prototipo (Fig. 20), el firmware y el hardware asociado fueron realizados por el autor de este proyecto con el apoyo técnico de profesionales de diversas universidades del país.



UNSAM
UNIVERSIDAD
NACIONAL DE
SAN MARTÍN

PROYECTO FINAL INTEGRADOR
INGENIERÍA ELECTRÓNICA
"SISTEMA DE AUDIO COMPATIBLE CON
RESONANCIA MAGNÉTICA"

AUTOR:
JOSE MANUEL MOTTA

AÑO: 2017 / REV. 7



Figura 20: prototipo del Módulo Principal



4. Glosario

ADC: del inglés *analog to digital conversion* o conversión digital-analógica

Bit: del inglés *binary digit* o dígito binario

Buffer: espacio en memoria que se utiliza para almacenar datos de forma temporaria

C: lenguaje de programación

CEUNIM: centro universitario de imágenes médicas

cm: centímetros (unidad de longitud)

CMSIS-DSP: biblioteca de software que posee un conjunto de funciones comunes para procesamiento digital de señales optimizadas para dispositivos basados en Cortex-M

DAC: del inglés *digital to analog conversion* o conversión analógica-digital

dB SPL: decibeles de nivel de presión sonora

Debugging: depuración del programa

DSP: del inglés *digital signal processor* o procesamiento digital de señales

EDU-CIAA: computadora industrial abierta argentina

Electret: micrófono electrostático

EPI: del inglés *echo-planar imaging*

fMRI: del inglés *functional magnetic resonance imaging* o resonancia magnética funcional

GPIO: del inglés *general purpose input/output* o Entrada/Salida de Propósito General

Hz: *hertz* o hercio. Unidad de frecuencia del Sistema Internacional de Unidades.

IA: interfaz analógica

IDE: del inglés *integrated development environment* o entorno de desarrollo integrado

ISR: del inglés *interrupt service routine* o rutina de atención de interrupción

LCD: del inglés *liquid crystal display* o pantalla de cristal líquido

LMS: del inglés *least mean square* o cuadrados mínimos

MATLAB: herramienta de software matemático

MP: módulo principal

ms: milisegundo (unidad de tiempo)

PC: del inglés *personal computer* o computadora personal



Project charter: acta de constitución del proyecto

RF: radiofrecuencia

RM: resonador magnético

RTC: del inglés *real time clock* o reloj de tiempo real

sAPI: del inglés *simplified Application Programming Interface* o interfaz de programación de aplicación simplificada

SD: del inglés *secure drive* o medio de almacenamiento seguro (memoria de almacenamiento de estado sólido)

SNR: relación señal-ruido

SPI: del inglés *serial peripheral interface*

TF: transformada de Fourier

Tick: unidad arbitraria de medición de tiempos de un procesador o sistema operativo

Timer: temporizador, reloj

3T: 3 Tesla, intensidad de campo magnético del resonador

UNSAM: Universidad Nacional de San Martín

WAV: del inglés *WAVE form audio format* o formato de audio de ondas

WBS: del inglés *work breakdown structure* o desglose de tareas

μs: microsegundo



5. Fuentes

Referencias

- [1] TEXAS INSTRUMENTS, Mayo 2004. [Online].
<http://www.ti.com/lit/ds/symlink/lm386.pdf>
- [2] NXP Semiconductors, Marzo 2016. [Online]. https://www.nxp.com/docs/en/data-sheet/LPC435X_3X_2X_1X.pdf.
- [3] J. P. D. W. P. J. P. D. W. P. J. S. C. M. R. K. P. David L. Price MSc, "Investigation of acoustic noise on 15 MRI scanners from 0.2 T to 3 T," 2001.
- [4] 2012. [Online]. <http://www.uoverip.com/voice-fundamentals-human-speech-frequency/>.
- [5] D. Reay, "Digital signal processing using the ARM Cortex-M4," 2015.
- [6] [Online]. <https://www.youtube.com/watch?v=DpL5OqoAMUo>

Bibliografía

- Richard Barry, "The FreeRTOS Reference Manual", *Real Time Engineers*, 2011
- Richard Barry, "Using the FreeRTOS Real Time Kernel", *Real Time Engineers*, 2011
- Stanley E. Portny, "Project Management for Dummies – 3rd Edition", *Wiley*, 2010

Publicaciones

- Zvyagintsev, Klasen, Krystyna Mathiak, Weber, Edgar, Klaus Mathiak, "Real-Time Noise Cancellation for Speech Acquired in Interactive Functional Magnetic Resonance Imaging Studies", *Wiley-Liss*, 2010



- Andrade, Sanches, “An Active Noise Cancellation System for fMRI auditive paradigms”
- Lim, Lee, Li, et al., “Evaluation of MRI compatible headphones for active noise control” Noise Control Engineering Journal, 2013

Enlaces

http://bjtp.ap.health.ge.com/bjtp/manage/files/docs/5366639-8EN_r19/media/1092118-1092119.pdf

http://docs.lpcware.com/lpcopen/v1.03/group_examples_periph_18_x43_x_x_adc.html#ga7522a59ba81bf5fee22e23d48aeb259c

http://docs.lpcware.com/lpcopen/v1.03/group_rtc_18_x43_x_x.html#gafdad7b4e551aa7e6d52b49a7a072f4ae

http://elm-chan.org/docs/mmc/mmc_e.html

http://elm-chan.org/fsw/ff/00index_e.html

<http://godzilla.kennedykrieger.org/fmri/>

<http://hackaday.com/2014/09/30/the-ten-sy-audio-library/>

<http://online-audio-converter.com/es/>

<http://sim.okawa-denshi.jp/en/CRtool.php>

<http://stackoverflow.com/questions/15884566/setting-an-adc-sample-rate-trouble-reading-datasheet>

<http://stackoverflow.com/questions/20039038/setting-sampling-frequency-for-adc-in-arm-cortex-m3>

<http://symbioticdevices.com.au/products/mri-subject-feedback/>

<http://www.edn.com/design/test-and-measurement/4389017/Prepare-signals-for-microcontroller-ADCs>

<http://www.electronicshub.org/arm-tutorial/>

<http://www.fibersound.com/fiberopticmicrophone.html>

<http://www.firmcodes.com/microcontrollers/8051-3/interfacing-lcd-with-8051/lcd-commands-and-understanding-of-lcd/>

<http://www.forosdeelectronica.com/f24/tutorial-freertos-103057/>

<http://www.freertos.org/a00110.html>

<http://www.freertos.org/Debugging-Hard-Faults-On-Cortex-M-Microcontrollers.html>

<http://www.head-fi.org/t/731964/10-bits-of-sample-depth-is-more-than-enough-for-audio-with-tests-that-you-can-perform-to-prove-it-to-yourself>

http://www.hyperhot.com/pm_wbs_sw1.htm

<http://www.instructables.com/id/Arduino-Audio-Input/>

<http://www.instructables.com/id/LM386-Audio-Amplifier/>

<http://www.keil.com/forum/61679/>

<http://www.moty22.co.uk/sd.php>

<http://www.ni.com/tutorial/13569/en/>

<http://www.ocfreaks.com/interfacing-16x2-lcd-with-lpc2148-tutorial/>

<http://www.optoacoustics.com/medical/fomri-iii/features>



<http://www.safibra.cz/en/mri-optical-microphones>

<http://www.stm32duino.com/viewtopic.php?t=1453>

http://www.theaudioarchive.com/TAA_Resources_File_Size.htm

<http://www.wavelengthcalculator.com/wlc-es-calculadora-longitud-de-onda-frecuencia.php>

<https://community.nxp.com/thread/434090>

<https://developer.mbed.org/forum/mbed/topic/2920/>

<https://dsp.stackexchange.com/>

<https://electronics.stackexchange.com/questions/15006/is-is-feasible-to-build-an-audio-recorder-with-sd-card-storage-using-atmega-16-o>

http://elm-chan.org/fsw/ff/00index_e.html

https://es.wikipedia.org/wiki/Algoritmo_LMS

https://exploreembedded.com/wiki/LPC1768:_Timers

<https://forum.allaboutcircuits.com/threads/ad-c-clock-and-sampling-rates.83659/>

<https://forum.arduino.cc/index.php?topic=381051.0>

<https://github.com/adamjvr/makodsp>

https://github.com/ARM-software/CMSIS_5

https://github.com/ciaa/firmware_v2

<https://github.com/epernia/SAPI>

https://github.com/MatiasFerraro/CIAA-MatiasFerraro/blob/nuevo/projects/drivers_bm/src/lcd.c

<https://github.com/pridolfi/workspace>

<https://github.com/samtools/sapi/blob/master/sapi.h>

<https://groups.google.com/forum/#!forum/embebidos32>

[https://groups.google.com/forum/#!searchin/embebidos32/lcd\\$20edu-ciaa%7Csort:relevance/embebidos32/JFtSCpO42gU/zSZ10KhiAwAJ](https://groups.google.com/forum/#!searchin/embebidos32/lcd$20edu-ciaa%7Csort:relevance/embebidos32/JFtSCpO42gU/zSZ10KhiAwAJ)

<https://learningmsp430.wordpress.com/2013/11/16/16x2-lcd-interfacing-in-4-bit-mode/>

https://www.nxp.com/docs/en/data-sheet/LPC435X_3X_2X_1X.pdf

<https://sourceforge.net/projects/sox/files/sox/>

<http://sox.sourceforge.net/>

<https://stackoverflow.com/questions/28024705/from-adc-samples-to-wav-file-in-c>

<https://www.arde.cc/foro/viewtopic.php?t=884>

https://www.keil.com/pack/doc/CMSIS/DSP/html/group__LMS.html

<https://www.lpcware.com/content/project/lpc43xx-dual-core-examples>

<https://www.lpcware.com/content/project/lpc43xx-dual-core-notes>

<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2270941/>

<https://www.quora.com/Are-there-robust-methods-for-imaging-speech-production-in-fMRI-while-recording-the-speech-output>

<https://www.scansound.com/mri-products/pneumatic-mri-headphones>

<http://www.ti.com/lit/ds/symlink/lm386.pdf>



UNSAM
UNIVERSIDAD
NACIONAL DE
SAN MARTÍN

PROYECTO FINAL INTEGRADOR
INGENIERÍA ELECTRÓNICA
“SISTEMA DE AUDIO COMPATIBLE CON
RESONANCIA MAGNÉTICA”

AUTOR:
JOSE MANUEL MOTTA

AÑO: 2017 / REV. 7

<https://www.videograbber.net/sound-grabber.html>

<https://www.youtube.com/watch?v=6Aj2QspPf7s&t=1739s>

ANEXO I

Matriz de riesgos

Probabilidad	Valor numérico	Impacto	Valor numérico	Tipo de riesgo	Probabilidad x Impacto									
Muy improbable	0.1	Muy bajo	0.05	Bajo	<=0.05	Casi certeza	0.9	0.045	0.09	0.18	0.36	0.72		
Relativamente probable	0.3	Bajo	0.1	Moderado	0.05<x<0.18	Muy probable	0.7	0.035	0.07	0.14	0.28	0.56		
Probable	0.5	Moderado	0.2	Alto	>=0.18	Probable	0.5	0.025	0.05	0.1	0.2	0.4		
Muy probable	0.7	Alto	0.4			Relativamente probable	0.3	0.015	0.03	0.06	0.12	0.24		
Casi certeza	0.9	Muy alto	0.8			Muy improbable	0.1	0.005	0.01	0.02	0.04	0.08		
								0.05	0.1	0.2	0.4	0.8		
								Muy bajo	Bajo	Moderado	Alto	Muy alto		

Código del riesgo	Descripción del riesgo	Causa	Afectaciones al proyecto	Probabilidad de Ocurrencia	V	Objetivo afectado	Estimación del impacto	V	Tipo de riesgo	V	Mitigación
R01	Implementación del firmware en modo multicore con FreeRTOS	Falta de recursos, información o soporte	Tiempo de desarrollo afectado	Muy probable	0.7	Alcance	Muy bajo	0.1	Alto	0.63	Interacción con grupos de sistemas embebidos. Exploración de posibilidades con 1 núcleo y justificación.
						Tiempo	Muy alto	0.8			
						Costo					
						Calidad	Muy bajo	0.1			
R02	Filtrado del ruido/Inteligibilidad	Limitación tecnológica	Calidad del audio final	Relativamente probable	0.3	Alcance			Moderado	0.15	Ensayos previos en MATLAB. Simulación en banco de pruebas. Investigación con especialistas de resonancia magnética.
						Tiempo	Moderado	0.2			
						Costo	Bajo	0.1			
						Calidad	Moderado	0.2			
R03	Pruebas de campo	Demora en la puesta en marcha del resonador del CEUNIM	Tiempo de desarrollo afectado	Probable	0.5	Alcance	Moderado	0.2	Alto	0.3	Gestión para pruebas en otro resonador similar. Evaluación en banco de pruebas.
						Tiempo	Alto	0.4			
						Costo					
						Calidad					
R04	Cambios en el propósito o alcance del proyecto	Falta de tiempo para desarrollar la sección de estímulo	Tiempo de desarrollo afectado	Relativamente probable	0.3	Alcance	Moderado	0.2	Moderado	0.12	Redefinición del alcance. Reestructuración del proyecto de común acuerdo con los interesados.
						Tiempo	Moderado	0.2			
						Costo					
						Calidad					

ANEXO II

Código fuente

* @brief MAIN DEL PROTOTIPO DE LA SECCION DE RESPUESTA DEL SISTEMA DE AUDIO COMPATIBLE CON RESONANCIA MAGNETICA

* Implementacion bajo FreeRTOS para EDU-CIAA-NXP.

* Funciones

* Muestra hora actual para sincronizar con MR.

* Toma datos de 2 canales del ADC0 (señal y ruido; ruido), los filtra y los transfiere a la tarjeta SD.

* Saca por DAC la señal procesada proveniente del filtro.

* Escribe en la SD a una frec de 8kHz con un buffer de 500 muestras de 8 bits.

* El formato del archivo es RAW grabado en la SD y se convierte a WAV en la PC mediante el programa SOX.

*

* @author J.M.M.

* @version 3.0

* @date April-October 2017

*/

#include "main.h"

#include "board.h"

#include "sapi.h"

#include "sapi_rtc.h"

#include "sapi_dac.h"

#include "lcd.h"

#include "ff.h"

#include "FreeRTOS.h"

#include "FreeRTOSConfig.h"

#include "task.h"

#include "semphr.h"

#include "queue.h"

#include "portmacro.h"

/* DSPLib */

#include "arm_math.h"

#include "math_helper.h"

#define CMSIS_DSPLIB_CM4

#define Cant_Coef_filtro 8

/*=====macros y definiciones=====*/

//#define FILENAME "adc2sd.raw"

#define ADC_RATE 100000

#define MR 1
// Match Register MR{0:3} --> Channel 0,1,2,3

#define Cant_Muestras 500 //Tamaño del buffer

/*=====definiciones de datos internos=====*/

static FATFS fs; /**< FatFs work area needed for each volume */

static FIL fp; /**< File object needed for each open file */

static uint16_t ch1_signal, ch2_ref; //Muestra de ADC0_CH1 y ADC0_CH2

static uint16_t adcCntr, indice, i;

static uint16_t dac [Cant_Muestras];

static uint16_t ch1_bufferIN [Cant_Muestras]; //Buffer de entrada (ADC0_CH1)

static uint16_t ch2_bufferIN [Cant_Muestras]; //Buffer de entrada (ADC0_CH2)

q15_t ch1_signal_bufferOUT [Cant_Muestras]; //Buffer de salida

q15_t ch2_ref_bufferOUT [Cant_Muestras]; //Buffer de salida

static uint8_t grabar = 0, finalizar = 0, cerrar=0, filtrar = 0, ERROR_SD = 0, salir;

q7_t error_aux_sd [Cant_Muestras];

static UINT nbytes;

static char filename [10] = "HHMMss.raw";

TaskHandle_t recLEDhandle = NULL;

TaskHandle_t FiltroLMSTaskhandle = NULL;

TaskHandle_t SD_Write_TaskHandle = NULL;

TaskHandle_t diskTaskHandle = NULL;

TaskHandle_t closeSDTaskHandle = NULL;



```
TaskHandle_t clock_conf_TaskHandle = NULL;
```

```
rtc_t RTC,RTC_read;
```

```
char hora_dec,hora_uni,min_dec,min_uni,seg_dec,seg_uni;
```

```
char p;
```

```
/* Global attributes */
```

```
q15_t Buffer_sr[Cant_Muestras];
```

```
q15_t Buffer_r[Cant_Muestras];
```

```
q15_t error_aux[Cant_Muestras];
```

```
q15_t State_Buffer[Cant_Muestras+Cant_Coef_filtro-1];
```

```
q15_t Output[Cant_Muestras];
```

```
q15_t coeficientes_del_filtro[Cant_Muestras];
```

```
q15_t u;
```

```
arm_lms_instance_q15 EstructuraDelfiltro;
```

```
/* CANCELADOR DE RUIDO USANDO EL ALGORITMO LMS
```

```
*
```

```
* x(n)=d(n)+v1(n) e(n)=x(n)-est_d(n)
```

```
* -----+----->----->(+)------+----->
```

```
* | ^ | e(n)=st_v1(n)
```

```
* | | |
```

```
* | | |
```

```
* | +-----+ | |
```

```
* | | Algoritmo | | |
```

```
* +----->| L M S +----->+ V
```

```
* | | est_d(n) |
```

```
* +-----+ |
```

```
* ^ |
```

```
* | |
```

```
* | e(n) |
```

```
* +-----<-----+
```

```
*/
```

```
/*=====[declaraciones de funciones  
internas]=====*/
```

```
static void initTimer(void);
```

```
static void mountSDCard( void );
```

```
static void mostrarTexto (char []);
```

```
static void mostrarHora (void);
```

```
/*=====[declaraciones de  
tareas]=====*/
```

```
// Prototipo de funcion de la tarea clock_conf_Task
```

```
void clock_conf_Task( void *pvParameters );
```

```
// Prototipo de funcion de la tarea clock_conf_Task
```

```
void clockTask( void *pvParameters );
```

```
// Prototipo de funcion de la tarea SD_Write_Task
```

```
void SD_Write_Task( void *pvParameters );
```

```
// Prototipo de funcion de la tarea diskTask
```

```
void diskTask( void *pvParameters );
```

```
// Prototipo de funcion de la tarea recLEDTask
```

```
void recLEDTask( void *pvParameters );
```

```
// Prototipo de funcion de la tarea FiltroLMSTask
```

```
void FiltroLMSTask ( void *pvParameters );
```

```
/* Hardware initialization */
```

```
static void initHardware(void)
```

```
{
```

```
ADC_CLOCK_SETUP_T adc_setup;
```

```
Board_Init();
```

```
//HABILITAR GPIO0 (P6_1 <--- GPIO3[0]) COMO PIN DE SALIDA  
PARA EL TOGGLE
```



```
Chip_SCU_PinMux(3, 0, SCU_MODE_INACT |  
SCU_MODE_INBUFF_EN, SCU_MODE_FUNC0); /* EDU-CIAA GPIO0  
=> GPIO3[0], SALIDA PARA TEST */
```

```
Chip_GPIO_SetPinDIROutput(LPC_GPIO_PORT, 3, 0);
```

```
Chip_GPIO_SetPinState(LPC_GPIO_PORT, 3, 0, (bool) false); //Estado  
inicial del pin = 0
```

```
//HABILITAR GPIO2 (P6_5 <--- GPIO3[4]) COMO PIN DE SALIDA  
PARA EL TOGGLE
```

```
Chip_SCU_PinMux(3, 4, SCU_MODE_INACT |  
SCU_MODE_INBUFF_EN, SCU_MODE_FUNC0); /* EDU-CIAA GPIO2  
=> GPIO3[4], SALIDA PARA TEST */
```

```
Chip_GPIO_SetPinDIROutput(LPC_GPIO_PORT, 3, 4);
```

```
Chip_GPIO_SetPinState(LPC_GPIO_PORT, 3, 4, (bool) false); //Estado  
inicial del pin = 0
```

```
gpioConfig( LED2, GPIO_OUTPUT );
```

```
gpioConfig( GPIO2, GPIO_OUTPUT ); //SD
```

```
gpioConfig( GPIO8, GPIO_OUTPUT ); //Medicion de tiempos
```

```
// Configuracion de pines de salida LCD
```

```
gpioConfig( LCD1, GPIO_OUTPUT );
```

```
gpioConfig( LCD2, GPIO_OUTPUT );
```

```
gpioConfig( LCD3, GPIO_OUTPUT );
```

```
gpioConfig( LCD4, GPIO_OUTPUT );
```

```
gpioConfig( LCDRS, GPIO_OUTPUT );
```

```
gpioConfig( LCDEN, GPIO_OUTPUT );
```

```
//Configuracion de pulsadores
```

```
gpioConfig( T_FIL3, GPIO_INPUT_PULLUP ); //PULSADOR 1
```

```
gpioConfig( T_COL0, GPIO_INPUT_PULLUP ); //PULSADOR 2
```

```
gpioConfig( T_FIL2, GPIO_INPUT_PULLUP ); //PULSADOR 3
```

```
// Setup ADC0: 10-bit, ADC_RATE (frec. de muestreo del ADC)
```

```
Chip_ADC_Init(LPC_ADC0, &adc_setup);
```

```
Chip_ADC_SetSampleRate(LPC_ADC0, &adc_setup, ADC_RATE);
```

```
//Disable all channels
```

```
/* ADC0-CH1 -----> SEÑAL + RUIDO */
```

```
Chip_ADC_EnableChannel(LPC_ADC0, ADC_CH1, DISABLE);
```

```
Chip_ADC_Int_SetChannelCmd(LPC_ADC0, ADC_CH1, DISABLE);
```

```
/* ADC0-CH2 -----> RUIDO DE REFERENCIA */
```

```
Chip_ADC_EnableChannel(LPC_ADC0, ADC_CH2, DISABLE);
```

```
Chip_ADC_Int_SetChannelCmd(LPC_ADC0, ADC_CH2, DISABLE);
```

```
// Disable ADC0 burst mode
```

```
Chip_ADC_SetBurstCmd(LPC_ADC0, DISABLE);
```

```
//Init DAC
```

```
dacConfig(DAC_ENABLE);
```

```
//Enable RTC count
```

```
//Chip_RTC_Init(LPC_RTC);
```

```
//Chip_RTC_DeInit(LPC_RTC);
```

```
Chip_RTC_Enable(LPC_RTC, ENABLE);
```

```
/* SPI configuration */
```

```
spiConfig( SPI0 );
```

```
}
```

```
void initTimer(){
```

```
    //1ms timer
```

```
    Chip_TIMER_Init(LPC_TIMER0);
```

```
    Chip_RGU_TriggerReset(RGU_TIMER0_RST);
```

```
    while (Chip_RGU_InReset(RGU_TIMER0_RST)) {}
```

```
    uint32_t imstimerFreq =  
    Chip_Clock_GetRate(CLK_MX_TIMER0);
```

```
    //TCR: Counter reset
```

```
    Chip_TIMER_Reset(LPC_TIMER0);
```

```
    //Hace un reset para eliminar configuraciones anteriores
```



```
//MCR config
Chip_TIMER_PrescaleSet(LPC_TIMER0, 0); //PR = 0 ---->
Cuenta en cada ciclo de clock

Chip_TIMER_ResetOnMatchEnable(LPC_TIMER0, MR);
// Reseteo en match para el MR[n] elegido

Chip_TIMER_SetMatch(LPC_TIMER0, MR, 25500);
// Seteo del timer match para 8000cuentas/seg

//fm=8kHz--> 204M / 8000 = 25500

//fm=16kHz--> 204M / 16000 = 12750

//fm=10kHz-->204M / 10000 = 20400

//fm=~22kHz--> 204M / 22000 = 9274

Chip_TIMER_MatchEnableInt(LPC_TIMER0, MR);
// Habilito la interrupcion match que se dispara
cuando la cuenta (TC)

//alcanza el valor del match counter

Chip_TIMER_Enable(LPC_TIMER0);
//Habilita el timer (comienza la cuenta)

NVIC_ClearPendingIRQ(TIMERO_IRQn); //Borra
interrupciones pendientes del timer

//NVIC_EnableIRQ(TIMERO_IRQn); //Activa la
interrupción del timer

}

void TIMERO_IRQHandler (void)
{

    if (Chip_TIMER_MatchPending(LPC_TIMER0,
MR)) {

        Chip_TIMER_ClearMatch(LPC_TIMER0, MR);

ch1

// Inicio de conversión para el ADC0

Chip_ADC_EnableChannel(LPC_ADC0, ADC_CH1,
ENABLE);

Chip_ADC_SetStartMode(LPC_ADC0, ADC_START_NOW,
ADC_TRIGGERMODE_RISING);

// Esperar el final de conversión:
pooling

while(
(Chip_ADC_ReadStatus(LPC_ADC0 , ADC_CH1 ,
ADC_DR_DONE_STAT) != SET)

);

// Leer el valor convertido

Chip_ADC_ReadValue(LPC_ADC0,ADC_CH1, &ch1_signal);

//gpioWrite (GPIO8,ON);//Medicion
de tiempos de lectura del ADC - 13.3us dif CH1vsCH2

Chip_ADC_EnableChannel(LPC_ADC0, ADC_CH1,
DISABLE);

// Inicio de conversión para el ADC0
ch2

Chip_ADC_EnableChannel(LPC_ADC0, ADC_CH2,
ENABLE);

Chip_ADC_SetStartMode(LPC_ADC0, ADC_START_NOW,
ADC_TRIGGERMODE_RISING);

// Esperar el final de conversión:
pooling

while(
(Chip_ADC_ReadStatus(LPC_ADC0 , ADC_CH2 ,
ADC_DR_DONE_STAT) != SET)

);

// Leer el valor convertido

Chip_ADC_ReadValue(LPC_ADC0 ,
ADC_CH2 , &ch2_ref);

//gpioWrite (GPIO8,OFF); //Medicion
de tiempos de lectura del ADC - 13.3us dif CH1vsCH2

Chip_ADC_EnableChannel(LPC_ADC0 , ADC_CH2,
DISABLE);

//Transfiere contenido de los buffer
ADC (16 bits) a los buffer IN

//Invierto las entradas luego de que
saque el conector de audio!! 18/10
```



```
ch1_bufferIN[adcCntr]=ch1_signal;//ch2_ref;

ch2_bufferIN[adcCntr]=ch2_ref;//ch1_signal;

//dac[adcCntr]=ch1_bufferIN[adcCntr]; //Habilita
salida directa del ADC al DAC---Mantener comentado para funcionamiento
normal---

dac[adcCntr]=error_aux[adcCntr];

dac[adcCntr]=(((dac[adcCntr]^0x8000)>>6)&0x03ff); //Con la
xor con 0x8000 convierto q15.1 a uint16,

//luego desplazo 6 bits a la derecha para
que la variable sea de 10 bits (por el DAC) y enmascaro con AND

//y 0x03ff para que no haya valores mas
grandes que 1024 (2^10)

//Escritura en DAC
Chip_DAC_UpdateValue( LPC_DAC,
dac[adcCntr] );

//Incrementa posicion en buffers de
entrada y salida
adcCntr++;

//Transfiere buffers de entrada a los de
salida cuando se llega a 500 muestras (cada 62.5ms---1/8000x500)

if (adcCntr == Cant_Muestras ){
    adcCntr=0;
    for (i=0 ; i <
Cant_Muestras ; i++){

        ch1_signal_bufferOUT[i]=((ch1_bufferIN[i]<<6)^0x8000);

        ch2_ref_bufferOUT[i]=((ch2_bufferIN[i]<<6)^0x8000);

    }

    filtrar = 1;

}

//Cuando se presiona P2 se cierra el
archivo

if( !gpioRead(T_COL0) ){
    cerrar = 1;
}

}

}

// Se monta el sistema de archivos y Tarjeta SD.

static void mountSDCard( void ){

    // Give a work area to the default drive

    if( f_mount( &fs, "", 1 ) != FR_OK ){

        ERROR_SD=1;

        // If this fails, it means that the function could not register a file
        // system object. Check whether the SD card is correctly connected.

    }

    else

    {

        ERROR_SD=0;

    }

}

//Escribe texto en la segunda fila del display LCD

void mostrarTexto( char texto [])

{

    for (p=1 ; p<17 ; p++){

        lcd_gotoxy(p,2);

        lcd_putc(texto[p-1]);

    }

}

void mostrarHora( void)

{

    rtcRead (&RTC_read);

    //Lee las unidades de horas, min y seg del RTC

    hora_uni=(RTC_read.hour-(RTC_read.hour/10)*10)+48;

    min_uni=(RTC_read.min-(RTC_read.min/10)*10)+48;

    seg_uni=(RTC_read.sec-(RTC_read.sec/10)*10)+48;
```



```
//Lee las decenas de horas, min y seg del RTC
hora_dec=(RTC_read.hour/10)+48;
min_dec=(RTC_read.min/10)+48;
seg_dec=(RTC_read.sec/10)+48;

//Escribe horas, min y seg en el display
lcd_gotoxy(1, 1);
lcd_putc(hora_dec);

lcd_gotoxy(2, 1);
lcd_putc(hora_uni);

lcd_gotoxy(4, 1);
lcd_putc(min_dec);

lcd_gotoxy(5, 1);
lcd_putc(min_uni);

lcd_gotoxy(7, 1);
lcd_putc(seg_dec);

lcd_gotoxy(8, 1);
lcd_putc(seg_uni);
}

//Implementacion de tareas

// Implementacion de tarea clock_conf_Task
void clock_conf_Task (void *pvParameters)
{
    //Configuracion del reloj

    //1. Mostrar hora actual y preguntar por 1=hora OK
    //o 2=configurar. Queda en loop hasta que se tome accion alguna.....

    //2. Si hora esta OK entonces abrir archivo RAW
    //nuevo con hora y minutos actuales y suspender ESTA tarea!!! Iniciar
    //grabacion.

    //3. Si hora no esta OK permitir configurar hora y
    //luego volver a 1.

    lcd_init();

    char texto[16];
    char horaOK;

    //Configuracion del reloj
    RTC.year = 2017;
    RTC.month = 10;
    RTC.mday = 24;
    RTC.wday = 2;

    //rtcConfig(&RTC);

    //Escribo en display separadores de horas, minutos
    //y segundos
    lcd_gotoxy(3, 1);
    lcd_putc(':');
    lcd_gotoxy(6, 1);
    lcd_putc(':');

    while( TRUE )
    {
        mostrarTexto("P1=GRABAR");

        mostrarHora();

        //Inicia Grabacion SD
        if (!gpioRead(T_FIL3))
        {
            salir=0;

            while(!gpioRead(T_FIL3)); //No avanza hasta que se suelta la
            tecla
        }
    }
}
```



AUTOR:
JOSE MANUEL MOTTA

AÑO: 2017 / REV. 7

62



```
hora_uni=(RTC.hour-(RTC.hour/10)*10)+48;

hora_dec=(RTC.hour/10)+48;

lcd_gotoxy(1, 1);

lcd_putc(hora_dec);

lcd_gotoxy(2, 1);

lcd_putc(hora_uni);

while(!gpioRead(T_FIL3)); //No sale hasta que se suelta la tecla

}

//Incrementa minutos

if(!gpioRead(T_COL0))

{

    RTC.min++; //Incrementa 1 minuto

    if(RTC.min>=60)

    {

        RTC.min=0;

    }

    min_uni=(RTC.min-(RTC.min/10)*10)+48;

    min_dec=(RTC.min/10)+48;

    lcd_gotoxy(4, 1);

    lcd_putc(min_dec);

}

while(!gpioRead(T_FIL2)); //No sale hasta que se suelta la tecla

}

//Guarda hora y minutos

if(!gpioRead(T_FIL2))

{

    while(!gpioRead(T_FIL2)); //No sale hasta que se suelta la tecla

    lcd_gotoxy(16, 1);

    lcd_putc('*');

    if(!rtcConfig (&RTC)) //Inicializa y escribe la hora en el RTC

    {

        rtcWrite (&RTC);

    }

    lcd_gotoxy(16, 1);

    lcd_putc('

');

    horaOK=1;

}

}

}

void FiltroLMSTask( void *pvParameters)

{

    // portTickType xLastWakeTime;

    // *****
```



```
// PASO (1)
// Inicializo el filtro

u = 819; //Valor original u = 819
// *****

// PASO (1) -> Inicializacion del filtro

arm_lms_init_q15(&EstructuraDelfiltro, Cant_Coef_filtro,
coeficientes_del_filtro, State_Buffer, u, Cant_Muestras, 2);
// *****

// Para implementar un delay relativo
// xLastWakeTime = xTaskGetTickCount();

while (TRUE)
{

    if(filtrar){

        filtrar = 0; //reseteo flag de filtrado

        ready

        //gpioWrite (GPIO8, ON); //Medicion
        de tiempo de tarea de filtrado. Flanco subida.

        arm_lms_q15(&EstructuraDelfiltro,
ch1_signal_bufferOUT, ch2_ref_bufferOUT, Output, error_aux,
Cant_Muestras);

        //ch1_signal_bufferOUT = señal +
        ruido ch2_ref_bufferOUT = ruido de referencia

        //The error signal (error_aux) equals
        the difference between the reference signal d[n] (ch2_ref_bufferOUT) and the
        filter output (Output):

        // e[n] = d[n] - y[n]

        grabar = 1; //flag de grabacion SD

        //gpioWrite (GPIO8, OFF); //Medicion
        de tiempo de tarea de filtrado. Flanco bajada.

    }

    //vTaskDelay (10 / portTICK_RATE_MS);

}

}

// Implementacion de la tarea diskTask

void diskTask( void *pvParameters )
{
    // ----- CONFIGURACIONES -----

    // ----- REPETIR POR SIEMPRE -----

    while(TRUE)
    {

        disk_timerproc(); // Disk timer process

        vTaskDelay( 10 / portTICK_RATE_MS ); // Envia
        la tarea al estado bloqueado durante 10ms

    }

}

// Implementacion de la tarea SD_Write_Task

void SD_Write_Task( void *pvParameters )
{
    while(TRUE)
    {

        // Cuando se finaliza el filtrado se escribe en la SD

        if (grabar ){

            grabar = 0;

            //arm_q15_to_q7
            ch1_signal_bufferOUT , error_aux_sd , Cant_Muestras);

            arm_q15_to_q7 ( error_aux ,
            error_aux_sd , Cant_Muestras);

            f_write( &fp, error_aux_sd ,
            Cant_Muestras , &nbytes );

        }

    }

    if (cerrar)
```



```
{
    f_close(&fp); //Cierre del archivo en la
SD
    cerrar=0; //reseteo del flag "cerrar"
    finalizar =1;
}

//Deshabilita int timer

(FiltroLMSTaskhandle);

vTaskSuspend

vTaskSuspend

(SD_Write_TaskHandle);

vTaskSuspend (diskTaskHandle);

// Envia la tarea al estado bloqueado durante 50ms
vTaskDelay( 50 / portTICK_RATE_MS );
}

}

// Implementacion de funcion de la tarea recLEDTask

void recLEDTask( void *pvParameters )
{
    char textofin [16] = "FINALIZADO->P1=R";
    char texto_gr[16] = "GRABANDO->P2=FIN";

    // ----- REPETIR POR SIEMPRE -----
    while(TRUE)
    {
        if (!finalizar)
        {
            gpioWrite (LED1 , ON);
            mostrarTexto
(texto_gr);/"GRABANDO->P2=FIN"
            mostrarHora();
        }

        else{
            //Suspende todas las tareas de
discTask, filtrado y grabacion

            finalizar =0; // resetea el flag finalizar

            dacConfig (DAC_DISABLE);

            NVIC_DisableIRQ(TIMER0_IRQn);

            vTaskSuspend

            vTaskSuspend

            vTaskSuspend (diskTaskHandle);

            gpioWrite (LED2 , ON);
            gpioWrite (LED1 , OFF);
            gpioWrite (LED3 , OFF);

            mostrarTexto(textofin);

            /"FINALIZADO->P1=R"

            while (gpioRead(T_FIL3)); //Captura
en este loop hasta que se presione P1

            char texto[16] = "P1=GRABAR

            P2=cfg";

            mostrarTexto(texto);

            vTaskResume
(clock_conf_TaskHandle); //Reanima la tarea clock_conf_Task de max
prioridad, reiniciando el programa

        }

        // Envia la tarea al estado bloqueado durante 10ms
vTaskDelay( 10 / portTICK_RATE_MS );
    }

}

/* Funcion principal (main) */
int main(void)
{

```



```
0, tskIDLE_PRIORITY+1, &diskTaskHandle );

adcCntr=0;

indice = 0;

initHardware();

SystemCoreClockUpdate(); //Guarda el valor del SystemCoreClock

// LED para dar senial de vida
gpioWrite( LED3, ON );

// Crear tarea clock_conf_Task

//Esta tarea (clock_conf_Task) es la única a ejecutar inicialmente.
Se inicia con la mas alta prioridad.

//Pregunta si la hora esta correcta y si no lo está, debe permitir su
configuración.

//Una vez configurada la hora se suspende la tarea, habilitando la
ejecución de las otras tareas.

xTaskCreate(
    clock_conf_Task,          // Funcion de la tarea a
    ejecutar
    (const char
    *)"clock_conf_Task",      // Nombre de la tarea como String amigable para el
    usuario
    configMINIMAL_STACK_SIZE*2, // Tamano del stack de la tarea

    0,                        // Parametros de tarea

    tskIDLE_PRIORITY+2,      // Prioridad de la tarea

    &clock_conf_TaskHandle    // Puntero a la tarea creada en el sistema
);

// Crear tarea SD_Write_Task

xTaskCreate(
    SD_Write_Task, (const char *)"SD_Write_Task",
    configMINIMAL_STACK_SIZE*2,

    0,

    tskIDLE_PRIORITY+1, &SD_Write_TaskHandle );

// Crear tarea diskTask

xTaskCreate( diskTask, (const char *)"diskTask",
    configMINIMAL_STACK_SIZE,
```