CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS
DEL INSTITUTO POLITÉCNICO NACIONAL

UNIDAD ZACATENCO
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA
SECCIÓN DE BIOELECTRÓNICA

Sistema de monitoreo electroencefalográfico y visual para estudiar el comportamiento de pequeños roedores

Tesis que presenta

José Ernesto Pliego Sánchez

para obtener el Grado de

Maestro en Ciencias

en la Especialidad de

Ingeniería Eléctrica

Director de la Tesis: Dr. David Elías Viñas

México, D.F. Diciembre 2015

Agradecimientos

A mis padres por el aliento brindado durante mis estudios y por ser un apoyo en todo momento.

A mis hermanas y mi tía Marti por su compañía y los momentos de convivencia.

A Karina por su cariño, apoyo intenso y compañía en este viaje de aprendizaje, por los momentos alegres, las risas, los debates y los momentos difíciles.

A mi asesor el Doctor David Elias Viñas por sus consejos, su buen humor y ayuda en cada etapa de la maestría.

A mis compañeros de laboratorio Berno y Esteban por su apoyo desinteresado y las charlas en las que cuando abríamos los ojos aún estaban cerrados, junto con Isis, Emanuel, Jenny, Edgar, Nestor y todos los que compartido esos momentos.

A mis compañeros de maestría Roy, Deiniel, José Perez por su amistad y momentos agradables, de sobremesa y en la cancha, a Fersaurio y Karinosauria, Efren, Piter,Tony,etc .

Al Conacyt por el apoyo económico brindado durante mis estudios de maestría.

A todos aquellos que por alguna razón no he mencionado y que han formado parte de este proceso.

Sistema de monitoreo electroencefalográfico y visual para estudiar el comportamiento de pequeños roedores

Contenido

1 Resumen

El registro de potenciales de acción es un paradigma que se ha visto beneficiado gracias al desarrollo de arreglos de microelectrodos implantables, mediante los cuales es posible monitorear varios sustratos neuronales simultáneamente, tarea que generalmente se realiza mediante cables que llevan la información del arreglo de microelectros al dispositivo de adquisición y procesamiento de datos.

Existen algunos experimentos en los que resulta importante mantener las características ambientales y el estado de los animales de laboratorio lo más natural posible, para conseguir mediciones exactas. Por esta razón se realizó el diseño de un sistema compacto e inalámbrico que permite adquirir señales extracelulares para monitorear la actividad cerebral en animales despiertos y en libre movimiento, en el cual los animales desarrollen sus actividades ordinarias de manera lo más natural posible.

Se presenta el diseñó un sistema compacto de adquisición de potenciales extracelulares, que consiste de tres partes fundamentales: preamplificacíon, filtrado y procesamiento digital. Las dos primeras etapas del sistema limitan la banda de frecuencias útil (0.8-7.4Hz) mientras que la última etapa digitaliza y almacena la información en una memoria digital.
Se incluye en el sistema, una señal luminosa piulsante mediante un led que permite, sincronizar, la señal de EEG y la señal de video.

Mediante este sistema, se digitalizan señales eléctricas sin necesidad de utilizar conexiones alámbricas, además de poder sincronizar esta información con datos de video adquiridos simultáneamente, mediante la codificación de la señal luminosa del sistema y el procesamiento digital de imágenes del video.

2 Abstract

Action potentials have a central role in neurophysiology studies .In recent years, there has been an increasing interest in record these, thanks to development of microelectronic device. Commonly registration of action potentials is transmitted for scanning and processing by wires.

Despite its safety and efficacy, the paradigm suffers drawbacks when brain activity in freely moving is studied, by the use of wires. For this reason it was decided design and development of a compact and wireless monitoring system for action potential. The aim of this thesis is generate a tool to understand how the brain activity *and patterns behavior in animals are related.*

The design of the system is based on three modules: instrumentation amplifiers filter amplifiers with gain, and a SD data acquisition circuit board to record from 1 up to 4 channels. The printed circuit board contains also a flashing LED light for video tracking and synchronization.

The wireless monitoring system record the electrical activity and occurrence flashing LED light in SD memory, allowing the synchronization of the data with video recording, to identify the occurrence flashing LED light in video recording, the digital image processing is used.

3 Introducción

Nuestro cuerpo, su estructura y su funcionamiento han sido siempre objeto de asombro y estudio. Es pues para el ser humano prioritario entender aquellos fenómenos que afectan su bienestar, desde las enfermedades hasta los problemas sociales. En los estudios actuales de neurociencias se ha logrado profundizar tanto en los aspectos somáticos como sociales teniendo como principio rector, la idea de que todo aspecto sensible y todo comportamiento tienen asociado un sustrato neuronal. Partiendo de la premisa anterior, resulta lógico pensar que el estudio neuronal es fundamental para entender el funcionamiento del cerebro y que partes son las responsables de las funciones básicas y de la conducta, de tal manera que se pueda estudiar origen de algunas enfermedades neurológicas así como también del comportamiento general.

El presente proyecto pretende ofrecer una herramienta útil para la investigación en el estudio de diferentes situaciones de comportamiento en modelo animal (en pequeños roedores), mediante la adquisición del registro de potencial de algún sustrato neuronal específico, con el fin de observar los cambios eléctricos originados en dicho sustrato debido a estímulos asociados con el comportamiento social de las ratas. De la misma forma se pretende obtener videos (sincronizados con los registros eléctricos), para poder establecer la relación existente entre la actividad eléctrica y los patrones de conducta en animales despiertos y en movimiento libre.

El sistema necesita dos tipos de señal: La adquisición del potencial del sustrato neuronal y la adquisición de video, la primera utiliza un arreglo de amplificadores operacionales y componentes pasivos mediante los cuales se obtiene una señal fiable y además que permitan una fácil reproducción del sistema, dentro de este sistema es aplicada una señal luminosa intermitente mediante un led, cuya ocurrencia es registrada en la señal extracelular y es utilizada como señal de sincronía con el video, una vez obtenida la señal completa se guarda para su análisis posterior . La adquisición de video, se realiza mediante una cámara (cualquier cámara con una resolución mayor a 3mpixeles). La sincronía de estas dos señales (video y señal extracelular con pulsos luminosos) son posteriormente procesadas para su

sincronización mediante una aplicación que utiliza análisis de imágenes para identificar cuando enciende el LED.

3.1 Planteamiento del Problema

Actualmente se cuenta con tecnología para obtención de EEG y la mayoría de electroencefalógrafos son relativamente sencillos de conseguir e incluso de fabricar, sin embargo, en su gran mayoría están diseñados para los seres humanos y los escasos diseños para animales son de difícil acceso. La situación anterior aunada a la falta de software especializado y bloques que sincronicen varias señales de interés, han originado un retardo en las investigaciones fisiológicas que pudiera acelerarse con mejores proyectos tecnológicos.

3.2 Trabajos actuales

El estudio tanto de las señales del cerebro como del comportamiento se han popularizado y los métodos de adquisición y sobre todo de análisis se han diversificado.

Se han creado modelos neuronales biológicamente plausibles relacionados con la atención selectiva visual, simulando con éxito algunos experimentos realizados previamente [1].Estudios sobre la complejidad del EEG basados en la entropía de la muestra y el índice de entropía compuesta, en el dominio del tiempo, han podido determinar el estado de reposo o atención visual en diferentes zonas de la corteza cerebral, con lo cual se pretende diseñar “alarmas” de seguridad vial[2] Los estudios sobre ritmos de EEG has sido siempre de utilidad para definir los estados de vigilia, sin embargo es necesario contar con un método de análisis eficiente, que permita discriminar entre las señales registradas y almacenadas [3].Sin embargo también se pueden utilizar otras herramientas como la espectroscopia funcional por infrarrojo [4] o el estudio de imágenes PET [5].

Existen trabajos enfocados principalmente al control de prótesis, que utilizan la estimación de los estados cognitivos, los cuales representan la serie de “decisiones” que el cerebro ha tomado, respecto al movimiento a realizar y al entorno. La obtención de la señal para la m

estimación de los estados cognitivos se lleva a cabo en diferentes zonas del cerebro como son la corteza promotora y motora primaria, aunque estudios recientes afirman que el monitoreo de la corteza somato sensorial, permite el la estimación de estados cognitivos para tares complejas [6].

La mayoría de dispositivos comerciales diseñados para registro de potenciales extracelulares, tienen dimensiones que afectan las condiciones, en la que se encuentran los sujetos de estudio. La mayoría de estos dispositivos son diseñados para registros en humanos por lo que resulta complicado adaptarlos a mediciones con roedores.

Existe en el mercado un sistema con las características planteadas en este trabajo, de la empresa “Triangle Biosystems”, el cual permite monitoreo de 5 canales de señales extra celulares, y son enviadas mediante radiofrecuencia a un módulo de captura, para ser procesadas posteriormente [7]. La ventaja principal del sistema de “Triangle Biosystems” es que sus sistemas son de dimensiones muy pequeñas, además de que envían las señales de manera analógica, lo que les permite el envío de muchos canales, con procesadores de velocidades fáciles de conseguir. Sin embargo el diseño sistemas de radio frecuencia para la adquisición de la señal se hace más complejo, si se quiere tener señales limpias y de alta calidad.

3.3 Objetivos

3.3.1 Generales

Diseñar y construir un dispositivo compacto, para el registro de la actividad eléctrica extracelular de neuronas de animales en libre movimiento y vigilia y que mediante un video permita relacionar la actividad eléctrica con patrones conductuales de los animales en estudio.

3.3.2 Objetivos particulares

Diseñar y construir un circuito de no más de 15gr que permita la obtención del registro que contenga de la actividad eléctrica extracelular de neuronas.

Implementar un sistema que genere pulsos luminosos periódicos y que almacene la ocurrencia de estos pulsos junto con los registros de la actividad extracelular para la sincronización de señales de video con los registros extracelulares.

3.4 Descripción del trabajo

Este trabajo se encuentra organizado en seis capítulos principales: el primer capítulo llamado antecedentes, está dividido en tres partes.

En primer lugar se encuentra la descripción teórica de la comunicación entre neuronas así como la generación de potenciales de acción, una vez descrito este proceso, se plantean los métodos utilizados en la ciencia para la medición de dichos potenciales según el objetivo del estudio. Posteriormente se muestran algunos protocolos de estudios conductuales.

Dentro de este mismo capítulo se describen de manera sencilla los diseños electrónicos que permiten la medición de los potenciales neuronales, así como su digitalización , se describe también el protocolo de comunicación que utilizan la memorias SD en este desarrollo para almacenar las señales cerebrales.

En el capítulo de Desarrollo se describe como se diseñó el sistema. Primero se dan detalles de la adquisición de la señal y de los filtros utilizados, continúa el capítulo con el desarrollo de funciones para el protocolo de comunicación con la memoria SD, así como la descripción del protocolo de comunicación USB mediante el PIC18F2550 que permite la lectura de los datos, y su procesamiento dentro de la computadora .

Posteriormente en el capítulo Pruebas se muestra el funcionamiento del sistema, las señales adquiridas, así como el método utilizado para la obtención de dichos datos.

En el capítulo de resultados se muestran los datos obtenidos así como su análisis, funcionamiento del circuito, el muestreo de la señal, adquisición mediante puerto USB, análisis de oscilaciones de alta frecuencia y sincronización de señales.

La interpretación de los resultados así como sus implicaciones son planteadas en el capítulo de discusión, aquí se define el alcance del sistema, que tipo de señales y que tipo de estudios pueden utilizar el sistema, así como su versatilidad.

Por último se define la importancia de utilizar sistemas inalámbricos y como el diseño desarrollado, puede permitir estudios en espacio libre que emulen más fielmente el ambiente natural de los sujetos de prueba. Además se exponen las características que pueden mejorarse, y los trabajos futuros que se han planeado.

4 Antecedentes

4.1 La neurona y su potencial de acción

La célula como unidad anatómica y funcional en los sistemas vivos juega un papel importante en el estudio del comportamiento de estos. Las neuronas siendo una de las células que componen el cerebro constituyen un eslabón necesario para la comprensión fisiológica del encéfalo y por lo tanto del comportamiento.

En general las neuronas están formadas por un cuerpo celular, dendritas (mediante las cuales recibe información) y un axón (con el cual envía información).

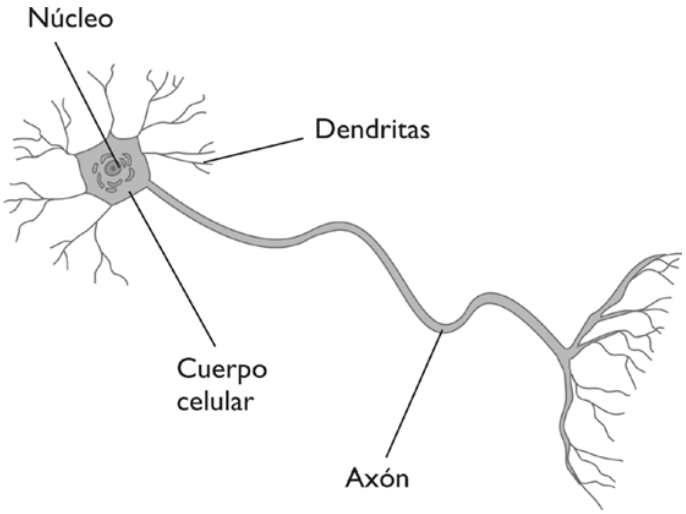


Figura 1 Partes de una neurona [8]

Todas las neuronas independientemente de que sean sensoriales o motoras, grandes o pequeñas, tienen una característica en común ya que su actividad es de dos tipos: eléctrica y química.

La trasferencia de información entre neuronas se lleva a cabo en la sinapsis, que es el espacio entre la célula que envía mediante la terminación pre sináptica (axón) y la neurona que recibe mediante la terminación pos sináptica (dendrita).El intercambio de información puede ser de tipo químico o de tipo eléctrico .Las señales químicas recibidas, por las

dendritas procedentes de los axones que las contactan, se transforman en señales eléctricas y se incorporan (adicionándose o sustrayéndose) al resto de señales procedentes de las otras sinapsis, decidiéndose si la señal se propaga hacia la siguiente neurona o no. Por lo tanto, los potenciales eléctricos viajan por el axón hacia la sinapsis, pasando a las dendritas de la siguiente neurona.

La actividad química de las neuronas es llevada a cabo principalmente por los neurotransmisores y estos juegan un papel importante en la comunicación neuronal del sistema nervioso que puede reflejarse en el comportamiento de los animales, mantienen las funciones cerebrales y los procesos vitales del organismo, pues un déficit o exceso de ellos produce producir alteraciones en el comportamiento los efectos de los neurotransmisores pueden ser excitatorios o inhibitorios. Además un mismo neurotransmisor puede tener diferentes tipos y grados de acción, dependiendo del tipo de receptor con el que interactúe [9].

Name	Location	Effect	Function
Acetylcholine (ACh)	Brain, spinal cord, peripheral nervous system, especially some organs of the parasympathetic nervous system	Excitatory in brain and autonomic nervous system; inhibitory elsewhere	Muscle movement, cognitive functioning
Glutamate	Brain, spinal cord	Excitatory	Memory
Gamma-amino butyric acid (GABA)	Brain, spinal cord	Main inhibitory neurotransmitter	Eating, aggression, sleeping
Dopamine (DA)	Brain	Inhibitory or excitatory	Muscle disorders, mental disorders, Parkinson's disease
Serotonin	Brain, spinal cord	Inhibitory	Sleeping, eating, mood, pain, depression
Endorphins	Brain, spinal cord	Primarily inhibitory, except in hippocampus	Pain suppression, pleasurable feelings, appetities, placebos

Figura 2 Principales Neurotransmisores [10]

Cuando una dendrita recibe uno de los mensajeros químicos liberados por uno de los axones al espacio que los separa, se crean en ella corrientes iónicas. Estas corrientes

pueden ser catiónica y dirigirse a la célula, y son llamadas excitatorias o bien ser anionicas y moverse hacia fuera de la célula, y entonces son llamadas inhibitorias.

Todas estas corrientes positivas y negativas se acumulan en las dendritas y se dispersan posteriormente por el cuerpo celular. Si estas corrientes no crean suficiente actividad al sumarse acaban muriendo y no ocurre nada más. Sin embargo, si estas corrientes al sumarse superan el potencial umbral de la membrana, entonces la neurona enviará un mensaje a las otras neuronas vecinas, a través de los axones en forma de pulsos eléctricos llamados potenciales de acción.

Cuando un potencial de acción se inicia en el cuerpo celular, los canales que se abren en primer lugar son los canales de Na+ activados por voltaje. Una corriente de iones sodio entra directamente en la célula y en cuestión de milisegundos se establece un nuevo equilibrio de potencial. En un instante, el voltaje de membrana cambia en aproximadamente 100 mV. Se transforma de un potencial negativo de reposo dentro de la membrana (aproximadamente -70mV) a uno positivo (aproximadamente +30mV). Este cambio de potencial hace que los canales de activados por voltaje K+ se abran, iniciando una corriente iones de K+ hacia el exterior de la célula, casi tan rápido como el flujo de iones de Na+, lo que hace que el potencial dentro de la célula vuelva nuevamente a su valor negativo original. Se necesitan muy pocos iones atravesando la membrana para producir este efecto y la concentración de Na+ y K+ dentro del citoplasma durante un potencial de acción no varía significativamente. De todas formas, a largo plazo el equilibrio iónico dentro de la célula se mantiene gracias al trabajo de las bombas iónicas, que se encargan de eliminar el exceso de sodio-potasio.

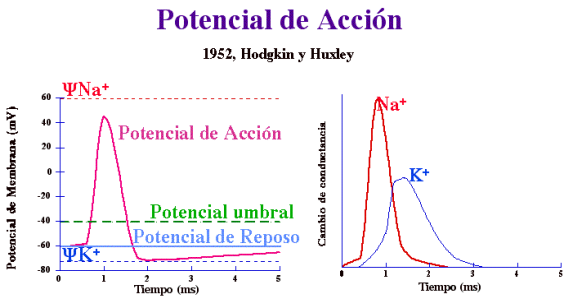


Figura 3 Potencial de acción [11]

4.2 Registro de potencial

Existen diferentes técnicas para la medición de los potenciales de acción de las células, estos se pueden dividir en: medición de respuestas masivas y medición de células aisladas.

Dentro de la medición masiva, se puede mencionar la medición del potencial de acción del nervio, potenciales lentos y descargas unitarias sumadas (como es el caso de Electromiograma). Mientras que las mediciones de células aisladas presentan técnicas como, registro intracelular y extracelular.

El registro extracelular consiste en introducir, en el tejido nervioso, un electrodo sumamente fino, hasta que su punta llegue lo bastante cerca de la célula que se desee medir su respuesta, aunque esta técnica mide de la actividad de una sola neurona, puede ocurrir que este registrando la actividad de varias, una forma de observar si esta actividad es la deseada, es comparando las descargas de punta que genera la células (una actividad normal en células sanas), si estas descargas tienen la misma altura el registro se está realizando sobre una sola célula.

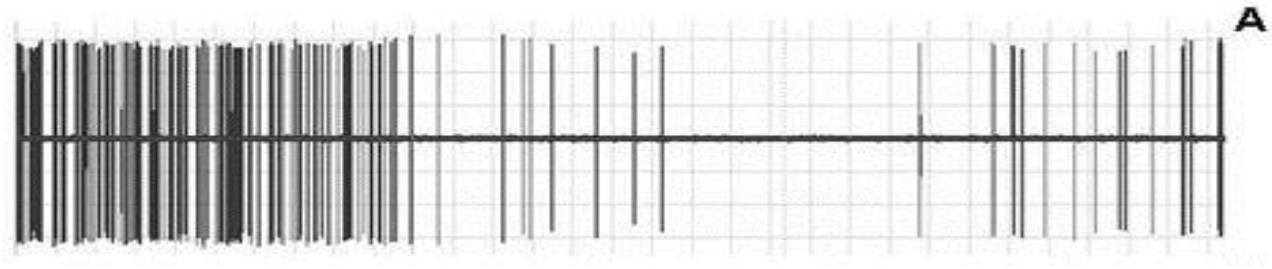


Figura 4 Registro de una sola neurona. [12]

4.3 Áreas cerebrales r estudios de comportamiento

El cerebro humano contiene alrededor de 12 000 000 000 células nerviosas. El número posible de interconexiones entre todas las células nerviosas de un solo cerebro es mayor que el número de partículas atómicas que constituyen el universo entero. Es por esto que determinar cómo funciona este sistema para producir la infinidad de modelos de conducta resulta una tarea difícil[13].

La organización neuronal ha permitido el desarrollo la corteza cerebral (neo corteza) que es el logro más reciente de la evolución del sistema nervioso de los vertebrados. Los peces y los anfibios no tienen corteza cerebral, mientras que las aves y los reptiles poseen indicios rudimentarios de ella, es decir se puede afirmar que existe una correlación entre la magnitud del desarrollo cortical de una especie y su nivel filogenético además del grado de complejidad y flexibilidad de su conducta, siendo el ser humano quien posee mayor complejidad.

Existen en la corteza cerebral diferentes zonas bien definidas cuya función está relacionada con aspectos sensoriales y motores: Área visual que está situada en el lóbulo occipital. En ella se aprecian zonas específicas para la visión de la mácula o central; para la periferia de la retina y para las mitades superior e inferior de la retina. Área auditiva se halla situada en los lóbulos temporales, por debajo de la cisura lateral o de Silvio. Parece ser que cada oído tiene representación bilateral en la corteza por lo que al extirpar un lóbulo temporal no se sufre mayor disminución de la audición. Área olfativa se sitúa en la circunvolución del hipocampo, próxima a la auditiva.

Áreas motrices, la principal área motora, 4 de Brodmann, se halla situada delante del surco central o cisura de rolando. Posee células gigantes de las que nacen las vías corticoespinal y corticobulbar con axones para los músculos estriados del organismo. En la parte más alta de esta área se localiza la zona para los movimientos de los miembros más distantes: pies, rodillas, cadera; y en las partes más bajas los músculos para la masticación, deglución, caza cabeza, cuello y las zonas más próximas de la corteza premotora. Además de esta área, existe otra situada por delante de ella, que se considera promotora y cuya lesión produce pérdida temporal de las destrezas adquiridas. Estás áreas envían los impulsos para la acción voluntaria, participando en la misma otros centros, ya que el sistema nervioso funciona en forma integral

Existe además una buena porción de neocorteza que no es ni motora ni sensitiva, esta región de corteza denominadas comúnmente como áreas de asociación, ha crecido enormemente en el desarrollo filogenético de los mamíferos, es por eso que la corteza asociativa, se ha vinculado entre otras cosas con la conducta.

En muchos trabajos se han distinguido tres tipos de fenómenos conductuales: habituación, atención y aprendizaje. Para el estudio de estos fenómenos se ha utilizado el análisis de los potenciales evocados por distintos estímulos en diferentes niveles del sistema nervioso central, desde los núcleos sensoriales de primer orden hasta la corteza cerebral.

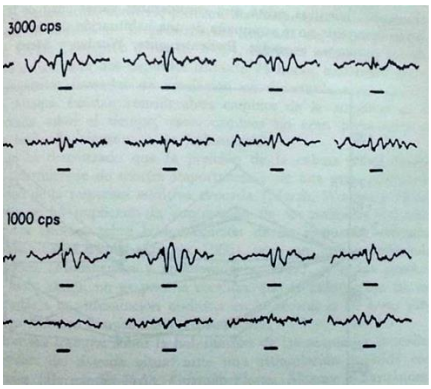


Figura 5 Respuesta auditiva en el núcleo coclear. Primeras y últimas cuatro muestra de una serie de veinte estímulos auditivos a 3000 y 1000 Hz. [14]

Durante el estudio de la atención se registran los potenciales de la variable de interés cuando el sujeto está en reposo y posteriormente se distrae con otro estimulo mayor para medir nuevamente y comparar ambas señales, lo cual nos da una idea del grado de atención. En la figura 6 se muestran las fotografías de un gato en reposo(superior e inferior) mientras se producía un ruido fuerte, así como su respuesta evocada y en la fotografía central el gato es distraído con un ratón, se puede observar como la respuesta evocada disminuye de amplitud debido al desvió de atención hacia el ratón.

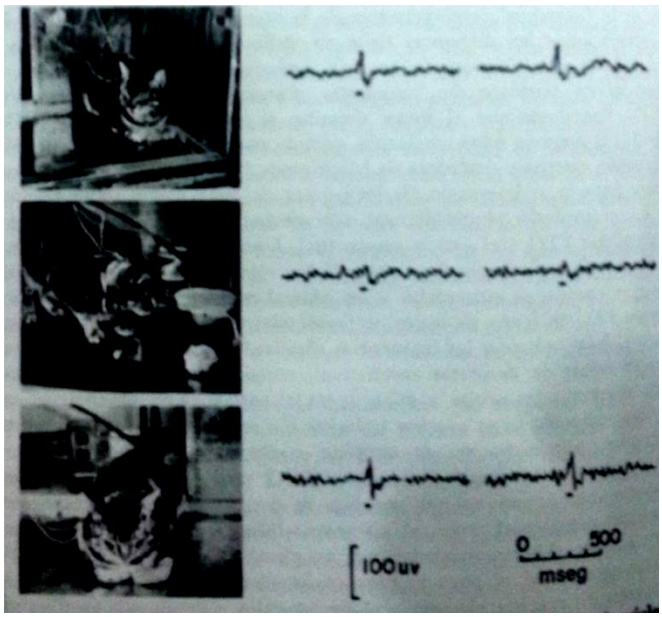


Figura 6 Respuesta evocada en estudio conductual de atención [15]

En el estudio de aprendizaje se han utilizado dos enfoques, el más aceptado es el estudio de potenciales evocados. En estos estudios se condiciona al sujeto mediante dos señales apareadas, posteriormente se suspende la señal condicionante y se observa el grado de condicionamiento (aprendizaje), del sujeto.

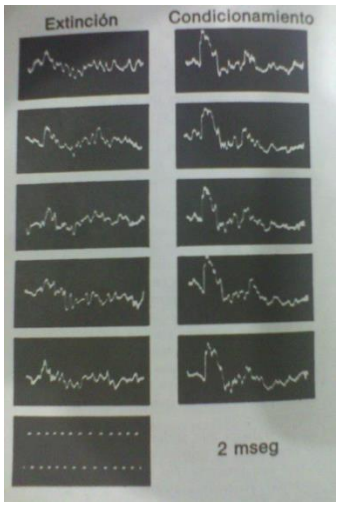


Figura 7 Respuesta evocada en estudio conductual de aprendizaje. [16]

En la figura 7 se observan los potenciales evocados en los núcleos cocleares, a la izquierda los potenciales obtenidos mediante una señal sonora extinguida (habituada), y a la derecha la señal tomada en el mismo lugar y con el mismo estimulo sonoro pero condicionado con una descarga eléctrica (en ninguno de los dos se presenta la descarga, pero en el registro de la derecha se observa el acondicionamiento).

4.4 Filtros

4.4.1 Filtro Sallen Key (VCVS) de segundo orden pasa-bajas (*Butterworth*)

Los filtros de fuente de voltaje controlada por voltaje se construyen, mediante un amplificador no inversor que hace las veces de la fuente controlada, según el número de elementos que almacenan energía podemos determinar el orden del filtro. Para filtros de segundo orden su topología es la que se muestra en la figura 8.

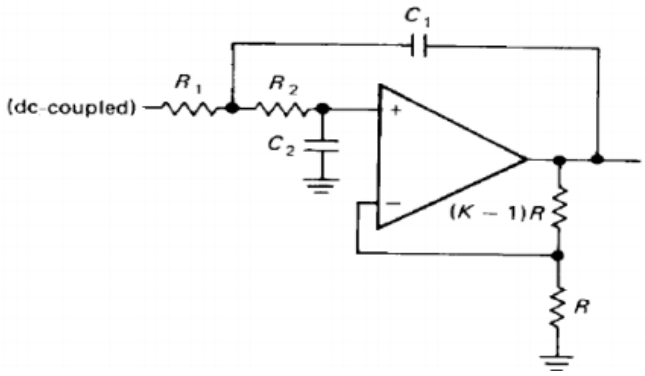


Figura 8 Tipología VCVS Filtro pasa bajas[17]

Existen diferentes métodos para calcular los componentes del filtro, y estos dependen de la aplicación. Si se desea hacer un control del régimen transitorio de la señal de salida conviene diseñar el circuito a partir de la función de transferencia, sin embargo existe otro método de diseño, basado en tablas, calculadas a partir de dicha función de transferencia, las cuales nos proporcionan la ganancia del circuito a partir del tipo de filtro y el número de polos (grado del filtro) que se desee. La tabla se muestra a continuación.

Tabla 1 Valores de ganancia para diseño de filtros VCVS

Poles	Butter-worth	Bessel		Chebyshev (0.5dB)		Chebyshev (2.0dB)	
	K	f_n	K	f_n	K	f_n	K
2	1.586	1.272	1.268	1.231	1.842	0.907	2.114
4	1.152	1.432	1.084	0.597	1.582	0.471	1.924
	2.235	1.606	1.759	1.031	2.660	0.964	2.782
6	1.068	1.607	1.040	0.396	1.537	0.316	1.891
	1.586	1.692	1.364	0.768	2.448	0.730	2.648
	2.483	1.908	2.023	1.011	2.846	0.983	2.904
8	1.038	1.781	1.024	0.297	1.522	0.238	1.879
	1.337	1.835	1.213	0.599	2.379	0.572	2.605
	1.889	1.956	1.593	0.861	2.711	0.842	2.821
	2.610	2.192	2.184	1.006	2.913	0.990	2.946

Para un filtro Butterworth de segundo orden, por simplicidad se define R1 = R2 = R, and C1 = C2 = C. La relación entre la resistencia, el capacitor y la frecuencia de corte está dada por la ecuación:

$$f_c = \frac{1}{2\pi RC}$$

Las resistencias R se calcula en base a la ganancia establecida en la tabla.

4.5 Amplificador de instrumentación

El amplificador de instrumentación es un amplificador diferencial tensión-tensión cuya ganancia puede establecerse de forma muy precisa y que ha sido optimizado para que opere de acuerdo a su propia especificación aún en un entorno hostil. Es un elemento esencial de los sistemas de medición, en los que se ensambla como un bloque funcional que ofrece características especiales e independientes de los restantes elementos con los que interacciona.

Se caracteriza por proporcionar rechazo de señales que se encuentren en las dos terminales del amplificador y su grado de rechazo se mide mediante el parámetro denominado razón de rechazo en modo común (CMRR por sus siglas en ingles).

Existen en el mercado diferentes amplificadores, que presentan CMRR alto (80-100), y utilizan para su configuración de ganancia dos terminales para la conexión de la resistencia R_G. La relación entre R_G y la ganancia está dada por:

$$G = \frac{49.4k\Omega}{R_G} + 1$$

Donde

G es ganancia del amplificador de instrumentación

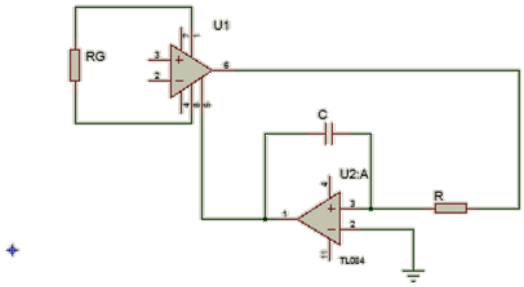
R_G es la resistencia de ganancia

4.6 Corrector de línea de base

El corrector de línea de base es un circuito que nos permite quitar las componentes de baja frecuencia de la señal especialmente de la de corriente continua, esto puede lograrse,

cambiando la referencia del amplificador de instrumentación por un voltaje igual pero negativo de las componentes que se quieren eliminar. Para definir una banda que se quiere rechazar, podemos utilizar un filtro pasa bajos de primer orden y utilizarlo como voltaje de referencia para eliminar dichas componentes.

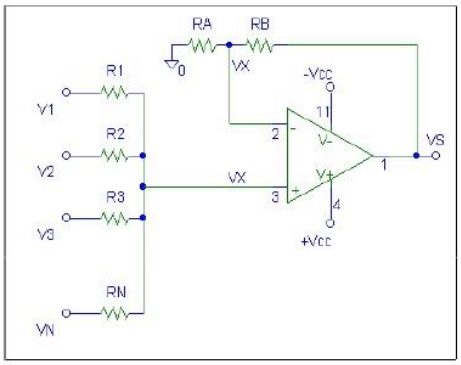
El circuito siguiente muestra el diseño de un corrector de línea de base y la ecuación que define la banda de rechazo.



$$f_c = \frac{1}{2\pi RC}$$

4.7 Amplificador sumador no inversor

El Circuito Sumador es un circuito muy útil, basado en la configuración estándar del amplificador operacional. Este circuito permite combinar múltiples entradas, es decir, permite añadir algebraicamente dos (o más) señales o voltajes para formar la suma de dichas señales. A continuación se muestra la configuración del circuito sumador no inversor.



$$\left.\begin{aligned}\frac{V_1 - V_X}{R_1} + \frac{V_2 - V_X}{R_2} + ... + \frac{V_N - V_X}{R_N} = 0 \\ \frac{V_X}{R_A} = \frac{V_S}{R_A + R_B}\end{aligned}\right\}$$

$$V_S = \left(1 + \frac{R_B}{R_A}\right) \cdot \frac{\sum_{i=1}^{n} \frac{V_i}{R_i}}{\sum_{i=1}^{n} \frac{1}{R_i}}$$

Si $R_1 = R_2 = R_3 = ... = R_N = R$

$$V_S = \frac{1}{n} \cdot \left(1 + \frac{R_B}{R_A}\right) \cdot \sum_{i=1}^{n} V_i$$

Figura 9 Sumador no inversor

4.8 Micro controlador PIC18F2550

Los microcontroladores son computadoras digitales integrados en un chip que cuentan con un microprocesador o unidad de procesamiento central (CPU), una memoria para almacenar el programa, una memoria para almacenar datos y puertos de entrada salida.

A diferencia de los microprocesadores de propósito general, como los que se usan en los computadores PC, los micro controladores son unidades autosuficientes y más económicas. El funcionamiento de los micros controladores está determinado por el programa almacenado en su memoria. Este puede escribirse en distintos leguajes de programación. Además, la mayoría de los micros controladores actuales pueden reprogramarse repetidas veces. Por las características mencionadas y su alta flexibilidad, los micro controladores son ampliamente utilizados como el cerebro de una gran variedad de sistemas embebidos que controlan máquinas, componentes de sistemas complejos,etc.

El PIC es un micro controlador fabricado exclusivamente por la compañía Microchip, que se divide en varias familias. Arquitectura de 8 bits tiene en orden ascendente el rendimiento y el tamaño del PIC10, PIC12, PIC16 y PIC18. Con la arquitectura 16-bit, hay los PIC24F y PIC24H micro controladores y controladores de señal dsPIC30 y dsPIC33. La mayoría de los

micros controladores recibe como programa el archivo binario generado por un lenguaje compilado.

La familia 18F2550 presenta algunas características especiales como 32KB de memoria flash para programación, RAM de 2KB,EEPROM: 256 B, Velocidad: 48MHz, 24 puertos de entrada/salida,10 canales de ADC de 10 bits,4 Timers, Interfaz: I²C, SPI, UART/USART, USB ,Voltaje de alimentación: 4,2V ~ 5,5V. En la figura 10 se muestra la configuración de pines del micro controlador Pic18f2550.

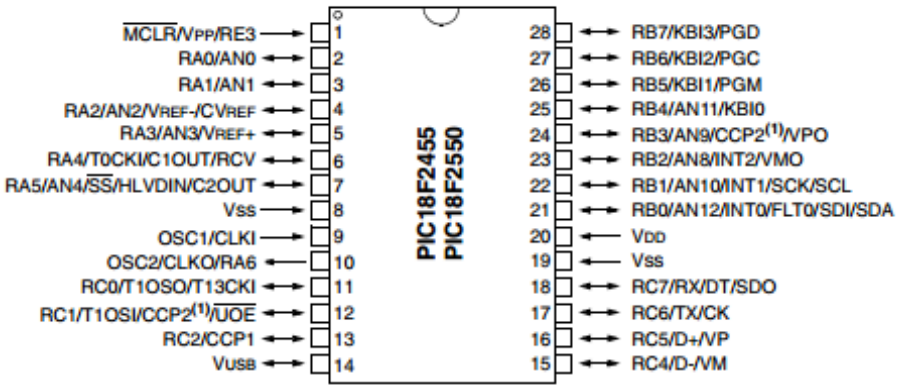


Figura 10 Configuración de pines del pic18f2550

La velocidad máxima de CPU de este micro controlador es de 48MHz, es decir puede realizar 12MIPS (mega instrucciones por segundo), para lograr esta velocidad, se necesita utilizar un cristal oscilador externo y activar dentro del micro controlador el PLL que permita generar una señal de 96MHz a partir del cristal externo, esto se logra mediante los registros CONFIG1L, CONFIG1H, OSCCON OSCTUNE.

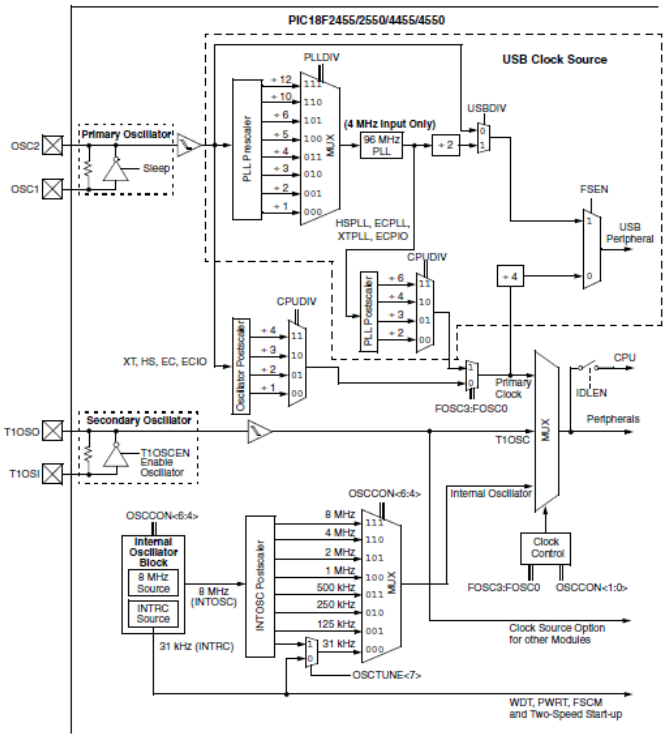


Figura11 Diagrama de bloques del oscilador del PIC18f2550 [18]

El módulo de comunicación SPI de PIC18f2550 permite la transmisión de datos de 8 bits y recepción simultánea, esta comunicación se logra mediante los pines SDI, SDO, SCK para un dispositivo maestro y adicional mente el pin SS para dispositivos esclavo. Para configurar las características del módulo de comunicación SPI se utilizan el registro SSPCON1, el registro SSPSTAT muestra el estado del dispositivo SPI mientras que los datos trasmitidos o recibidos se almacenan en el registro SSPBUF.

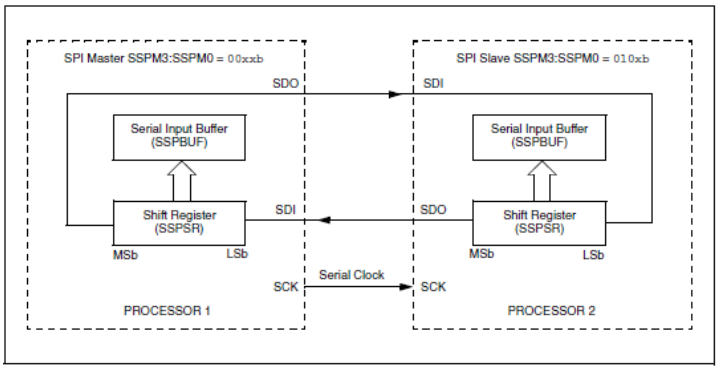


Figura 12 Conexiones de Módulo SPI[18][ii]

Para la conversión analógica digital, el micro controlador cuenta con un convertidor de 10-bits, el registro ADCON0 controla el funcionamiento del módulo, por su parte ADCON1 permite configurar las funciones analógicas de los pines y ADCON2 es el registro en el que se configura la velocidad y el tiempo de conversión. Para poder leer el dato digital se utilizan los registros ADRESH y ADRESL.

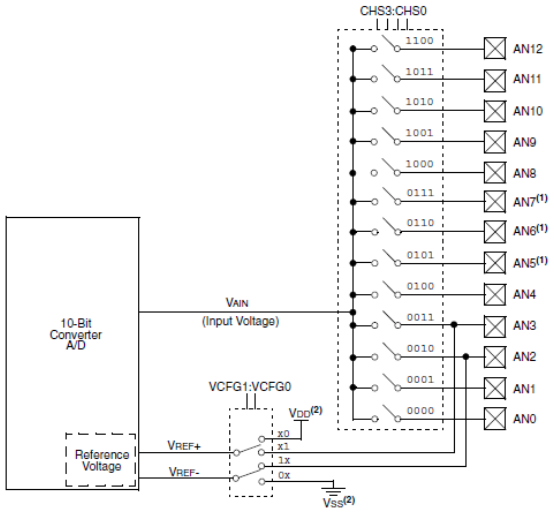


Figura 13 Diagrama de módulo ADC PIC18f2550[18]

Para medición de tiempo dentro del PIC se existe los módulos de timer, los cuales son contadores de 8bits o 16 bits que se incrementan mediante pulsos del reloj del timer, cuando el contador alcanza su valor máximo un bit dentro del byte llamado INTCON cambia su valor a 1 lógico. Si se configura la interrupción del timer, en el momento que se desborde el contador, se ejecuta la subrutina definida por el usuario previamente.

El registro T0CON configura la fuente de reloj así como su pre-escalamiento, mediante los registros INTCON e INTCON2 muestran el estado del contador, por otro lado los registros TMR0L y TMR0H guardan el conteo del timer.

FIGURE 11-1: TIMER0 BLOCK DIAGRAM (8-BIT MODE)

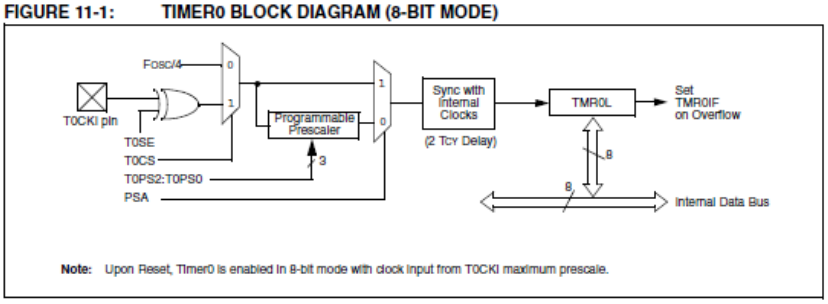


FIGURE 11-2: TIMER0 BLOCK DIAGRAM (16-BIT MODE)

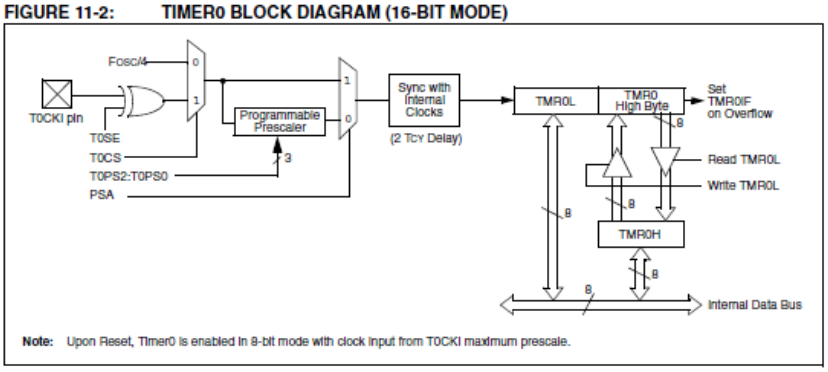


Figura 14 Diagrama del TIMER0

El PIC18f2550 cuenta con una interface serial para comunicación USB, la cual permite comunicación de alta y baja frecuencia, con un host . Para la configuración de este módulo se utiliza el registro UCFG el control se lleva a cabo mediante el registro UCON, mientras que el estado puede monitorearse en el registro USTAT. Es necesario en el protocolo de comunicación contar con algunos parámetros como son la dirección del dispositivo, el número de paquetes enviados y los endpoint utilizados, estos parámetros son configurados mediante los registros UADDR , UFMRH:UFMRL y UEPn respectivamente.

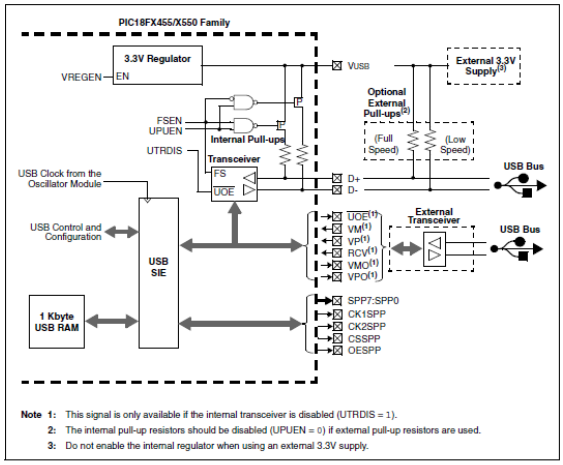


Figura 15 Diagrama del módulo USB [18]

4.9 Compilador CCS[19]

El lenguaje C estándar es independiente de cualquier plataforma. Sin embargo, para la programación de micro controladores es necesario disponer de determinados comandos que se refieran a partes específicas de su hardware, como el acceso a memoria, temporizadores, etc. Por este motivo, además de los comandos, funciones y datos del lenguaje ANSI C, el compilador CCS incluye bibliotecas que incorporan determinados comandos que no son estándar, sino específicos para la familia de micro controladores PIC. Éstos son básicamente de dos tipos: directivas del preprocesador y funciones pre compiladas.

Directivas del preprocesador.

El compilador CCS dispone de 7 tipos básicos de directivas:

Directivas derivadas del estándar de C, que permiten, entre otras funciones, un control básico del código y del flujo en el proceso de compilación:

#DEFINE,#ELIF,#ELSE,#ENDIF,#ERROR,#IF,#IFDEF,#IFNDEF,#INCLUDE,#LIST,#NOLIST ,#PRAGMA,#UNDEF.

Directivas asociadas a las bibliotecas pre-compiladas, que proporcionan al compilador información relacionada con estas bibliotecas:
#USE DELAY,#USE FAST_IO,#USE FIXED_IO,#USE I2C,#USE RS232,#USE STANDARD_IO
Directivas relacionadas con la especificación del dispositivo, por un lado, para definir los mapas de memoria y el juego de instrucciones, y por otro, incluir información necesaria para la programación del dispositivo en los ficheros de salida de la compilación:
#DEVICE,#ID,#FUSES,#TYPE
Directivas de cualificación de funciones, para identificar características especiales de una función:
#INLINE,#INT_DEFAULT,#INT_GLOBAL,#INT_xxxxx, #SEPARATE

Directivas de control del compilador, para definir opciones referidas a la compilación del código del programa:

#CASE,#OPT,#ORG,#PRIORITY

Directivas de control de la memoria del micro controlador, para gestionar y reservar el uso de determinadas zonas de memoria para variables:

#ASM,#BIT,#BYTE,#ENDASM,#LOCATE,#RESERVE,#ROM,#ZERO_RAM

Identificadores predefinidos. Todas las directivas citadas hasta ahora, son comandos destinados a ser interpretados por el compilador, no por el micro controlador. Dentro del término genérico de directiva se incluyen, además de estos comandos, unas variables que contienen información sobre el proceso de compilación. Estas variables son lo que se denominan identificadores predefinidos del compilador:

__DATE__

__DEVICE__

__PCB__

__PCH__

__PCM__

En un programa, las directivas se reconocen fácilmente porque comienzan por el símbolo #, mientras que los identificadores empiezan y acaban por doble subrayado (__).

Proporciona una gama de funciones que puede facilitar considerablemente la tarea de programación como son leer la entrada de un teclado o imprimir un determinado mensaje en una pantalla LCD conectada como salida. Existen funciones en C incluidas en el compilador PCW para manejar los diferentes recursos del micro controlador como son:

Funciones de I/O serie RS232

Funciones de I/O con el BUS I2C

Funciones de I/O DISCRETA

Funciones de RETARDOS

Funciones de CONTROL del PROCESADOR

Funciones de CONTADORES/TEMPORIZADORES

Funciones de I/O PSP PARALELA
Funciones de I/O SPI A DOS HILOS

Funciones para el LCD
Funciones del C ESTÁNDAR
Funciones de Manejo de Cadenas
Funciones de ENTRADA A/D
Funciones CCP
Funciones para la EEPROM interna
Funciones de MANIPULACIÓN DE BITS

Las funciones utilizadas para el uso del módulo SPI son:
SETUP_SPI(mode) Esta función inicializa el SPI
mode puede ser: o SPI_MASTER, SPI_SLAVE o SPI_L_TO_H, SPI_H_TO_L o SPI_CLK_DIV_4, SPI_CLK_DIV_16, o SPI_CLK_DIV_64, SPI_CLK_T2 o SPI_SS_DISABLED.

SPI_DATA_IS_IN() Devuelve 1 si se han recibido datos en el SPI.

SPI_READ() Devuelve un valor leído por el SPI.

SPI_WRITE(value) Escribe el valor por el SPI.

Para configurar y uso del Timer se utilizan:

SETUP_TIMER_0 (rtcc_state| ps_state) Inicializa el timer 0 . El rtcc_state determina qué es lo que activa el timer0 y puede ser RTCC_INTERNAL RTCC_EXT_L_TO_H RTCC_EXT_H_TO_L. El ps_state establece un pre-scaler para el timer 0. El prescaler alarga el ciclo del contador indicado y puede ser RTCC_DIV_2 RTCC_DIV_4 RTCC_DIV_8 RTCC_DIV_16 RTCC_DIV_32 RTCC_DIV_64 RTCC_DIV_128 RTCC_DIV_256

SET_TIMER0(value) Esta función activan el timer o temporizador al valor especificado. RTCC y Timer0 son el mismo. Timer1 es de 16 bits y los otros son de 8 bits. value depende del tiempo que se desee contar.
Si se desea utilizar el timer en modo interrupción las funciones siguientes son utilizadas:
DISABLE_INTERRUPTS(level) Desactiva la interrupción del nivel dado en level. El nivel GLOBAL prohíbe todas las interrupciones, aunque estén habilitadas o permitidas. Los niveles de interrupción son: o GLOBAL o INT_AD o INT_CCP2 o INT_COMP o INT_EXT o INT_EEPROM o INT_SSP o INT_ADOF o INT_RTCC o INT_TIMER1 o INT_PSP o INT_RC o INT_RB o INT_TIMER2 o INT_TBE o INT_I2C o INT_AD o INT_CP1 o INT_RDA o INT_BUTTON

ENABLE_INTERRUPTS(level) Esta función activa la interrupción del nivel dado en level. El nivel GLOBAL permite todas las interrupciones que estén habilitadas de forma individual.
Las funciones disponibles para manejo del convertidor analógico digital son:

SETUP_ADC_PORTS(value) Configura los pines del ADC para que sean analógicos, digitales o alguna combinación de ambos. Las combinaciones permitidas varían, dependiendo del chip.

SETUP_ADC(mode) Configura el tiempo de conversión del conversor A/D.Los modos son: o ADC_OFF o ADC_CLOCK_DIV_2 o ADC_CLOCK_DIV_8 o ADC_CLOCK_DIV_32 o ADC_CLOCK_INTERNAL

SET_ADC_CHANNEL(channel) Especifica el canal a utilizar por la función READ_ADC(). El número de canal empieza en 0. Es preciso esperar un corto espacio de tiempo después de cambiar el canal de adquisición, antes de que se puedan obtener lecturas de datos válidos.

READ_ADC() Lee el valor digital del conversor analógico digital. Deben hacerse llamadas a SETUP_ADC() y SET_ADC_CHANNEL() en algún momento antes de la llamada a esta función.

El módulo USB utiliza las funciones siguientes:

USB_INIT() Inicializa el módulo USB .

USB_PUT_PACKET() envía un paquete al HOST.

USB_KBHIT() devuelve TRUE si hay datos enviados por HOST(PC).

USB_RX_PACKET_SIZE() devuelve el tamaño del paquete enviado por el HOST(PC).

USB_GET_PACKET() realiza la recepción de la información enviada por el HOST(PC).

USB_DETACH() Se Desconecta y se pone en estado DETACHED

USB_ATTACH() Se Conecta con el Host.

USB_ATTACHED() Devuelve TRUE si USB_CON_SENSE_PIN = TRUE el pin que indica si está conectado o no.

USB_TASK() Monitorea el estado de la conexión conectándose y desconectándose automáticamente.

USB_ENUMERATED() Devuelve TRUE si el HOST(PC) ya enumero el dispositivo o sea si Windows ya lo detecto.

USB_WAIT_FOR_ENUMERATION() Espera infinitamente hasta que el dispositivo fue enumerado.

USB_KBHIT() Devuelve TRUE si hay datos enviados por HOST(PC).

4.10 Memoria SD

4.10.1 Características

Las tarjetas SD (Secure Digital) son memorias flash utilizadas por una gran cantidad de dispositivos, principalmente dispositivos portátiles, como videocámaras, teléfonos celulares, etc. Evolucionaron a partir de las Multi Media Card .

Existen 3 modos de transferencia soportados por SD: Modo SPI (entrada separada serial y salida serial), Modo un-bit SD (separa comandos, canales de datos y un formato propietario de transferencia), Modo cuatro-bit SD (utiliza terminales extra más algunas terminales reasignadas) para soportar transferencias paralelas de cuatro bits.

Tabla 2 Características técnicas de memorias MMC y SD

	MMC	RS-MMC	MMC Plus	Secure MMC	SD	SDIO	miniSD	micro SD
Pines	7	7	13	7	9	9	11	8
Ancho	24 mm	24 mm	24 mm	24 mm	24 mm	24 mm	20mm	11 mm
Largo	32 mm	18 mm	32 mm	32 mm	32 mm	32+ mm	21'5 mm	15 mm
Grosor	1,4 mm	1,4 mm	1,4 mm	1,4 mm	2,1 mm	2,1 mm	1,4 mm	1 mm
Modo SPI	Opcional	Opcional	Opcional	Necesario	Necesario	Necesario	Necesario	Opcional
Modo 1 bit	Sí	Sí	Sí	Sí	Sí	Sí	Sí	Sí
Modo 4 bits	No	No	Sí	Sí	Opcional	Opcional	Opcional	Opcion

								al
Modo 8 bits	No	No	Sí	Sí	No	No	No	No
Reloj xfer	0–20 MHz	0–20 MHz	0–54 MHz	¿0–20 MHz?	0–25 MHz	0–25 MHz	0–20 MHz	0–12 MHz
XFER máximo	20 Mbit/s	20 Mbit/s	416 Mbit/s	¿20 Mbit/s?	100 Mbit/s	100 Mbit/s	100 Mbit/s	100 Mbit/s
SPI XFR máximo	20 Mbit/s	20 Mbit/s	54 Mbit/s	20 Mbit/s	25 Mbit/s	25 Mbit/s	25 Mbit/s	25 Mbit/s
Compatible con desarr. de código abierto	Sí	Sí	No	Sí	Solo SPI	Solo SPI	Solo SPI	Solo SPI

4.10.2 Protocolo SPI

El protocolo SPI se utiliza únicamente como vía de transmisión y recepción de datos ,y no como el protocolo de transferencia completo. La implementación SPI de las tarjetas de memoria SD usa un subconjunto del protocolo de la memoria SD y de los comandos.

La conexión serie consta de cuatro pines:

• SCLK (Serial Clock) es el reloj, manejado por el dispositivo maestro.

• MISO (Master In Slave Out data): une la entrada de datos con la salida de datos de la tarjeta SD.

• MOSI (Master Out Slave In data): conecta la salida de datos del maestro a la entrada de datos de la SD.

• SS (Slave Selector): permite seleccionar distintos dispositivos.

Físicamente los pines en una memoria micro SD se organizan como se muestra en la figura 16.

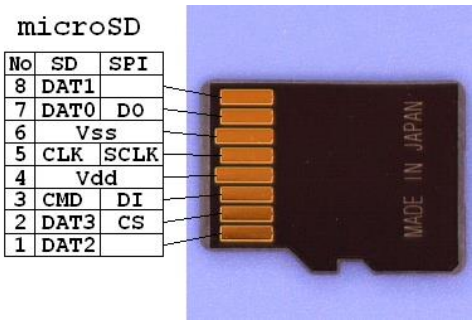


Fig. 16 distribución de pines micro SD

Los pines útiles en el protocolo SPI son:

DI : entrada de datos.

DO: salida de datos

CLK : señal de reloj

CS : chip select, activa a nivel bajo.

El protocolo de comunicación define el envío de seis Bytes mediante un formato de envío de bytes para interactuar con la memoria SD, en la figura 17 se muestra el formato para comunicación SPI en un SD.

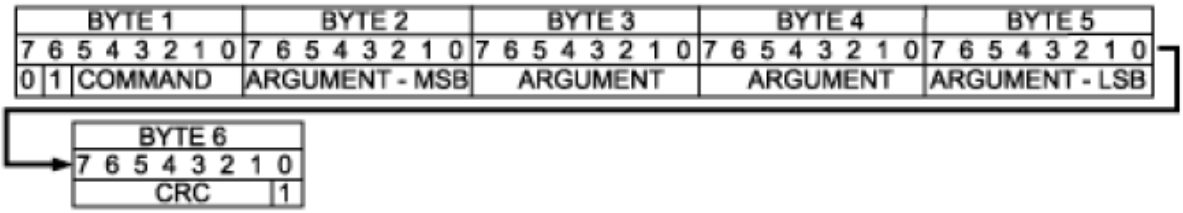


Figura 17 Formato de envío de comandos en memoria SD

Con cada comando recibido, la memoria envía un byte de respuesta, cada bit de la respuesta indica el estado de la memoria o algunos errores durante el proceso de lectura y escritura. La siguiente figura18 describe la información que de cada bit.

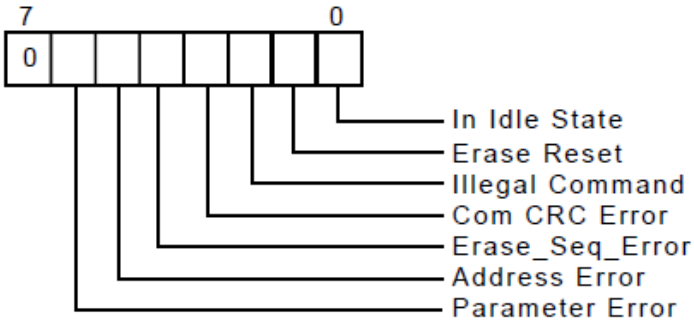


Figura 18 Bits de respuesta de memoria SD

Los comandos utilizados para comunicación mediante el protocolo SPI son:

Tabla 3 Comandos SPI de la memoria SD

CMD INDEX	SPI Mode	Argument	Resp	Abbreviation	Command Description
CMD0	Yes	None	R1	GO_IDLE_STATE	Resets the SD Card
CMD1	Yes	None	R1	SEND_OP_COND	Activates the card's initialization process.
CMD2	No				
CMD3	No				
CMD4	No				
CMD5	Reserved				
CMD6	Reserved				
CMD7	No				
CMD8	Reserved				
CMD9	Yes	None	R1	SEND_CSD	Asks the selected card to send its card-specific data (CSD).
CMD10	Yes	None	R1	SEND_CID	Asks the selected card to send its card identification (CID).
CMD11	No				
CMD12	Yes	None	R1b	STOP _TRANSMISSION	Forces the card to stop transmission during a multiple block read operation.
CMD13	Yes	None	R2	SEND_STATUS	Asks the selected card to send its status register.
CMD14	No				
CMD15	No				
CMD16	Yes	[31:0] block length	R1	SET_BLOCKLEN	Selects a block length (in bytes) for all following block commands (read & write).[1]
CMD17	Yes	[31:0] data address	R1	READ_SINGLE _BLOCK	Reads a block of the size selected by the SET_BLOCKLEN command.[2]
CMD18	Yes	[31:0] data address	R1	READ_MULTIPLE _BLOCK	Continuously transfers data blocks from card to host until interrupted by a STOP_ TRANSMISSION command.
CMD19	Reserved				
CMD20	No				
CMD21 ... CMD23	Reserved				
CMD24	Yes	[31:0] data address	R1[3]	WRITE_BLOCK	Writes a block of the size selected by the SET_BLOCKLEN command.[4]
CMD25	Yes	[31:0] data address	R1	WRITE_MULTIPLE_BLOCK	Continuously writes blocks of data until a stop transmission token is sent (instead of 'start block').

CMD INDEX	SPI Mode	Argument	Resp	Abbreviation	Command Description
CMD26	No				
CMD27	Yes	None	R1	PROGRAM_CSD	Programming of the programmable bits of the CSD.
CMD28[1]	Yes	[31:0] data address	R1b	SET_WRITE_PROT	If the card has write protection features, this command sets the write protection bit of the addressed group. The properties of write protection are coded in the card specific data (WP_GRP_SIZE).
CMD29[4]	Yes	[31:0] data address	R1b	CLR_WRITE_PROT	If the card has write protection features, this command clears the write protection bit of the addressed group.
CMD30	Yes	[31:0] write protect data address	R1	SEND_WRITE_ PROT	If the card has write protection features, this command asks the card to send the status of the write protection bits.[2]
CMD31	Reserved				
CMD32	Yes	[31:0] data address	R1	ERASE_WR_BLK_START _ADDR	Sets the address of the first write block to be erased.
CMD33	Yes	[31:0] data address	R1	ERASE_WR_BLK_END _ADDR	Sets the address of the last write block in a continuous range to be erased.
CMD34 .... CMD37	Reserved				
CMD38	Yes	[31:0] don't care*	R1b	ERASE	Erases all previously selected write blocks.
CMD39	No				
CMD40	No				
CMD41 ... CMD54	Reserved				
CMD55	Yes	[31:0] stuff bits	R1	APP_CMD	Notifies the card that the next command is an application specific command rather than a standard command.
CMD56	Yes	[31:0] stuff bits [0]: RD/WR.[3]	R1	GEN_CMD	Used either to transfer a Data Block to the card or to get a Data Block from the card for general purpose/application specific commands. The size of the Data Block is defined with SET_BLOCK_LEN command.
CMD57	Reserved				
CMD58	Yes	None	R3	READ_OCR	Reads the OCR register of a card.
CMD59	Yes	[31:1] don't care* [0:0] CRC option	R1	CRC_ON_OFF	Turns the CRC option on or off. A '1' in the CRC option bit will turn the option on, a '0' will turn it off.
CMD60-63	No				

5 Desarrollo

5.1.1 Solución propuesta

En el diseño propuesto, se planteó el monitoreo de un canal del potencial extracelular, sin embargo debido a que los encapsulados poseen varios amplificadores operacionales, se decidió aprovechar todo su potencial, por lo que la placa del sistema incluye cuatro canales.

Para la obtención de la señal extracelular, se plantea una solución con las siguientes etapas: Preamplificadores, Filtro pasa banda y procesamiento digital.

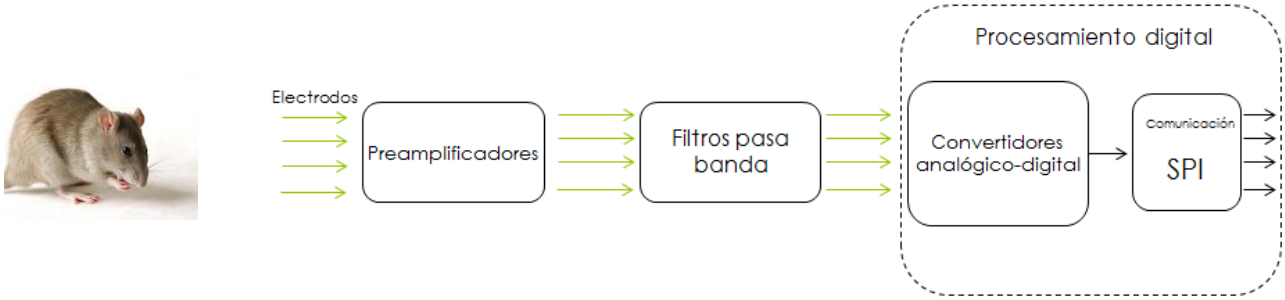


Figura 19 Etapas de sistema de adquisición

5.1.2 Preamplificadores

En esta etapa la señal de los electrodos es acoplada y amplificada, esta operación se realiza mediante un amplificador de instrumentación AD8222 con una razón de rechazo en modo común (CMRR) mínimo de 80dB a una ganancia igual a uno, siendo 110dB el CMRR estimado para ganancias superiores a 100 y menores de 1000 esta característica permite aumentar la relación señal a ruido (SNR) del sistema, obteniendo señales limpias y más exactas.

El amplificador seleccionado además de las características eléctricas antes mencionadas, posee dimensiones que permiten un desarrollo compacto y ligero, en la figura 20 siguiente se muestra el encapsulado que contiene los amplificadores de instrumentación y el amplificador utilizado,así como sus principales medidas.

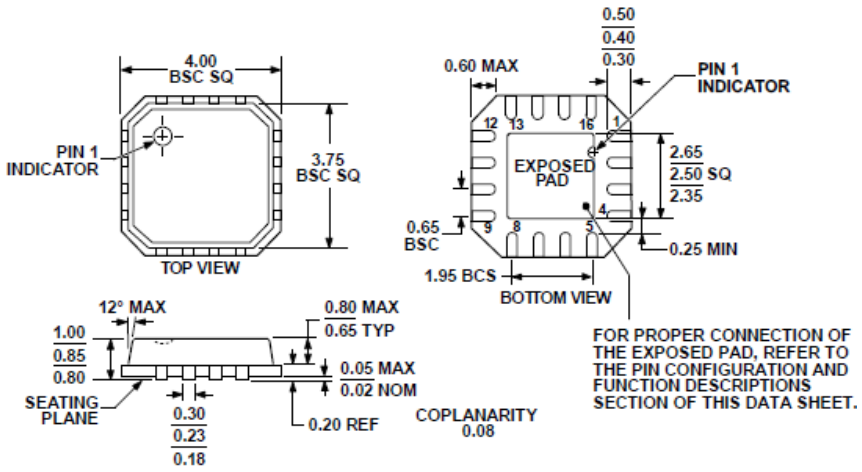


Fig. 20 Especificaciones mecánicas del amplificador de instrumentación

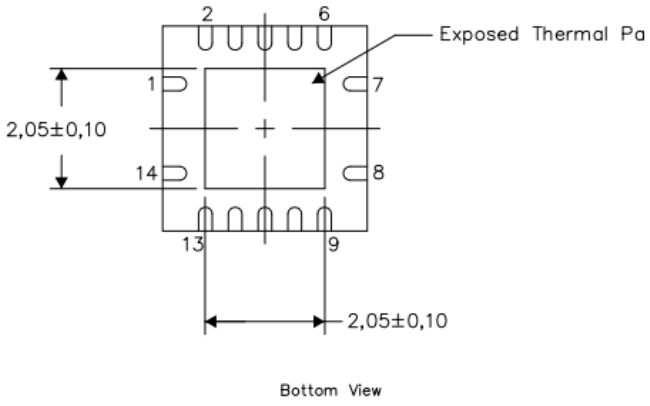


Figura 21 Especificaciones mecánicas del amplificador con sus medidas en milímetros

En esta etapa se propuso una ganancia en el amplificador de instrumentación de G=149.6, y según la ecuación del amplificador se necesita una resistencia de R_G= 330Ω.

$$R_G = \frac{49.4\ k\ \Omega}{G - 1}$$

En esta etapa se ha agregado un corrector de línea de base cuyos cálculos se realizan en el apartado de filtros. A continuación se muestra el diagrama eléctrico de esta etapa para un canal.

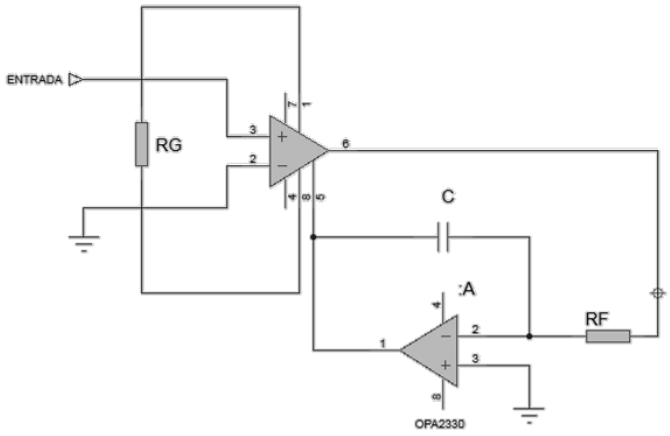


Figura 22 Diagrama eléctrico del preamplificador

Figura 23 Diseño físico de la etapa de preamplificadores (cuatro canales)

5.1.3 Filtrado

En la etapa de filtrado, la señal amplificada es primero tratada mediante un corrector de línea de base, conectado a la referencia del amplificador de instrumentación con una frecuencia de corte de 0.8Hz, los componentes, calculados mediante la ecuación:

$$RC = \frac{1}{2\pi f_c}$$

Son: R=198kΩ, C=1µF, posteriormente la señal es conducida a un filtro pasa bajos con frecuencia de corte de 7.2 KHz de segundo orden con topología VCVS , cuyos componentes calculados a partir de la ecuación siguiente :

$$f_c = \frac{1}{2\pi RC}$$

Son: R1=R2=RB=2.2kΩ, RA=1kΩ, C1=C2=22nF. Es importante mencionar que en esta etapa existe un factor de amplificación de 3.2 debido a las resistencias conectadas a la entrada inversora.

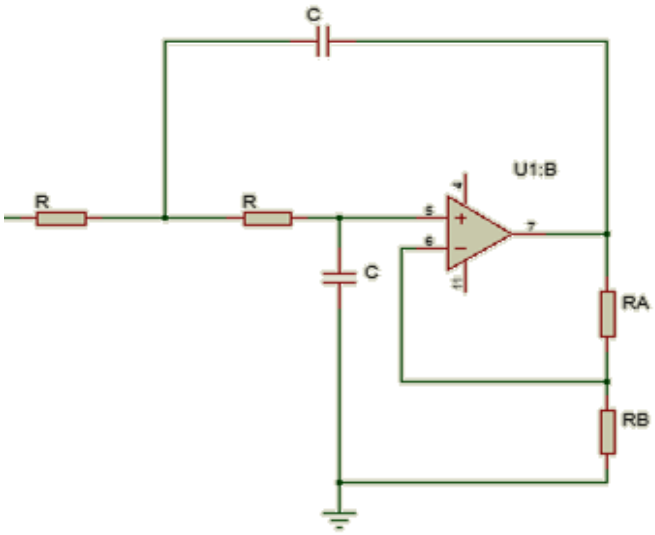


Figura 24 Diagrama de filtro pasa bajas

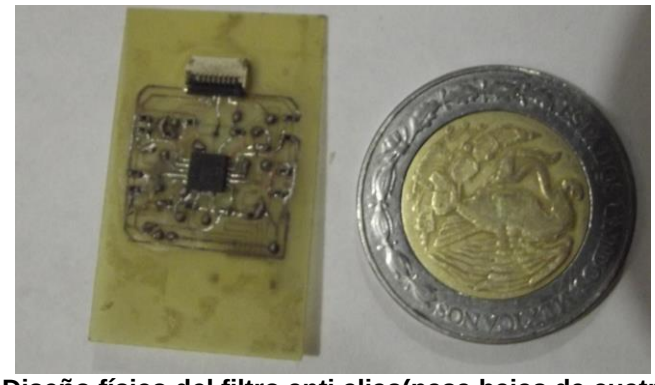


Figura 25 Diseño físico del filtro anti alias(pasa bajas de cuatro canales)

5.1.4 Acondicionamiento

Para la digitalización de la señal filtrada resulta necesario que dicha señal pueda ser leída por el micro controlador. Para realizar esta acción se monta la señal en una componente de corriente directa igual a la mitad de la señal de alimentación positiva. Además se agrega al circuito un diodo de protección para las entradas analógicas del PIC .El circuito sumador siguiente realiza la operación planteada.

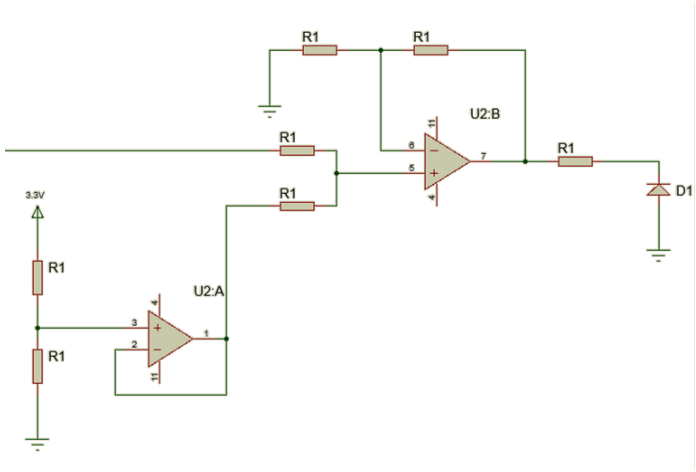


Figura 26 Circuito de acoplamiento para el PIC

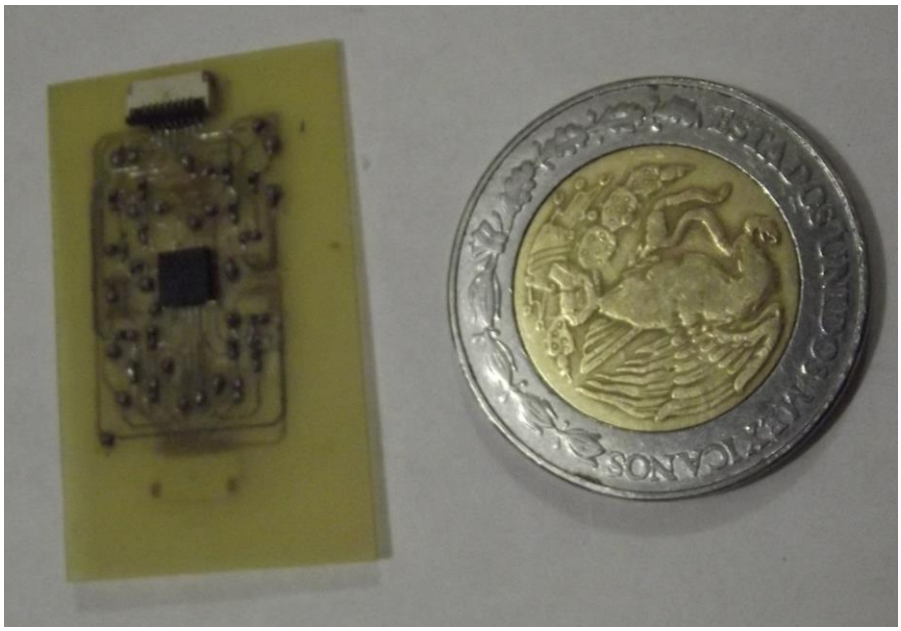


Figura 27 Diseño físico del circuito de acondicionamiento de señal (cuatro canales)

El circuito completo para adquisición de un canal se muestra en la figura 28:

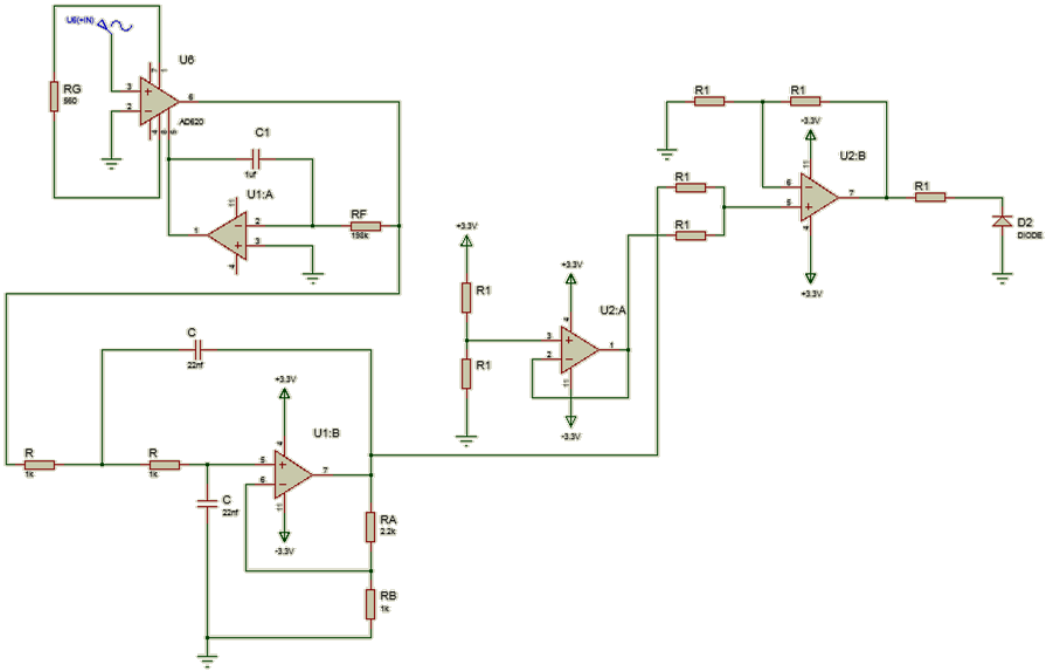


Figura 28 Circuito de adquisición y filtrado de potencial extracelular para un canal

5.1.5 Digitalización

Una vez que la señal esta filtrada y lista, para la digitalización esta se realiza mediante el convertidor analógico digital del PIC18f2550 , la captura de los datos se realiza cada 20us, es decir se capturan 50k datos por segundo. Posteriormente los datos son enviados a la memoria SD mediante el protocolo de comunicación SPI.

Para llevar a cabo la conversión analógica digital se configuro el dispositivo, se configura en el conversor analógico digital, el puerto A0 como entrada analógica, los 3.3V como voltaje de referencia y la velocidad de conversor a aproximadamente 700 ns . Las funciones utilizadas son:

```
 setup_adc_ports(AN0);
setup_adc( VSS_VDD );
set_adc_channel(0);
setup_adc( ADC_CLOCK_div_32 );
```

Para cronometrar el tiempo de adquisición se configuro el timer 0, para que genere una interrupción cada 20uS mediante:

```
SETUP_TIMER_0(RTCC_INTERNAL|RTCC_DIV_1|RTCC_8_bit);
enable_interrupts(INT_TIMER0);
set_TIMER0(0x10);
```

El dispositivo SPI se configura para que el clk funcione 4 veces más lento que el reloj del PIC, además se configura el inicio de la transmisión y el pulso de transmisión con los que trabaja la memoria SD mediante el comando:

```
SETUP_SPI(SPI_MASTER|SPI_CLK_DIV_4|SPI_H_TO_L|SPI_XMIT_L_TO_H|SPI_SAMPLE_AT_END );
```

El código complete de la digitalización y envío de datos se puede consultar en el apéndice A. La figura 29 muestra el proceso de adquisición y almacenamiento de datos.

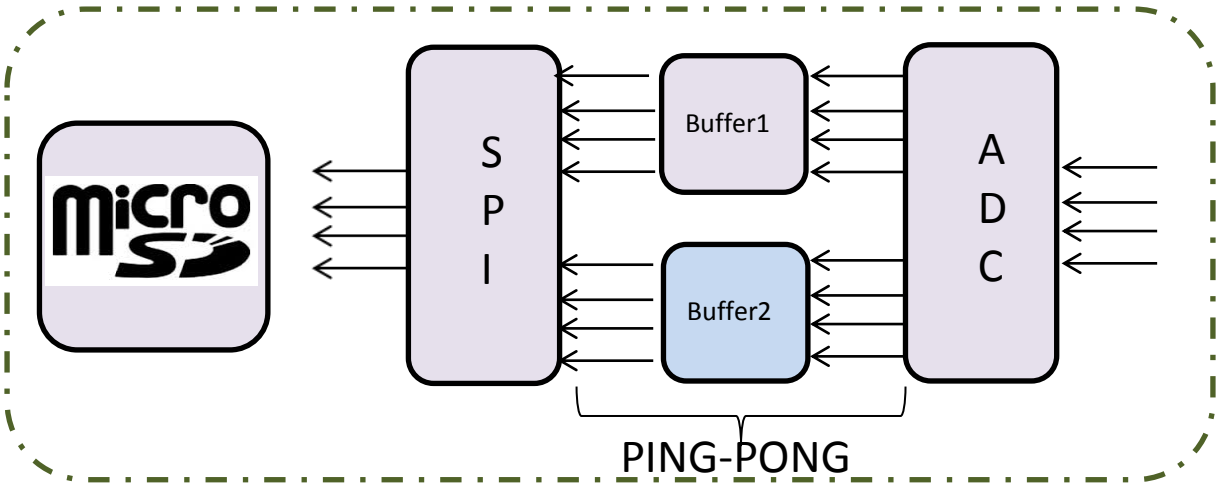


Figura 29 Diagrama del programa de digitalización

El programa descrito en el esquema anterior funciona de la siguiente manera.

- Los datos son convertidos o digitalizados cada 20us mediante la interrupción del timer, además con cada llamada a la interrupción se guardan en uno de los dos buffer mostrados en el diagrama el cual llamaremos buffer de conversión.
- Cuando el buffer de conversión se llena se convierte en buffer de envío, se enciende una bandera, y se comienza a llenar el otro buffer(ahora llamado buffer de conversión), la bandera permite que el código principal envíe el buffer de envío a la memoria SD, mientras el otro es llenado.
- Este proceso se repite y se denomina ping-pong.
- Cada segundo aproximadamente, se enciende un led y se guarda dentro de la serie de datos el momento en el que se encendió Esto permite la sincronización de los datos, con el archivo de video.

Las conexiones eléctricas necesarias para el funcionamiento del circuito de digitalización se muestran en la figura 30:

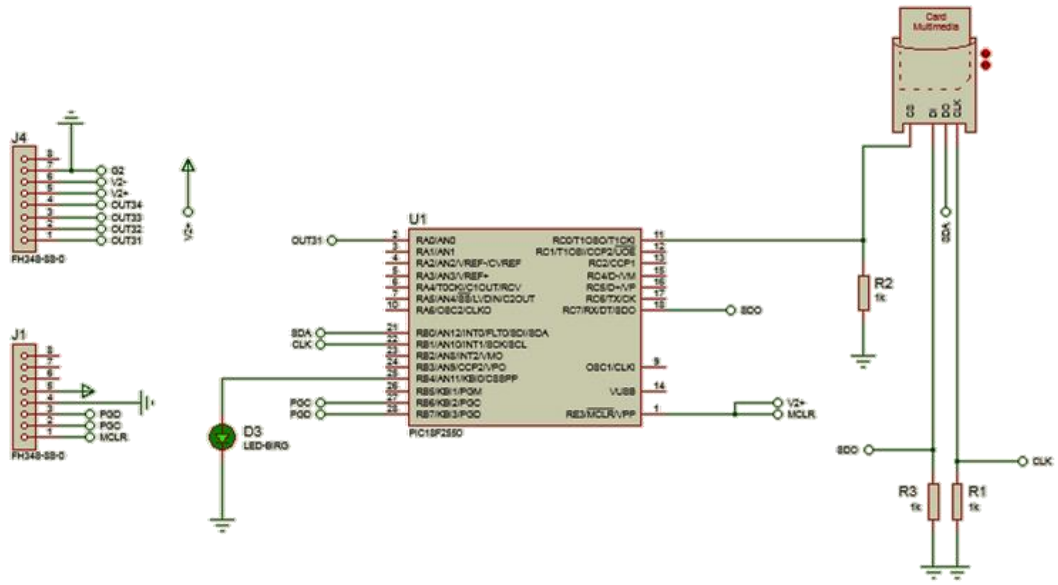


Figura 30 Diagrama del circuito de digitalización

5.1.6 Procesamiento de datos

La lectura de los datos así como su procesamiento se lleva a cabo mediante el programa de procesamiento Matlab. Para la lectura de los datos se diseñó un circuito, similar al de digitalización que incluye el puerto USB para comunicación con el programa echo en Matlab, cabe señalar que aunque existen programas para la lectura de memorias SD, el circuito se diseñó con la intensión de poder conectar el software para monitoreo en tiempo real en futuras versiones del dispositivo.

La lectura de datos, se realiza mediante el puerto usb, este es configurado en el PIC, utilizando la comunicación Bulk-transfer, en este tipo de comunicación, no se tiene prioridad dentro de la aplicaciones en la computadora, sin embargo permite la transmisión masiva de datos utilizando varios end points o canales de comunicación en un solo envío. Lo importante del envío masivo de datos radica en la configuración tanto en la biblioteca del PIC como en el programa en Matlab del número de canales que se utilizaran, además es importante cuidar que el id del proveedor y del producto sean iguales para poder llevar a cabo la comunicación. Dentro de Matlab se configuran estos end points mediante la librería mpusbapi de la cual obtenemos los apuntadores de cada canal mediante:

```
out_pipe1=libpointer('int8Ptr',[uint8('\MCHP_EP1') 0]);
```

Con los cuales podemos llamar a funciones de la misma librería, como:

MPUSBOpen, MPUSBWrite o MPUSBRead.

Para la configuración de los canales dentro del micro controlador, se modifican los descriptores que se incluyen en el código de transmisión, de la misma manera que se declaran en el programa principal. Los códigos completos de os descriptores en el micro controlador así como la configuración de los end points en Matlab se muestran en el apéndice A.

Para visualización de los datos guardados se programó la siguiente GUI que facilita al usuario el manejo de dichos datos, y como futura interfaz para monitoreo en tiempo real.

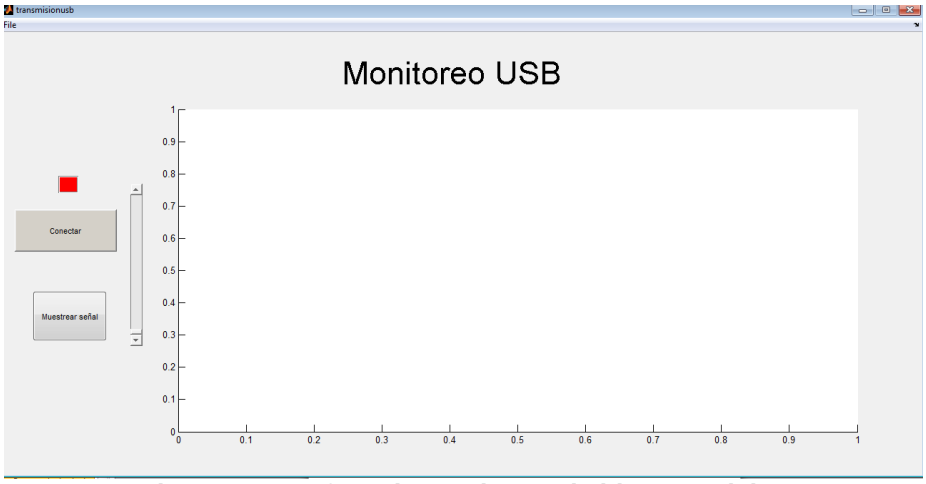


Figura 31 Interface de monitoreo de biopotenciales

Para la obtención de característica (específicamente oscilaciones de alta frecuencia) de las señales monitoreadas se programó una GUI en Matlab que lee los datos de previamente guardados en una estructura con los campos siguientes :

title: ' Nombre_del registro'
comment: 'No comment'
interval: intervalo de muestreo p. ej. 5.0000e-004
scale: Escala
offset: 0
units: 'mV'
start: 0
length: numero de muestras1585
values: valores registrados [1585x1 double]

Es necesario que los datos se guarden con este formato para que sean leídos por el script de Matlab. Se ha programado dentro de Matlab el reconocimiento de este tipo de archivo ya que permiten compartir de manera directa los archivos generados por otros programas como spike. El análisis de la señal pretende localizar oscilaciones de alta frecuencia, dentro de la señal cerebral, dichas oscilaciones se encuentran entre la banda de frecuencias de 80Hz a 500Hz, divididas en dos grupos: oscilaciones de alta frecuencia de banda de paso inferior

80HZ a 150Hz y la banda de paso inferior para frecuencias mayores que 200Hz , como se muestra en la figura 32.

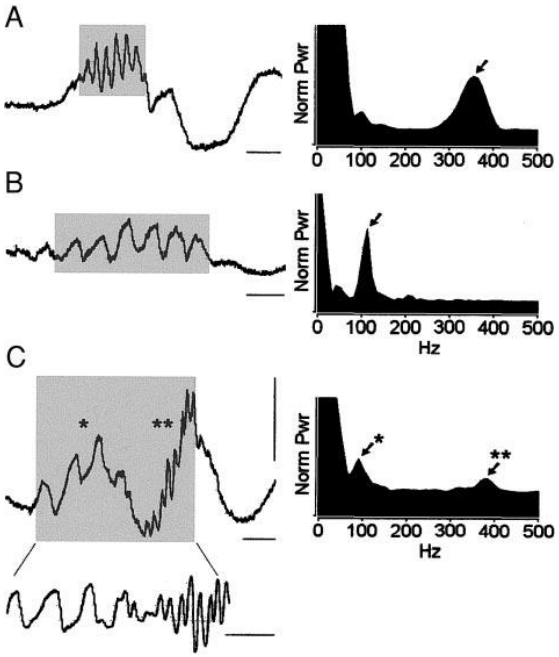


Figura 32 Oscilaciones de alta frecuencia en sñal cerebral[20]

Para la localización de oscilaciones de alta frecuencia dentro del registro de potencial cerebral, se llevan a cabo los pasos siguientes [20]:

- Filtrado de la señal dentro del rango 80Hz-500Hz
- Se calcula el valor RMS de toda la señal filtrada
- Se divide la señal en ventanas de 3ms y se calcula el valor RMS de cada ventana.
- Para encontrar un candidato a oscilación de baja frecuencia se buscan ventanas sucesivas (por lo menos dos) cuyo valor RMS supere por cinco desviaciones estándar el valor RMS de toda la señal.
- Se define como oscilación de alta frecuencia a aquellos candidatos que contienen por lo menos seis picos con amplitud superior a tres desviaciones estándar por encima de la media, de la señal filtrada y rectificada.

Para facilitar la manipulación de las variables que definen la detección de oscilaciones de alta frecuencia se programó una interfaz gráfica de usuario que permite cambiar dichos parámetros así como, guardar los resultados obtenidos, para su análisis y registro. El código del monitor de señal así como del programa para búsqueda de oscilaciones de alta frecuencia se presenta completo en el apéndice A La figura 33 muestra la imagen de la interfaz Ripples, que busca las oscilaciones de alta frecuencia.

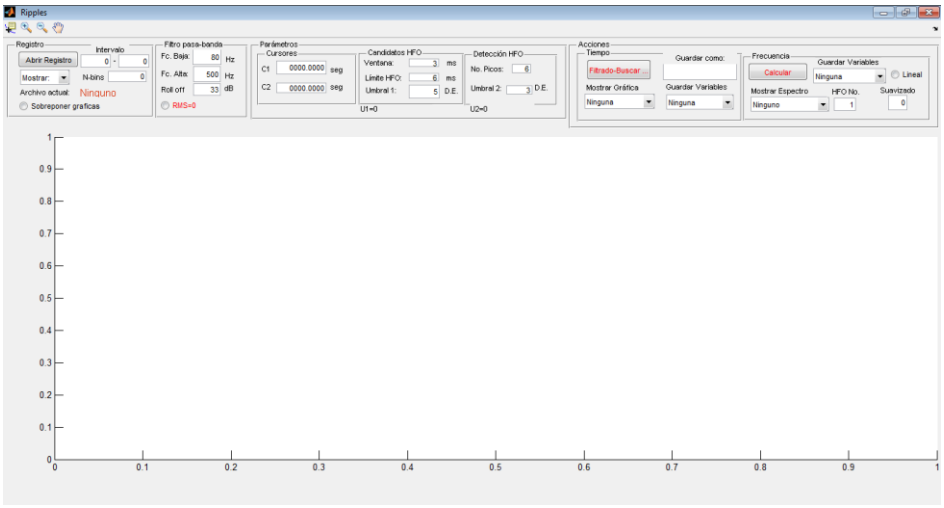


Figura 33 Interfaz Ripples.

En la figura 34 ilustra este proceso.

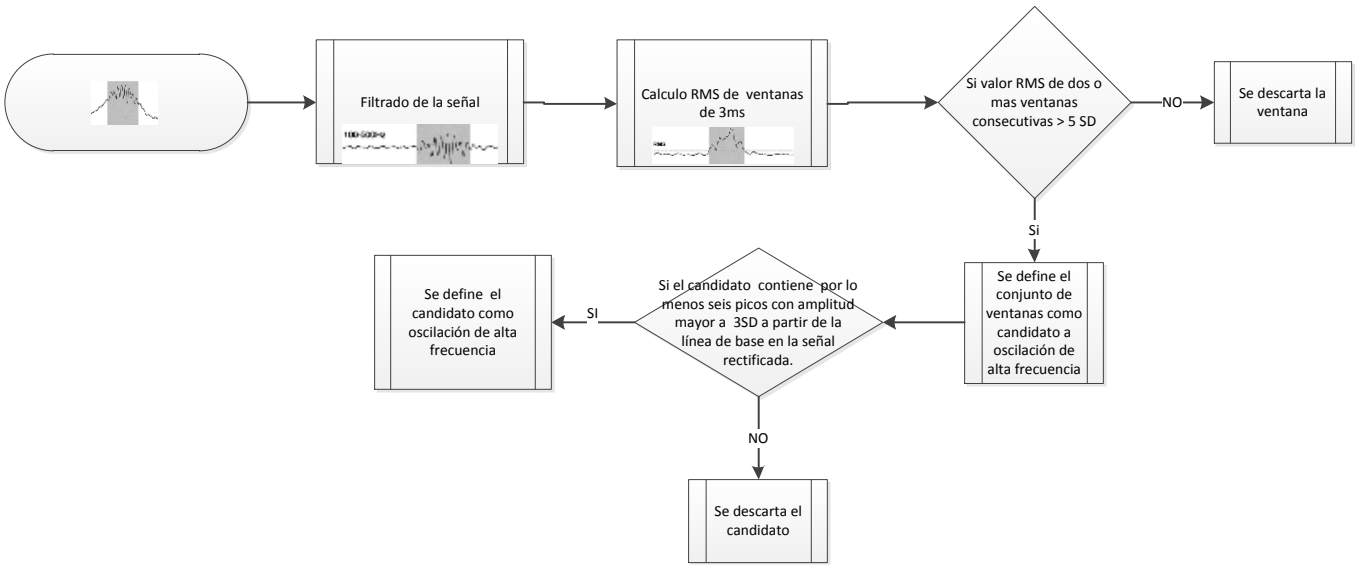


Figura 34 Diagrama de bloques del programa Ripples

6 Pruebas

6.1 Circuito de adquisición y acondicionamiento de señal

Se realizaron dos series de pruebas al circuito de adquisición y acoplamiento de la señal, el cual está compuesto por el amplificador de instrumentación, la etapa de filtro en la banda de 0.8Hz a 7.2kHz y la etapa de offset.

En la primera serie de pruebas se introdujo mediante un generador de funciones de la marca tenma 72-7650 una señal seno con amplitud de 5mV pico-pico, se realizó un barrido de frecuencias desde 1Hz hasta 10kHz , tomando la medición de la amplitud y offset de la salida en cada etapa del circuito, mediante un osciloscopio marca Tektronix TDS2004B, con el fin de comprobar su correcta respuesta en frecuencia.

En la segunda serie de pruebas se utilizó la misma metodología que en la primera serie , sin embargo se añadió ruido a las entrada del amplificador de instrumentación, con el fin de observar su rechazo en modo común así como, la correcta eliminación de frecuencia por parte de los filtro, que no están dentro de la banda de paso, en esta serie de pruebas se añadió un offset a la señal de entrada para verificar que se llevara a cabo el rechazo de la componente de CD .

6.2 Digitalización y almacenamiento de la señal

Para comprobar la adquisición y almacenamiento de la señal, se procedió de la siguiente manera:

Mediante un generador de funciones marca Tektronix TDS2004B se programan señales de 100Hz,1kHz,3kHz,7kHz,8kHz, posteriormente con un circuito utilizado como modelo del PIC18f2550 se digitalizo dicha señal mediante el convertidor analógico digital y los datos se guardaron en una memoria micro SD con capacidad de 2Gb marca SanDisk utilizando el protocolo SPI del mismo micro controlador.

Las señales se guardan en dos periodos distintos, primero se guardan durante cinco segundos, en esta prueba se verifica que el muestreo de la señal sea correcta, es decir se observa que la señales puedan describirse mediante sus muestra (mínimo cinco muestras por periodo).

La segunda prueba se realiza guardando la señal durante 15 minutos, con el propósito de verificar que el sistema no se satura durante periodos largos de registro.

6.3 Procesamiento de datos

Se construyó una placa de desarrollo para realizar las pruebas de adquisición de señales, la placa utilizada se muestra en la figura 35:

Figura 35 Circuito Físico y diseño para las pruebas de adquisición

Las pruebas se realizaron utilizando una señal de entrada de 1kHz , obtenida mediante el generador de funciones (Tektronics) la cual se introduce en el puerto A0 del convertidor analógico digital y es enviada mediante el puerto USB a la aplicación en Matlab.

Se utiliza una frecuencia de 1kHZ debido a que se utiliza la comunicación bulk sin prioridad y Matlab tiene esperas mínimas de muestreo de 1ms, lo cual podría registrar adecuadamente en el mejor de los casos señales de 1kHz utilizando únicamente la memoria RAM del micro controlador como buffer, en caso de retardos mayores a 1ms. La señal es almacenada y graficada mediante la interface de Matlab.

6.4 Pruebas de Ripples

En el programa de obtención de oscilaciones de alta frecuencia, se utilizaron graficas de cada etapa de la búsqueda para revisar manualmente el cálculo correcto de los parámetros de la señal.

Para llevar a cabo tal análisis se utilizaron registros previamente guardados mediante spike, estos registros fueron tomados de ratones las cuales se les induce el fenómeno que desencadena la presencia de oscilaciones de alta frecuencia. Como señal de control se utilizó una onda seno pura, ya que de esta señal se conocen fácilmente los parámetros y pueden compararse con los obtenidos en las señales cerebrales.

La primera etapa de las pruebas consistió en buscar oscilaciones en las señales de los ratones, y revisar que los parámetros así como el proceso se realizará adecuadamente en cada paso.

Posteriormente se realizaron las mismas pruebas con una señal seno generada en Matlab, con el fin de descartar errores en el procesamiento y los ciclos de cálculo del algoritmo, una vez que los cálculos se realizaron correctamente.

6.5 Pruebas de sincronización

Para llevar a cabo las pruebas de sincronización, se realizaron tres tipos de pruebas: primeramente se utiliza 20 videos grabados con una cámara web marca Logitech 860-000375 con fuentes luminosas y mediante un programa , se detecta en que parte del video hay fuentes luminosas, segmentando la fuente de luz .

La segunda prueba se utilizan 10 videos grabados con una cámara web marca Logitech 860-000375 en los que la fuente luminosa sea intermitente con periodos variables. Mediante un programa escrito en Matlab se genera una gráfica, que contenga en que frame está presente la fuente luminosa y en cuáles no.

Finalmente se cargan los videos grabados con una cámara web marca Logitech 860-000375 sobre un led intermitente con periodo regular y la señal obtenida mediante el circuito la cual contiene la ocurrencia de los pulsos, y se sincronizan el video y la señal se muestran simultáneamente en la interface creada.

7 Resultados

7.1 Circuito de adquisición y acondicionamiento de señal

La señal tomada en el filtro pasa bajos así como la frecuencia de entrada se presentan en la tabla 4.

Tabla 4 Voltajes de prueba en la etapa de filtrado y acondicionamiento

Frecuencia	Vpp de entrada al filtro pasa bajas	Vpp de salida del filtro pasa bajas	Ganancia filtro	Offset	Vpp de salida al sumador	Ganancia del sumador	Offset
1Hz	0.75	1.085	1.44	0V	1.2	0.90417	1.5V
10Hz	0.75	1.089	1.45	0V	1.193	0.91282	1.5V
100Hz	0.75	1.084	1.44	0V	1.195	0.90711	1.5V
1kHz	0.75	1.085	1.44	0V	1.19	0.91176	1.5V
2kHz	0.75	1.069	1.42	0V	1.18	0.90593	1.5V
3kHz	0.75	1.035	1.38	0V	1.14	0.90789	1.5V
4kHz	0.75	0.984	1.31	0V	1.08	0.91111	1.5V
5kHz	0.75	0.906	1.20	0V	1	0.906	1.5V
6kHz	0.75	0.814	1.08	0V	0.895	0.9095	1.5V
7kHz	0.75	0.721	0.96	0V	0.79	0.91266	1.5V
8kHz	0.75	0.619	0.82	0V	0.68	0.91029	1.5V
9kHz	0.75	0.534	0.71	0V	0.58	0.92069	1.5V
10kHz	0.75	0.458	0.61	0V	0.5	0.916	1.5V

Las gráficas de ganancia contra frecuencia se muestran en la figura 36.

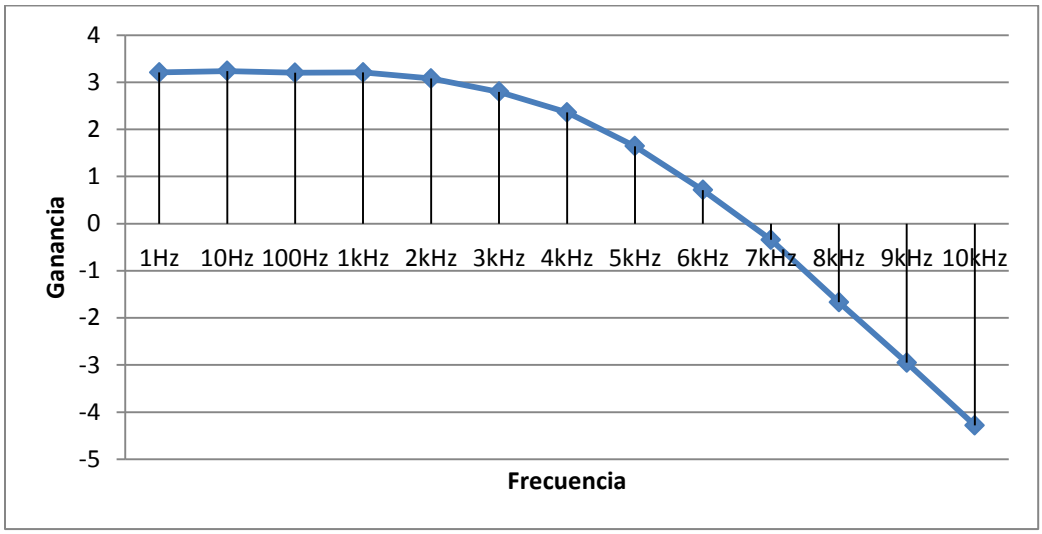


Figura 36 Respuesta del filtro pasa bajas

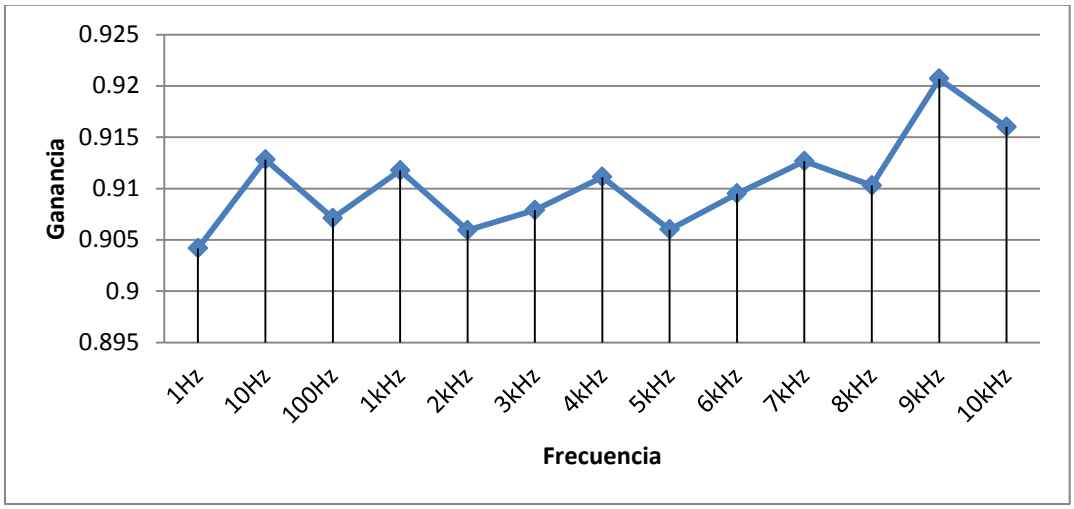


Figura 37 Respuesta del sumador (promedio=0.910)

7.2 Digitalización y almacenamiento de la señal

A continuación se presentan los datos almacenados en la memoria SD. Se han tomado cinco segundos de lectura pero por simplicidad solo se muestra aquí un ciclo para la señal seno de entada de tres volts pico-pico y offset de 1.5V.

Señal de entrada de 100Hz:

3.0000 3.0000 3.0000 3.0000 3.0000 3.0000 2.9804 2.9804 2.9804 2.9804 2.9804 2.9804
2.9804 2.9608 2.9608 2.9608 2.9412 2.9412 2.9412 2.9412 2.9216 2.9216 2.9216 2.9020
2.9020 2.8824 2.8824 2.8627 2.8627 2.8431 2.8431 2.8235 2.8235 2.8039 2.8039 2.7843
2.7843 2.7647 2.7451 2.7451 2.7255 2.7059 2.6863 2.6863 2.6667 2.6471 2.6471 2.6275
2.6078 2.5882 2.5882 2.5686 2.5490 2.5294 2.5098 2.4902 2.4706 2.4510 2.4314 2.4314
2.4118 2.3922 2.3725 2.3529 2.3333 2.3137 2.2941 2.2745 2.2549 2.2353 2.2157 2.1961
2.1765 2.1569 2.1373 2.1176 2.0980 2.0784 2.0392 2.0196 2.0000 1.9804 1.9608 1.9412
1.9216 1.9020 1.8627 1.8431 1.8235 1.8039 1.7843 1.7647 1.7451 1.7059 1.6863 1.6667
1.6471 1.6275 1.6078 1.5686 1.5490 1.5294 1.5098 1.4902 1.4706 1.4314 1.4118 1.3922
1.3725 1.3529 1.3333 1.2941 1.2745 1.2549 1.2353 1.2157 1.1961 1.1569 1.1373 1.1176
1.0980 1.0784 1.0588 1.0392 1.0196 0.9804 0.9608 0.9412 0.9216 0.9020 0.8824 0.8627
0.8431 0.8235 0.8039 0.7843 0.7647 0.7451 0.7255 0.7059 0.6863 0.6667 0.6471 0.6275
0.6078 0.5882 0.5686 0.5490 0.5294 0.5098 0.4902 0.4902 0.4706 0.4510 0.4314 0.4118
0.3922 0.3725 0.3725 0.3529 0.3333 0.3333 0.3137 0.2941 0.2941 0.2745 0.2549 0.2353
0.2353 0.2157 0.1961 0.1961 0.1765 0.1765 0.1569 0.1569 0.1373 0.1373 0.1176 0.1176
0.0980 0.0980 0.0980 0.0784 0.0784 0.0588 0.0588 0.0588 0.0392 0.0392 0.0392 0.0392
0.0392 0.0196 0.0196 0.0196 0.0196 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0.0196 0.0196 0.0196 0.0196 0.0392 0.0392
0.0392 0.0392 0.0588 0.0588 0.0588 0.0588 0.0784 0.0784 0.0980 0.0980 0.1176 0.1176
0.1373 0.1373 0.1569 0.1569 0.1765 0.1765 0.1765 0.1961 0.2157 0.2157 0.2353 0.2549
0.2549 0.2745 0.2941 0.2941 0.3137 0.3333 0.3529 0.3529 0.3725 0.3922 0.4118 0.4118
0.4314 0.4510 0.4706 0.4902 0.5098 0.5294 0.5490 0.5490 0.5686 0.5882 0.6078 0.6275
0.6471 0.6667 0.6863 0.7059 0.7255 0.7451 0.7647 0.7843 0.8039 0.8235 0.8431 0.8627
0.8824 0.9020 0.9216 0.9608 0.9804 1.0000 1.0196 1.0392 1.0588 1.0784 1.0980 1.1176
1.1569 1.1765 1.1961 1.2157 1.2353 1.2549 1.2941 1.3137 1.3333 1.3529 1.3725 1.3922
1.4314 1.4510 1.4706 1.4902 1.5098 1.5294 1.5686 1.5882 1.6078 1.6275 1.6471 1.6667
1.7059 1.7255 1.7451 1.7647 1.7843 1.8039 1.8431 1.8627 1.8824 1.9020 1.9216 1.9412
1.9608 1.9804 2.0000 2.0392 2.0588 2.0784 2.0980 2.1176 2.1373 2.1569 2.1765 2.1961
2.2157 2.2353 2.2549 2.2745 2.2941 2.3137 2.3333 2.3529 2.3725 2.3922 2.4118 2.4314
2.4510 2.4706 2.4902 2.5098 2.5294 2.5294 2.5490 2.5686 2.5882 2.6078 2.6078 2.6275
2.6471 2.6667 2.6667 2.6863 2.7059 2.7255 2.7255 2.7451 2.7647 2.7647 2.7843 2.8039
2.8039 2.8235 2.8235 2.8431 2.8431 2.8627 2.8627 2.8824 2.8824 2.9020 2.9020 2.9020
2.9216 2.9216 2.9216 2.9412 2.9412 2.9412 2.9608 2.9608 2.9608 2.9608 2.9804 2.9804
2.9804 2.9804 2.9804 2.9804 3.0000 3.0000 3.0000 3.0000 3.0000 3.0000 3.0000

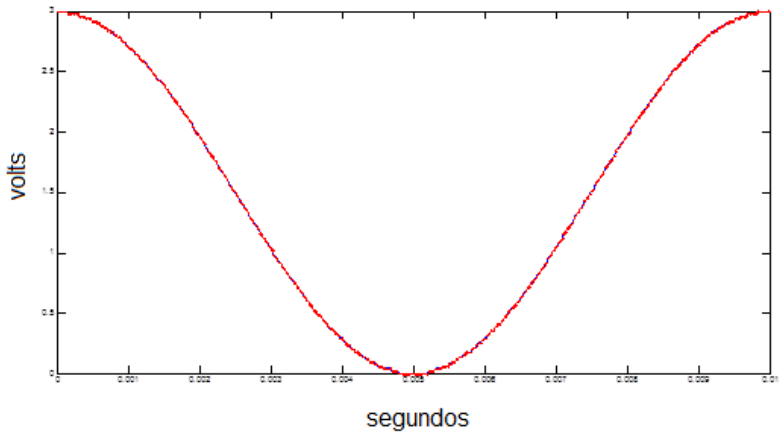


Figura 38 Ciclo con 411 muestras de una señal de 100Hz

En este caso se tienen en promedio 411 elemento en un intervalo de 1ms por lo que cada muestra se toma en promedio cada 24.3 microsegundos.

Señal de entrada de 1kHz:

3.0000 2.9804 2.9216 2.8235 2.7059 2.5686 2.3922 2.1961 1.9804 1.7647 1.5294 1.2941 1.0784 0.8627 0.6667 0.4706 0.3333 0.1961 0.0980 0.0392 0 0 0.0392 0.1176 0.2353 0.3529 0.5294 0.7059 0.9020 1.1373 1.3529 1.5882 1.8039 2.0392 2.2353 2.4314 2.6078 2.7451 2.8627 2.9412 2.9804 3.0000.

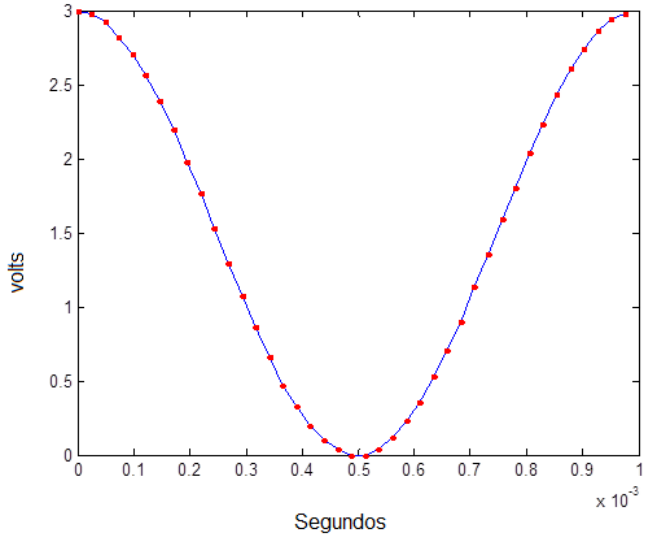


Figura 39 Señal de 1kHz con 42 muestras

En este caso se tienen en promedio 42 elemento en un intervalo de 1ms por lo que cada muestra se toma en promedio cada 23.81 microsegundos.

Señal de entrada de 3kHz:

2.9608 2.6863 2.1569 1.4902 0.8235 0.2941 0.0196 0.0588 0.3922 0.9412 1.6275 2.2745 2.7647 2.9804.

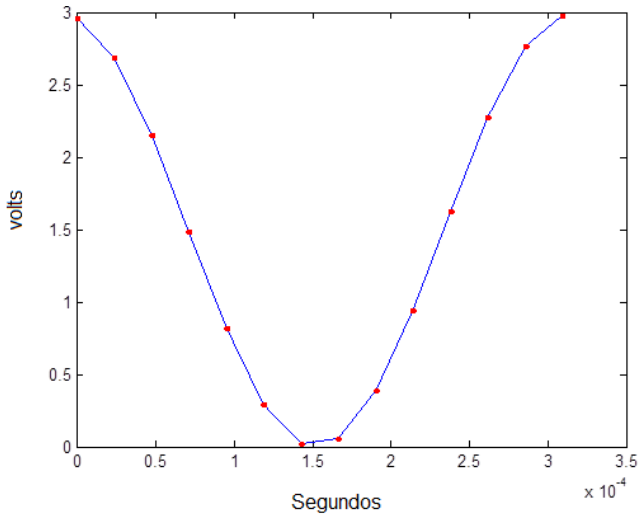


Figura 40 Señal de 3kHz con 14 muestras

En este caso se tienen en promedio14 elemento en un intervalo de 0.33 ms por lo que cada muestra se toma en promedio cada 23.57microsegundos.

Señal de entrada de 7kHz:

2.9804 2.3137 0.7843 0 0.7843 2.3137 3.0000

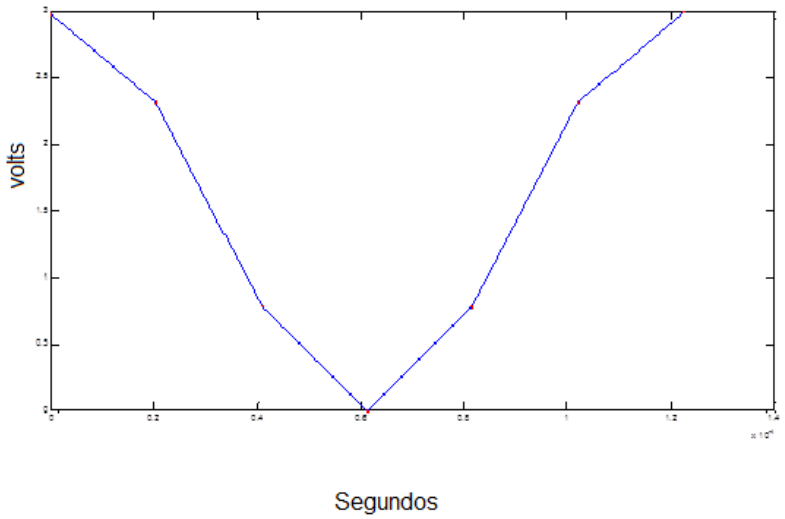


Figura 41 Señal de 7kHz con 7 muestras

En este caso se tienen en promedio 7 elemento en un intervalo de 0.142 ms por lo que cada muestra se toma en promedio cada 20.4 microsegundos.

Señal de entrada de 8kHz:

3.0000 1.9412 0.2941 0.2549 1.8627 3.0000

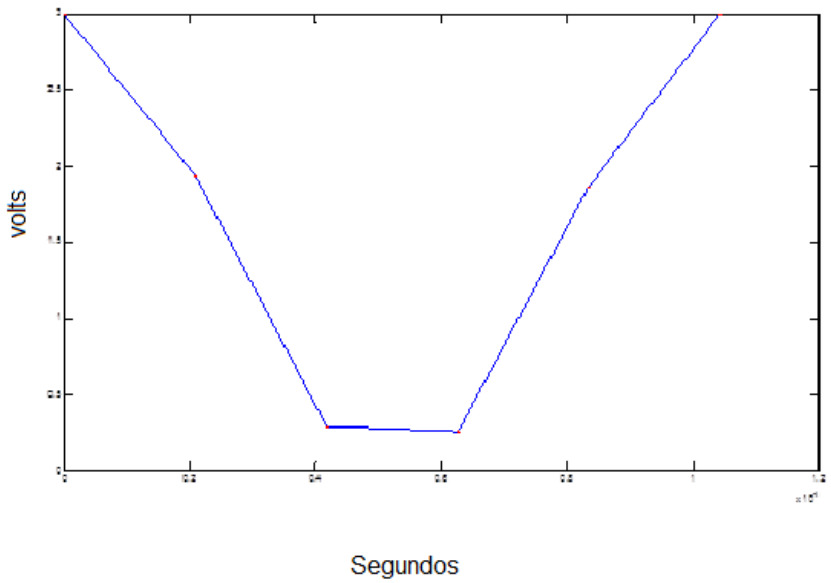


Figura 42 Señal de 8kHz con 6 muestras

En este caso se tienen en promedio 6 elemento en un intervalo de 0.142 ms por lo que cada muestra se toma en promedio cada 20.8 microsegundos.

A partir de estos registros y tomando en cuenta todos los datos guardados durante 15 minutos para 1kHz,2kHz, 3kHz, 4kHz, 5kHz, 6kHz, 7kHz, 8kHz.

Tabla 5 Tiempo de muestreo para señales a diferentes frecuencias.

Frecuencia	Tiempo entre muestras
1kHz	20.0µs
2kHz	25.0 µs
3kHz	21.6 µs

4kHz	22.5 µs
5kHz	24.0 µs
6kHz	23.3 µs
7kHz	23.5 µs
8kHz	23.7 µs

7.3 Procesamiento de datos

La recepción mediante USB se llevó a cabo en una computadora Samsung NP350V4C, se programó para ello una interface desarrollada en Matlab y la señal se introduce mediante un circuito diseñado para este proyecto como placa desarrollo del pic18f2550 que se muestra en la figura 43 .

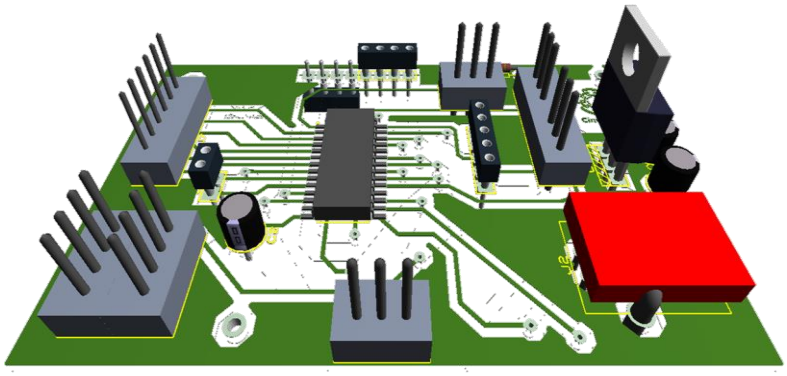


Figura 43 Placa de desarrollo pic18f2550

Se realizaron diez series de adquisiciones de la señal de 1kHz a una frecuencia de muestreo de 40µs a continuación se muestra los datos en volts obtenidos en un ciclo el código de adquisición se agrega en el apéndice.

2.8000 2.7634 2.6170 2.3608 2.0497 1.6654 1.2627 0.8784 0.5307 0.2562 0.0732 0 0.0366 0.2013 0.4575 0.7686 1.1529 1.5556 1.9399 2.2876 2.5621 2.7268 2.8000

El tiempo promedio de adquisición de mediante el puerto USB, fue de1.5µs.

La interface de las pruebas se observa en la figura 44, así como la señal que se introdujo mediante el generador de funciones y monitoreada por un osciloscopio Tektronix TDS2004B.

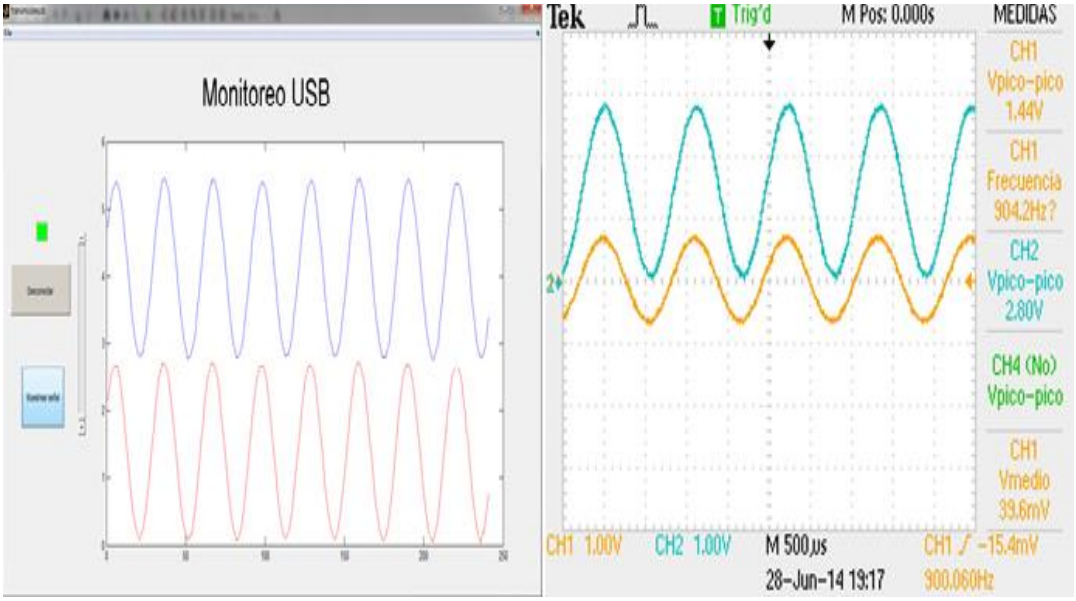


Figura 44 Interfaz para monitoreo de datos mediante USB

7.4 Pruebas de Ripples

La primera señal analizada tiene una duración de 5891 segundos, la figura 45 muestra dicha señal junto con la misma señal pero filtrada.

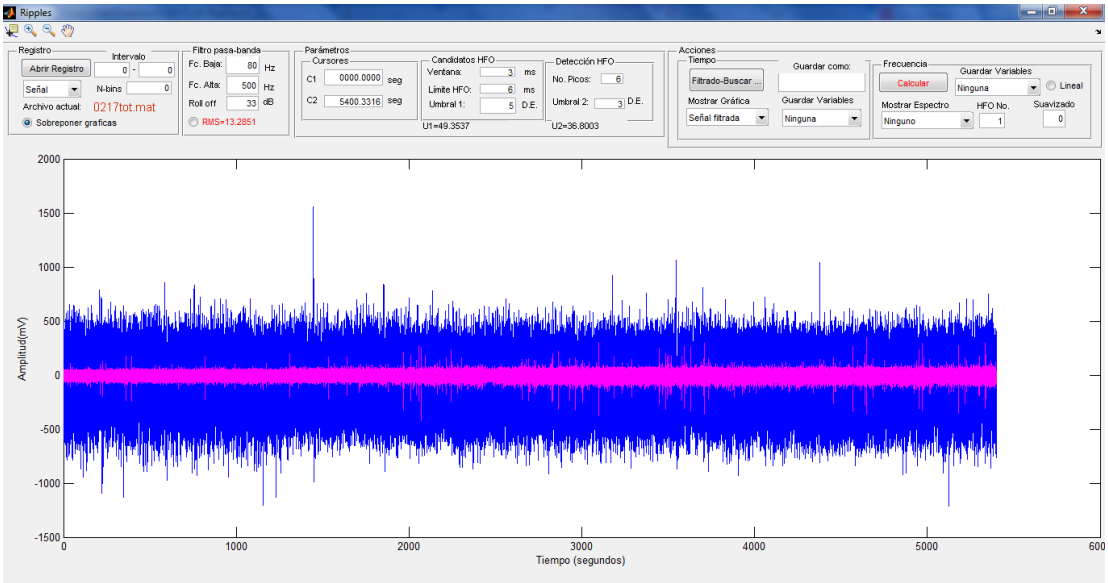


Figura 45 Análisis de la primera Señal (original y filtrada)

El valor rms de la señal filtrada que esta entre 80Hz y 500Hz es de una magnitud de 13.45 mV , se configura un tamaño de ventana de 3ms para la búsqueda de candidatos a ripples, fijando un umbral de cinco desviaciones estándar, por lo menos en seis milisegundos. La figura 46 muestra el sondeo y selección de candidatos para ripples.

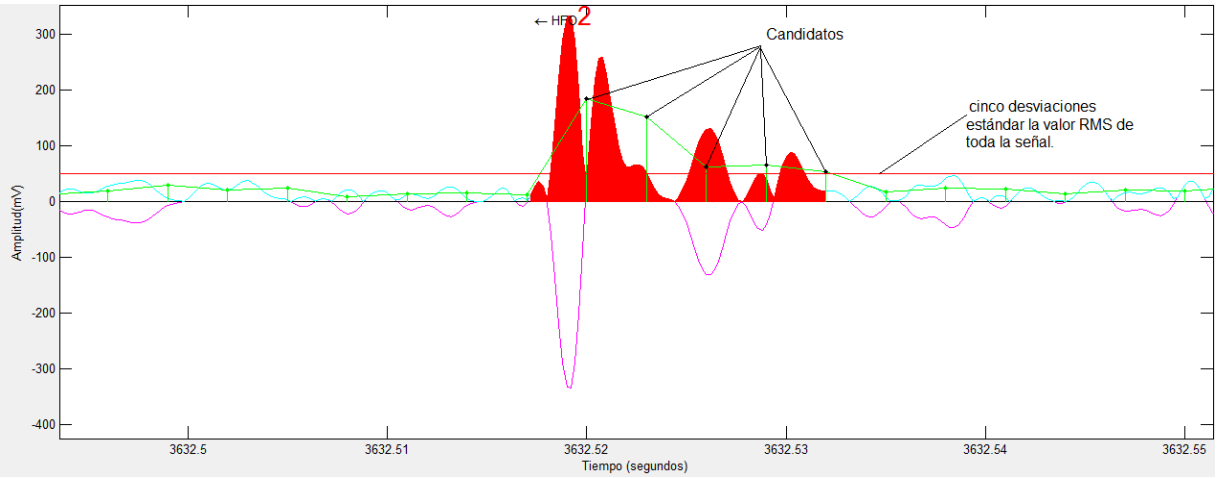


Figura 46 Candidatos a oscilaciones de alta frecuencia

Se define como oscilación la señal de alta frecuencia al establecier como mínimo el conteo a seis picos amplitud superior a tres desviaciones estándar por encima de la media, de la señal filtrada y rectificada dentro de los candidatos previamente definidos. La figura 47 muestra como identifica el programa estas oscilaciones de alta frecuencia, las cuales son marcadas con un índice etiquetado y en color rojo.

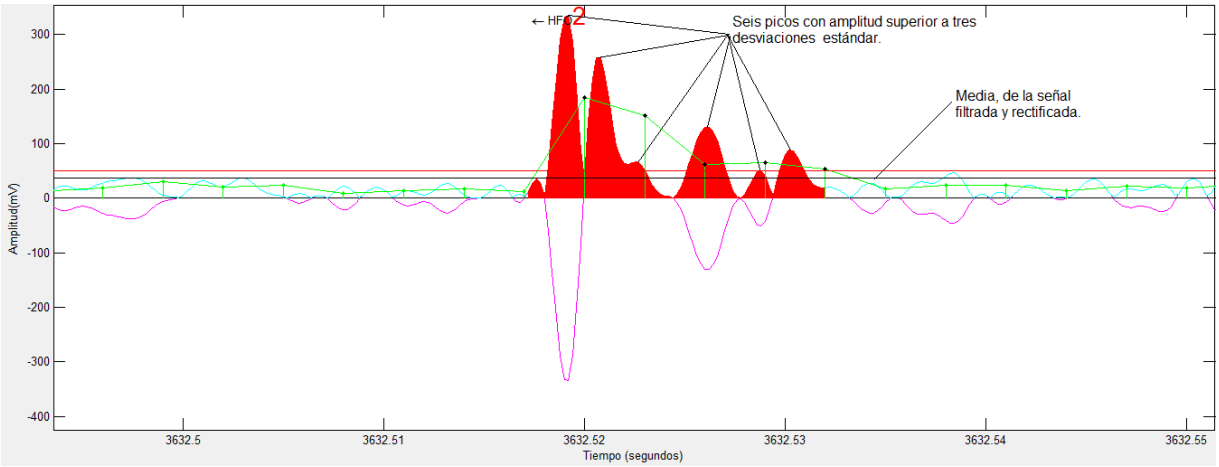


Figura 47 Reconocimiento de oscilaciones de alta frecuencia

Las oscilaciones identificadas se muestran ya ordenadas según su aparición en el tiempo, como se muestra en la figura 48.

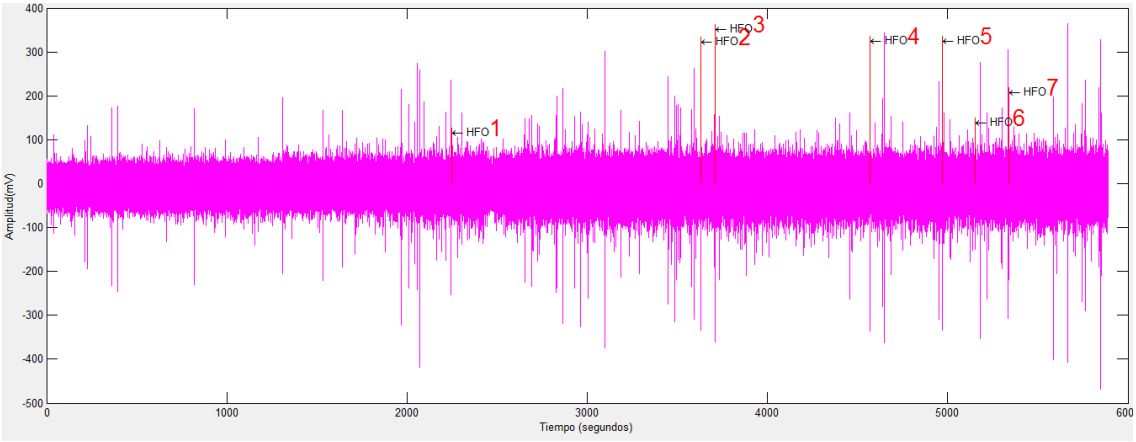


Figura 48 Oscilaciones de alta frecuencia encontradas

Las señales analizadas y una oscilación particular de cada una se muestran en las figuras 49 y 50.

Segunda señal : duración de 5400 segundos, un voltaje rms de 34.54mV.

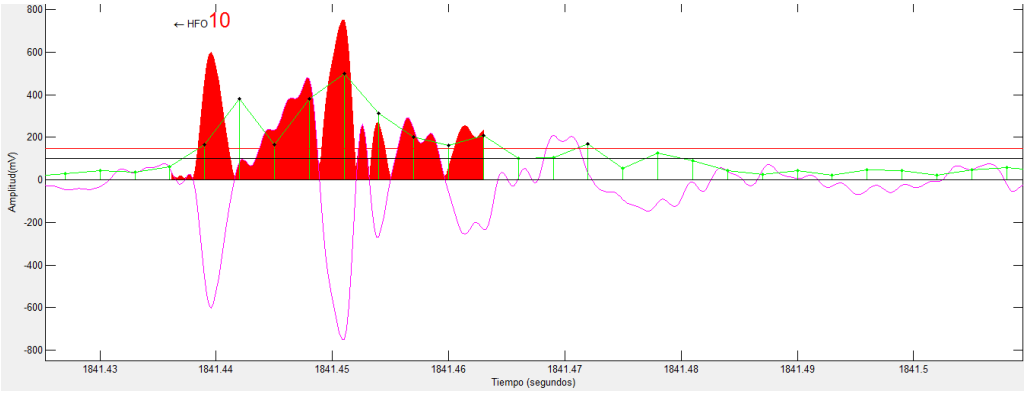


Figura 49 Decima oscilación de alta frecuencia

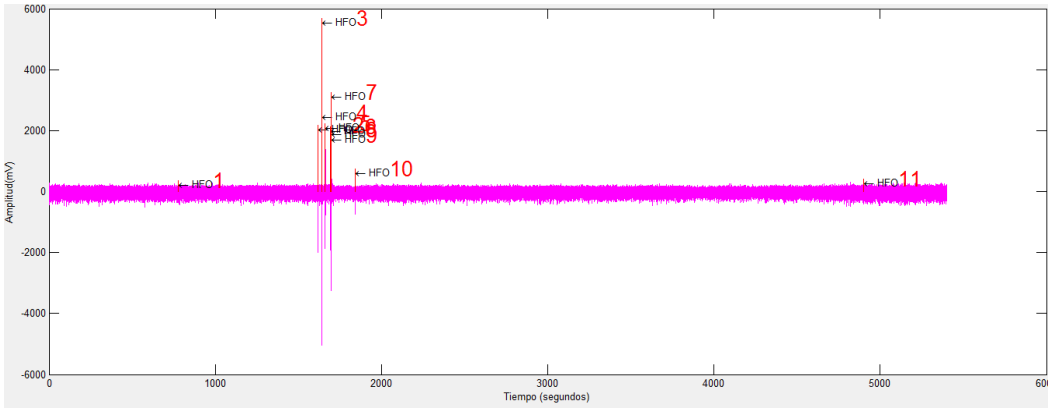


Figura 50 Once oscilaciones de alta frecuencia encontradas

Señal tres : duración de 500 segundos, un voltaje rms de 15.08mV.

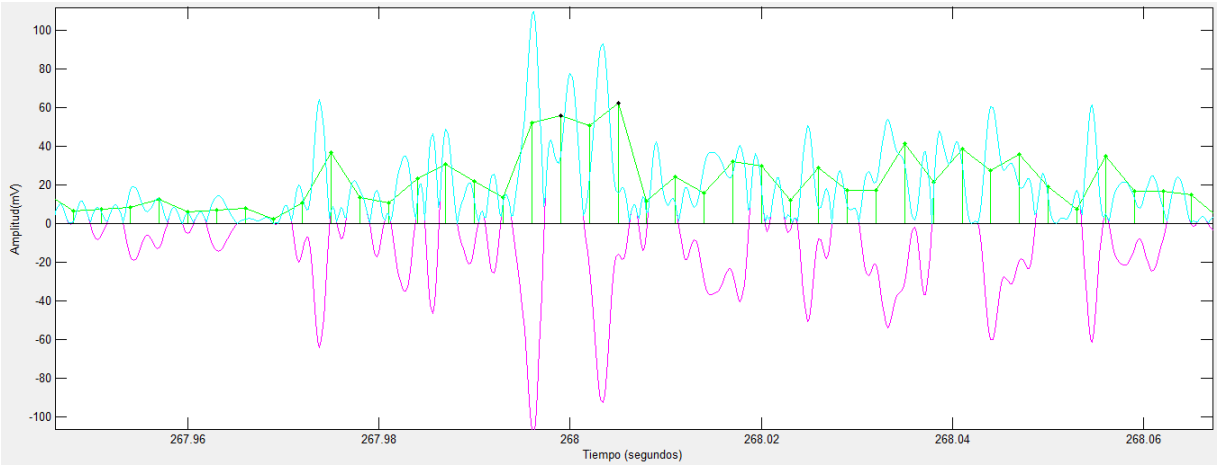


Figura 51 Candidatos con menos de seis picos por encima de la media.

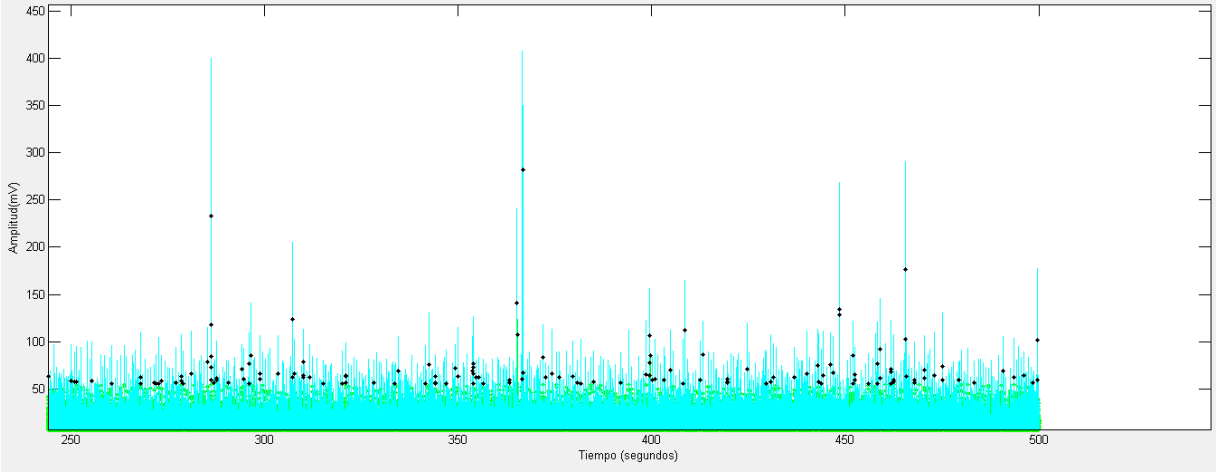


Figura 52 Señal sin oscilaciones identificadas

La segunda prueba se realiza introduciendo una señal senoidal de amplitud 1mV a 5kHz, el cálculo del valor eficaz o valor rms obtuvo 0.7081mV.

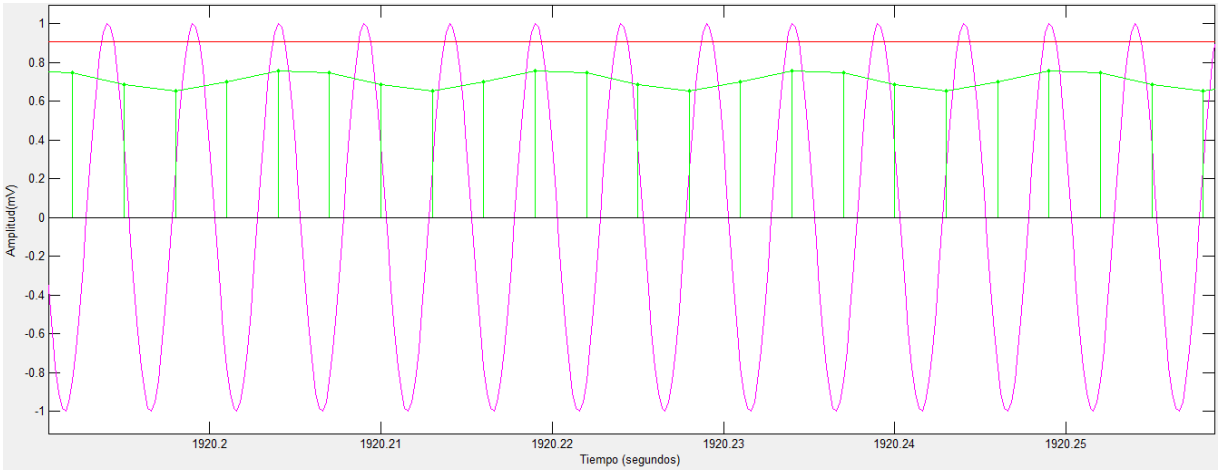


Figura 53 Análisis de señal senoidal a 5kHz

7.5 Pruebas de sincronización

La prueba para identificar una fuente de luz se realiza utilizando una interface que permite seleccionar el video previamente guardado, mediante una barra deslizadora se pueden seleccionar los diferentes frames del video, los cuales se muestran en la imagen de la izquierda en la figura 54, mientras que en la imagen de la derecha se muestra la discriminación del led encendido.

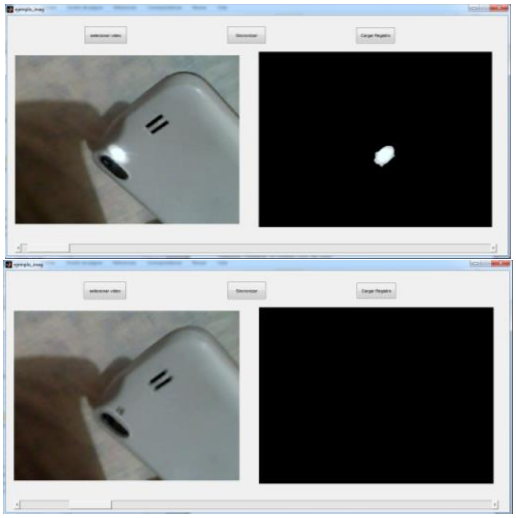


Figura 54 Procesamiento de imagen para identificar una fuente de luz

La segunda prueba se realizó utilizando 10 videos en los se enciende y apaga un led en periodos de tiempo arbitrarios, el programa realiza una gráfica cuyo valor en el eje y tiene el valor de uno en el instante en que se enciende el led y cero cuando se apaga. En la figura 55 se muestran una imagen de video con sus respectivas graficas pulsos luminosos.

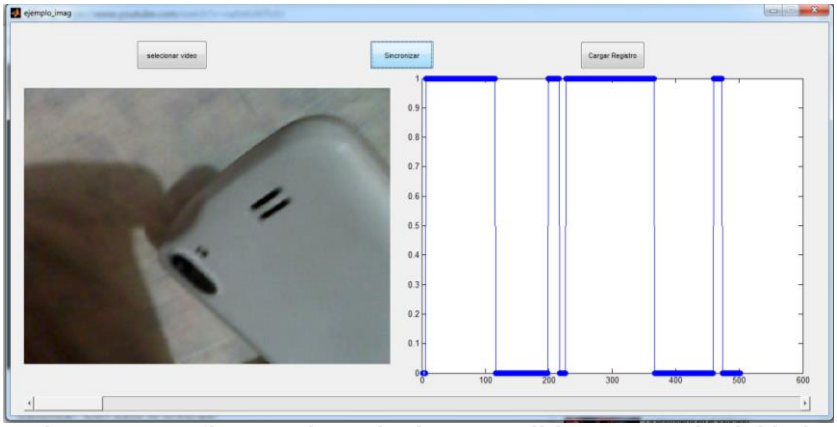


Figura 55 Grafica con la serie de encendido y apagado del led

En la última prueba se grafican conjuntamente los frames de video y la señal obtenida del circuito eléctrico , utilizando la gráfica de encendido y apagado obtenida en la prueba anterior para sincronizarlos con los pulsos contenidos en la señal guardada en la SD. Se muestran en la figura 56 las dos señales simultáneamente, el tiempo en que el led está encendido o apagado es de aproximadamente 1.25seg, tomando 50000 muestras de la señal en cada etapa de encendido o apagado. En la figura 57 se muestra el uso de las herramientas para manipulación de gráficas.

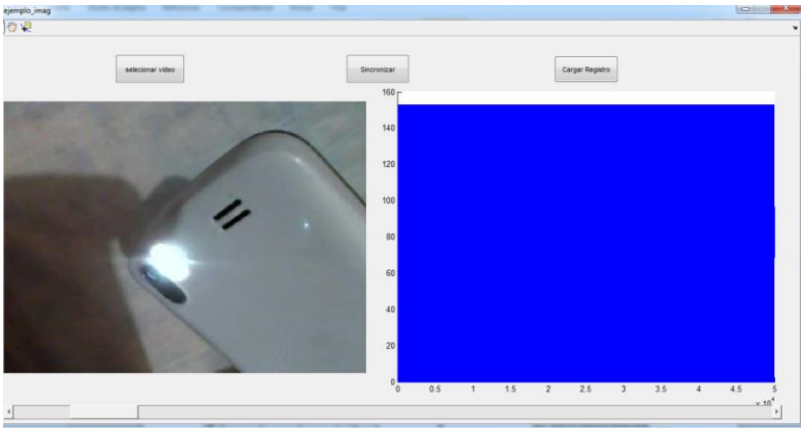


Figura 56 Sincronización de la señal de video y una señal seno de 3khz

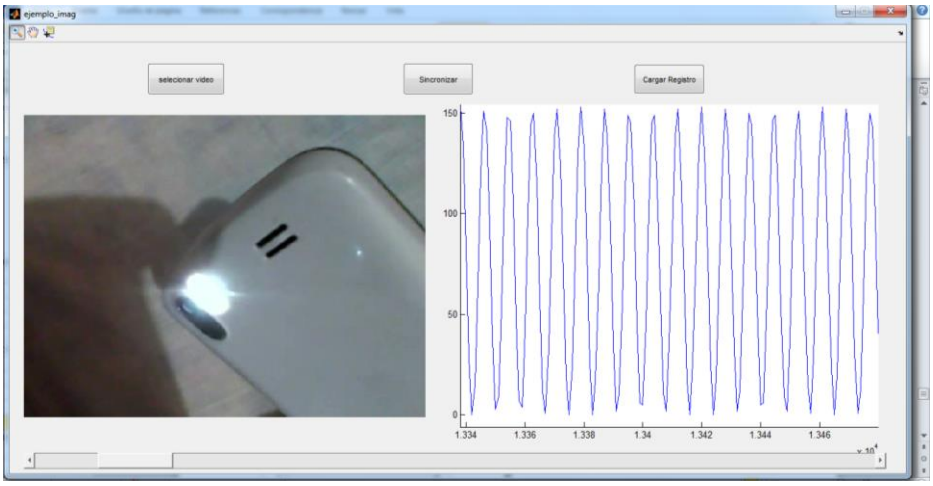


Figura 57 Sincronización de la señal de video, manipulación de las herramientas para la grafica

8 Discusión

8.1 Circuito de adquisición y acondicionamiento de señal

La respuesta en frecuencia del filtro diseñado se muestra en la figura 58 y debajo la respuesta calculada de los datos experimentales.

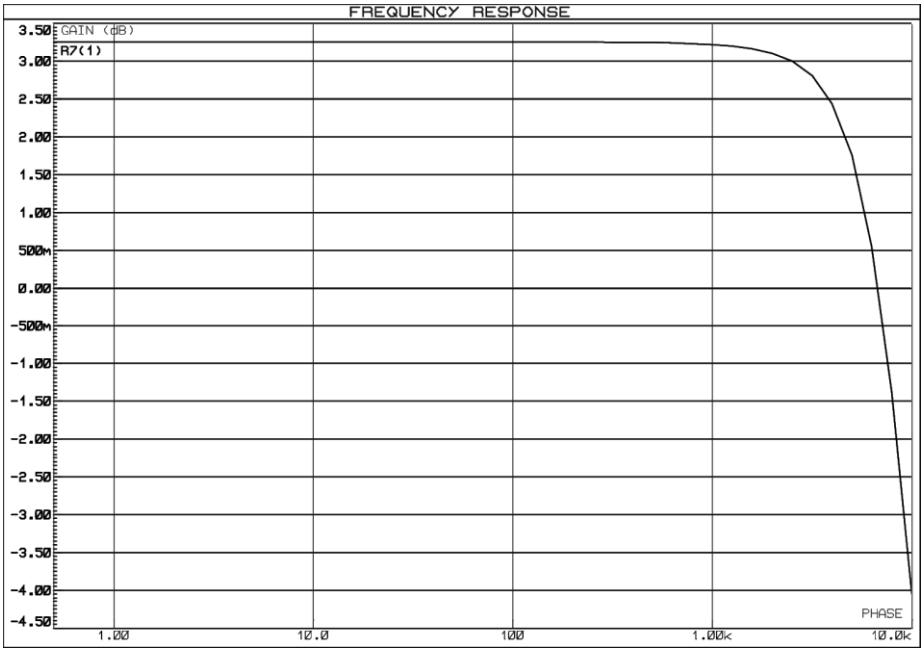


Figura 58 Respuesta en frecuencia del filtro pasa bajas a 7.2 kHz

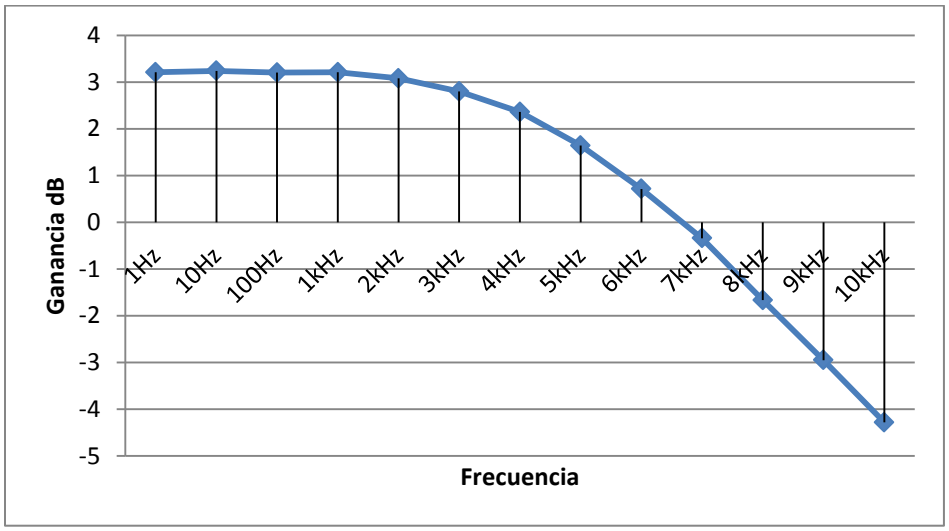


Figura 59 Respuesta experimental del filtro pasa bajas

Según el diseño del filtro, a la frecuencia de corte 7.2kHz debe haber una ganancia de 1.02, y según los datos obtenidos esta ganancia esta aproximadamente 0.93, que permite atenuar las frecuencias menores a 7.2kHz.

8.2 Digitalización y almacenamiento de la señal

Según la tabla 4 y los valores obtenidos para cada prueba de 15 minutos, se obtuvieron los siguientes valores de tiempo de muestreo.

Tabla 6 Tiempos de muestres para distintas señales

Frecuencia de la señal	*Tiempo de muesteo*
Señales tomadas durante cinco minutos	
100Hz	20.8 µs
1kHz	20.4 µs
3kHz	23.57 µs
7kHz	23.81 µs
8kHz	24.3 µs
Señales tomadas durante quince minutos	
1kHz	20.0 µs
2kHz	25.0 µs
3kHz	21.6 µs
4kHz	22.5 µs
5kHz	24.0 µs
6kHz	23.3 µs
7kHz	23.5 µs
8kHz	23.7 µs

Con los valores medidos se obtiene un periodo de muestreo promedio de 22.8µs.

Según el teorema de muestreo para recuperar una señal muestreada sin pérdida considerable de información la frecuencia de muestreo debe ser por lo menos el doble de la máxima frecuencia a capturar. Para nuestro sistema analógico se diseñó un filtro para bajas

que permite el paso hasta 7.2 kHz , por lo que la mínima frecuencia de muestreo es de 14.4 kHz o un intervalo de muestreo de 69.44 us.

A partir de los datos obtenido se obtuvo un intervalo de muestreo promedio de 22.8 µs o bien una frecuencia de 43.859, con la cual pueden obtenerse hasta seis muestras de una señal de 7.2kHz. Es decir que muestra frecuencia de muestreo cubre el ancho de banda propuesto utilizando el filtro pasa bajas o anti alias diseñado en la etapa analógica.

8.3 Procesamiento de datos

Para la adquisición de datos se utilizó el puerto USB en modo Bulk transfer , el cual nos permite un intercambio masivo de datos pero sin prioridad de uso de puerto, según la pruebas realizadas con señales de 1kHz , se obtuvo un tiempo de muestreo promedio de 1.5 µs es decir aproximadamente 660kB/s o 5.3Mb/s, resulta suficiente para una transmisión de los datos guardados en la memoria SD.

Para este trabajo se utiliza la comunicación USB únicamente para la descarga de los archivos dentro del programa de análisis. EL programa de adquisición en tiempo real muestra la capacidad del puerto, que en este caso está limitada por la velocidad del programa.

8.4 Pruebas de ripples

A partir de las pruebas realizadas en el programa de ripples mostrada en el siguiente capítulo, hay que destacar que la frecuencia de muestreo a la que se debe adquirir la señal debe estar por encima de 4kHz para que puedan existir por lo menos seis picos dentro de un candidato a ripple.

Respecto al cálculo del valor de voltaje rms, teóricamente el valor calculado para una señal con amplitud de uno, debe ser de 0.7071 sin embargo el programa realiza un cálculo de 0.708, debido a que el cálculo no se realiza exactamente sobre un número entero de ciclos.

9 Conclusiones y perspectivas

El diseño electrónico permite la captura de señales dentro del ancho de banda de 0.8 a 7.2kHz, mediante un diseño compacto que permite una mayor movilidad a los pequeños roedores durante estudios de laboratorio. El uso de una memoria SD permiten la captura de datos sin necesidad del uso de cables de comunicación y la señal luminosa permite la sincronización de los datos cerebrales con los videos tomados durante el experimento.

Sin embargo si existe la necesidad de observar la señal en tiempo real se han planteado y ya se ha comenzado, el diseño para trabajos futuros en los cuales se plantea una velocidad de comunicación USB mayor es decir por lo menos de 0.4Mb/s, por canal y debido a que el PIC puede enviar datos hasta una velocidad de 12Mb/s en modo Bulk transfer el bus de comunicación resulta óptimo. Por otro lado el programa de adquisición (escrito en la plataforma de Matlab) no tiene prioridad y por lo tanto los tiempos en los que recibe la información no son constantes, debido a esto el micro controlador debe guardar los datos mientras no sean solicitados por el host , si el tiempo de espera es muy grande el bus puede llenarse y perderse información . Mediante el uso de memoria externa 23LC1024 se ha planteado la solución de este problema.

El programa de búsqueda de ripples, resulta una herramienta útil para identificar eventos relacionados con procesos de aprendizaje. Aunque actualmente se utilizan para identificar crisis epilépticas, se cree que existe una relación entre estos ripples con procesos de aprendizaje. Debido a esto, se desarrolló este programa con algunas funciones importantes, como cálculo de las frecuencias que existen en la señal, histogramas y distribución de datos, entre otros que le den al investigador herramientas que faciliten su trabajo de investigación.

10 Apéndices

10.1 Programas

Sincronización

```
function varargout = ejemplo_imag(varargin)
% EJEMPLO_IMAG M-file for ejemplo_imag.fig
%      EJEMPLO_IMAG, by itself, creates a new EJEMPLO_IMAG or raises the existing
%      singleton*.
%
%      H = EJEMPLO_IMAG returns the handle to a new EJEMPLO_IMAG or the handle to
%      the existing singleton*.
%
%      EJEMPLO_IMAG('CALLBACK',hObject,eventData,handles,...) calls the local
%      function named CALLBACK in EJEMPLO_IMAG.M with the given input arguments.
%
%      EJEMPLO_IMAG('Property','Value',...) creates a new EJEMPLO_IMAG or raises
the
%      existing singleton*.  Starting from the left, property value pairs are
%      applied to the GUI before ejemplo_imag_OpeningFcn gets called.  An
%      unrecognized property name or invalid value makes property application
%      stop.  All inputs are passed to ejemplo_imag_OpeningFcn via varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help ejemplo_imag

% Last Modified by GUIDE v2.5 19-Nov-2015 02:33:53

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @ejemplo_imag_OpeningFcn, ...
                   'gui_OutputFcn',  @ejemplo_imag_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before ejemplo_imag is made visible.
```

```
function ejemplo_imag_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to ejemplo_imag (see VARARGIN)
global IMAG i grafica1 grafica2 barra valor
i=1;

handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes ejemplo_imag wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = ejemplo_imag_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;


% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global IMAG grafica1
%[Nombre,Direccion] = uigetfile({'*.tif;*.bmp;*.gif;*.jpg','All Image
Files'});%abrir archivo
% if ~isequal(Nombre, 0)
%     IMAG =  imread(strcat(Direccion,Nombre));
%     axes(grafica1);
%    imshow(IMAG)
% end
[Nombre,Direccion] = uigetfile({'*.avi;*.mpg;*.wmv;*.mp4','All Video
Files'});%abrir archivo
if ~isequal(Nombre, 0)
    IMAG = mmreader(strcat(Direccion,Nombre));

end




% --- Executes on button press in pushbutton2.
```

```
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global IMAG grafica2 grafica1
IMAG.FrameRate
IMAG.Duration
L=fix(IMAG.Duration*IMAG.FrameRate)

recon=zeros(1,L);
for i=1:L
IM=read(IMAG,i);
he = flipdim(IM,2);
hep=rgb2hsv(he);
a=uint8( (hep(:,:,2)<= 0.3)&(hep(:,:,3)>= 0.75)&(((( (hep(:,:,1)>= 0.125 ))))))); %
rf-0.0556..rs-0.1389   ((hep(:,:,1)<= 0.0417) | (hep(:,:,1)>= 0.9722))&
(hep(:,:,1)>= 0.2) & (hep(:,:,1)<= 0.4)  (hep(:,:,1)>= 0.29) & (hep(:,:,1)<=
0.3780)
a = bwareaopen(a,35);
 se = strel('disk',20);
 a = imclose(a,se);
a = imfill(a,'holes');
a=uint8(a);

if (sum(sum(a))~=0)
recon(i)=1;
else
recon(i)=0;
end
he(:,:,1)=he(:,:,1).*a;he(:,:,2)=he(:,:,2).*a;he(:,:,3)=he(:,:,3).*a;
axes(grafica2);
imshow(he)
axes(grafica1);
imshow(read(IMAG,i))

end
axes(grafica2);
plot(recon,'o')
hold on
plot(recon)
hold off




% s(i,1).imagen=IMAG;
% imshow(s(i,1).imagen)
% i=i+1;


% --- Executes during object creation, after setting all properties.
function axes1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to axes1 (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
global grafica1
% Hint: place code in OpeningFcn to populate axes1
grafica1=hObject;


% --- Executes during object creation, after setting all properties.
function axes2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to axes2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
global grafica2
% Hint: place code in OpeningFcn to populate axes1
grafica2=hObject;

% Hint: place code in OpeningFcn to populate axes2


% --- Executes on button press in pushbutton4.
function pushbutton4_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global IMAG grafica1 valor
velocidad=IMAG.FrameRate;
L=IMAG.NumberOfFrames;
tiempo=1/velocidad;

[fid,msg] = fopen('5khz.txt');
A=textscan(fid,'%c');
A=cell2mat(A);
L=length(A);i=1;j=1;valor=[];
hold on
while(i<=L)
    A(i:i+1);
    valor=[valor,hex2dec(strcat(A(i),A(i+1)))];
    i=i+2;
end
valor=valor(1:50000);
% axes(grafica1);
% for i=1:L
%     imshow(read(IMAG,i))
%     pause(tiempo);
% end


% --- Executes on slider movement.
function slider1_Callback(hObject, eventdata, handles)
% hObject    handle to slider1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global barra IMAG grafica2 grafica1 valor
% Hints: get(hObject,'Value') returns position of slider
```

```
%        get(hObject,'Min') and get(hObject,'Max') to determine range of slider
barra=hObject;
IMAG.FrameRate
IMAG.Duration
get(barra,'Value')
L=fix(IMAG.Duration*IMAG.FrameRate);
recon=zeros(1,L);
i=fix(get(barra,'Value')*L)
IM=read(IMAG,i);
he = flipdim(IM,2);
hep=rgb2hsv(he);
a=uint8( (hep(:,:,2)<= 0.3)&(hep(:,:,3)>= 0.75)&(((( (hep(:,:,1)>= 0.125 ))))));
rf-0.0556..rs-0.1389   ((hep(:,:,1)<= 0.0417) | (hep(:,:,1)>= 0.9722))&
(hep(:,:,1)>= 0.2) & (hep(:,:,1)<= 0.4)  (hep(:,:,1)>= 0.29) & (hep(:,:,1)<=
0.3780)
a = bwareaopen(a,35);
 se = strel('disk',20);
 a = imclose(a,se);
a = imfill(a,'holes');
a=uint8(a);

if (sum(sum(a))~=0)
recon(i)=1;
else
recon(i)=0;
end
he(:,:,1)=he(:,:,1).*a;he(:,:,2)=he(:,:,2).*a;he(:,:,3)=he(:,:,3).*a;
axes(grafica2);
plot(valor)
%imshow(he)
axes(grafica1);
imshow(read(IMAG,i))




% --- Executes during object creation, after setting all properties.
function slider1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end
```

COdigo del PIC

```
#include <18F2550.h>
#device adc=8
#fuses HSPLL,NOWDT,NOPROTECT,NOLVP,NODEBUG,USBDIV,PLL5,CPUDIV1,VREGEN
```

```
#use delay(clock=48000000)
#use rs232(baud=115200,parity=N,xmit=PIN_C6,rcv=PIN_A1,bits=8)
//#use rs232(baud=9600,parity=N,xmit=PIN_C6,rcv=PIN_A1,bits=8, force_sw)

#include <SD_tesis_h.c>
#BYTE TRISA  = 0x0F92  //TRISA
#BYTE TRISB  = 0x0F93  //TRISB
#BYTE TRISC  = 0x0F94  //TRISC
#BYTE PORTA  = 0x0F80  //PORTA
#BYTE PORTB  = 0x0F81  //PORTA
#BYTE ADCON0  = 0x0FC2
int32 tmp=0,tmp1=0;
int8 b,bbl;
int16 k=0;
/* ************************************************************************** */
char BufferSD[512]="¡¡¡!!!";//32
char BufferSDr[512]="¡¡¡!!!";
/* ************************************************************************** */

/*int8  Respuesta,*APBUF1,*APBUF2;
int8  i,b=0,bbl=0;
int16 x,tmp=0;
//int8  Caracter="2";                // Carácter a escribir
int32 direc,k=0;           // Número de Dirección para comenzar a leer y escribir
*/
#int_TIMER0
void TIMER0_isr ()
{  int8 dato;

   set_TIMER0(0x10);                    //inicializa el timer0
   read_adc(ADC_START_ONLY);          //Inicio de conversion
   while (bit_test(ADCON0,1));
   dato= read_adc(ADC_READ_ONLY);
   if (dato==255)
   dato=254;
   delay_us(17);
    if(tmp==50000 )
   {tmp=1; output_toggle(PIN_B4);
    dato= 255;//printf("\n\rBytes\n\r x=%Lu direc=%Lu\n\r Tiempo %Lu x 20us\n\r ",x,direc-1,50000);
   }
   if(b==0)
 {BufferSDr[k]=dato;}
 else
 {BufferSD[k]=dato;}

   k++;
  // printf("\n\r %Lu",k);
   tmp++;
   if(k>511 )
   {k=0;bbl=1;
   if(b==0)
 {b=1;}
 else
 {b=0;}
```

```
   }

   clear_interrupt(INT_TIMER0);      // clears the timer0 interrupt flag

}

/* ************************************************************************** */


void main()
{int sdop,j,k,bh,bl,debug;
int16 sector;
int32 i,errores;
i=512;
bit_clear(TRISB,4);
sdop=0;
j=1;
for (i=32;i<512;i++)
{
BufferSD[i]="0";
BufferSDr[i]="0";
j++;
}

   setup_adc_ports(NO_ANALOGS|VSS_VDD|RTCC_8_bit);
   setup_adc_ports(AN0);
   setup_adc( VSS_VDD );
   set_adc_channel(0);
   setup_adc( ADC_CLOCK_div_4 );
   //setup_adc(ADC_OFF);
   //setup_psp(PSP_DISABLED);
   setup_wdt(WDT_OFF);
   SETUP_TIMER_0(RTCC_INTERNAL|RTCC_DIV_1|RTCC_8_bit);
   setup_comparator(NC_NC_NC_NC);
   setup_vref(FALSE);
   bit_set(TRISB,5);


   debug=1;  ////////////////////////////////////////Se cambio a 0 para prueba
  disable_interrupts (INT_TIMER0);
  printf("Iniciando SD Card\r\n");
  while(debug){
if(input (PIN_B5))
{
   if(SDCard_init()==0){
      printf("Error\r\n");

      debug=0;BufferSD="¡¡¡BUFFER !!!";//32
      BufferSDr="¡¡¡Buffer          !!!";i=0;b=0;
      enable_interrupts(GLOBAL);enable_interrupts(INT_TIMER0);set_TIMER0(0x10);
   }else{
      printf("Se ha iniciado correctamente la memoria!!!\r\n\r\n");
      debug=0;BufferSD="¡¡¡BUFFER !!!";//32
      BufferSDr="¡¡¡Buffer         !!!";i=0;b=0;
```

```
    enable_interrupts(GLOBAL);enable_interrupts(INT_TIMER0);set_TIMER0(0x10);
   }
}

  }


  while(!debug)
  {
  if((input (PIN_B5))& i<=3842047 )
  {
  if(bbl)
  {if(b==0)
  {if(!SDCard_write_block(512*i,BufferSD)==0){ i++;bbl=0;}}
  else
  {if(!SDCard_write_block(512*i,BufferSDr)==0){ i++;bbl=0;}}

  }
  }
  else
  disable_interrupts (INT_TIMER0);
  }
}
```

USO VARIOS END POINT EN EL PIC

```
while (TRUE)
   {


     if(usb_enumerated())                    // si el Pic está configurado via USB
     {i=3;j=0;
      while(!usb_kbhit(1)){
           Prender(PIN_B3);
          read_adc(ADC_START_ONLY);          //Inicio de conversion PIN_B6
           while (bit_test(ADCON0,1));
          // delay_us(10);
           entrada = read_adc(ADC_READ_ONLY);
           buffer[i] = entrada;
           buffer[0] = make8(i,0);
            buffer[1] = make8(i,1);
           i++;
           if(i==639)
           {i=3;j++; }
           buffer[2] =j;
     }
     Prender(PIN_B2);
    // buffer[639] =13;
     while(!usb_kbhit(1));
     usb_get_packet(1,dato,10);
     usb_put_packet(1,buffer,64,USB_DTS_TOGGLE);
      while(!usb_kbhit(2));
     usb_get_packet(2,dato,1);
     usb_put_packet(2,buffer+64,64,USB_DTS_TOGGLE);
     while(!usb_kbhit(3));
```

```
      usb_get_packet(3,dato,1);
      usb_put_packet(3,buffer+128,64,USB_DTS_TOGGLE);
      while(!usb_kbhit(4));
      usb_get_packet(4,dato,1);
      usb_put_packet(4,buffer+192,64,USB_DTS_TOGGLE);
      while(!usb_kbhit(5));
      usb_get_packet(5,dato,1);
      usb_put_packet(5,buffer+256,64,USB_DTS_TOGGLE);
      while(!usb_kbhit(6));
      usb_get_packet(6,dato,1);
      usb_put_packet(6,buffer+320,64,USB_DTS_TOGGLE);
      while(!usb_kbhit(7));
      usb_get_packet(7,dato,1);
      usb_put_packet(7,buffer+384,64,USB_DTS_TOGGLE);
      while(!usb_kbhit(8));
      usb_get_packet(8,buffer,1);
      usb_put_packet(8,buffer+448,64,USB_DTS_TOGGLE);
      while(!usb_kbhit(9));
      usb_get_packet(9,dato,1);
      usb_put_packet(9,buffer+512,64,USB_DTS_TOGGLE);
      while(!usb_kbhit(10));
      usb_get_packet(10,dato,1);
      usb_put_packet(10,buffer+576,64,USB_DTS_TOGGLE);
       Apagar(PIN_B2);
       }
    }
}
```

Descriptores de usb

```
#DEFINE USB_TOTAL_CONFIG_LEN      32 //config+interface+class+endpoint

   //configuration descriptor
   char const USB_CONFIG_DESC[] = {
   //config_descriptor for config index 1
         USB_DESC_CONFIG_LEN,     //length of descriptor size
         USB_DESC_CONFIG_TYPE,         //constant CONFIGURATION (0x02)
         USB_TOTAL_CONFIG_LEN,0,  //size of all data returned for this config
         1,      //number of interfaces this device supports
         0x01,               //identifier for this configuration.  (IF we had more than one configurations)
         0x00,               //index of string descriptor for this configuration
        #if USB_CONFIG_BUS_POWER
         0x80, //bit 6=1 if self powered, bit 5=1 if supports remote wakeup (we don't), bits 0-4 unused and bit7=1
==7
        #else
         0xC0, //bit 6=1 if self powered, bit 5=1 if supports remote wakeup (we don't), bits 0-4 unused and bit7=1
==7
        #endif
         USB_CONFIG_BUS_POWER/2, //maximum bus power required (maximum milliamperes/2)  (0x32 =
100mA)   ==8

   //interface descriptor 0 alt 0
         USB_DESC_INTERFACE_LEN,  //length of descriptor
         USB_DESC_INTERFACE_TYPE,      //constant INTERFACE (0x04)
         0x00,               //number defining this interface (IF we had more than one interface)
```

```
        0x00,               //alternate setting
        2,       //number of endpoints, not counting endpoint 0.
        0xFF,               //class code, FF = vendor defined
        0xFF,               //subclass code, FF = vendor
        0xFF,               //protocol code, FF = vendor
        0x00,               //index of string descriptor for interface

    //endpoint descriptor
        USB_DESC_ENDPOINT_LEN, //length of descriptor
        USB_DESC_ENDPOINT_TYPE,     //constant ENDPOINT (0x05)
        0x81,               //endpoint number and direction (0x81 = EP1 IN)
        USB_EP1_TX_ENABLE,              //transfer type supported (0 is control, 1 is iso, 2 is bulk, 3 is interrupt)
        USB_EP1_TX_SIZE & 0xFF,USB_EP1_TX_SIZE >> 8,         //maximum packet size supported
        0x01,               //polling interval in ms. (for interrupt transfers ONLY)

    //endpoint descriptor
        USB_DESC_ENDPOINT_LEN, //length of descriptor
        USB_DESC_ENDPOINT_TYPE,     //constant ENDPOINT (0x05)
        0x01,               //endpoint number and direction (0x01 = EP1 OUT)
        USB_EP1_RX_ENABLE,              //transfer type supported (0 is control, 1 is iso, 2 is bulk, 3 is interrupt)
        USB_EP1_RX_SIZE & 0xFF,USB_EP1_RX_SIZE >> 8,         //maximum packet size supported
        0x01,               //polling interval in ms. (for interrupt transfers ONLY)
   };

    //****** BEGIN CONFIG DESCRIPTOR LOOKUP TABLES ********
    //since we can't make pointers to constants in certain pic16s, this is an offset table to find
    //  a specific descriptor in the above table.

    //NOTE: DO TO A LIMITATION OF THE CCS CODE, ALL HID INTERFACES MUST START AT 0 AND BE
SEQUENTIAL
    //      FOR EXAMPLE, IF YOU HAVE 2 HID INTERFACES THEY MUST BE INTERFACE 0 AND INTERFACE
1
    #define USB_NUM_HID_INTERFACES   0

    //the maximum number of interfaces seen on any config
    //for example, if config 1 has 1 interface and config 2 has 2 interfaces you must define this as 2
    #define USB_MAX_NUM_INTERFACES   1

    //define how many interfaces there are per config.  [0] is the first config, etc.
    const char USB_NUM_INTERFACES[USB_NUM_CONFIGURATIONS]={1};

    #if (sizeof(USB_CONFIG_DESC) != USB_TOTAL_CONFIG_LEN)
      #error USB_TOTAL_CONFIG_LEN not defined correctly
    #endif


////////////////////////////////////////////////////////////////////
///
///   start device descriptors
///
////////////////////////////////////////////////////////////////////

    //device descriptor
    char const USB_DEVICE_DESC[] ={
        USB_DESC_DEVICE_LEN,          //the length of this report
```

```
        0x01,           //constant DEVICE (0x01)
        0x10,0x01,        //usb version in bcd
        0x00,           //class code (if 0, interface defines class.  FF is vendor defined)
        0x00,           //subclass code
        0x00,           //protocol code
        USB_MAX_EP0_PACKET_LENGTH,   //max packet size for endpoint 0. (SLOW SPEED SPECIFIES 8)
        0xD8,0x04,          //vendor id (0x04D8 is Microchip)
        0x0D,0x00,          //product id
        0x01,0x00,          //device release number
        //USB_CONFIG_VID & 0xFF, ((USB_CONFIG_VID >> 8) & 0xFF), //vendor id       ==9, 10
        //USB_CONFIG_PID & 0xFF, ((USB_CONFIG_PID >> 8) & 0xFF), //product id, don't use 0xffff       ==11,
12
        //USB_CONFIG_VERSION & 0xFF, ((USB_CONFIG_VERSION >> 8) & 0xFF), //device release number
==13,14
        0x01,           //index of string description of manufacturer. therefore we point to string_1 array (see
below)
        0x02,           //index of string descriptor of the product
        0x00,           //index of string descriptor of serial number
        USB_NUM_CONFIGURATIONS   //number of possible configurations
    };


////////////////////////////////////////////////////////////////////
///
///   start string descriptors
///   String 0 is a special language string, and must be defined.  People in U.S.A. can leave this alone.
///
///   You must define the length else get_next_string_character() will not see the string
///   Current code only supports 10 strings (0 thru 9)
///
////////////////////////////////////////////////////////////////////

//the offset of the starting location of each string.
//offset[0] is the start of string 0, offset[1] is the start of string 1, etc.
const char USB_STRING_DESC_OFFSET[]={0,4,12};

#define USB_STRING_DESC_COUNT sizeof(USB_STRING_DESC_OFFSET)

// Here is where the "CCS" Manufacturer string and "CCS Bulk Demo" are stored.
// Strings are saved as unicode.
// These strings are mostly only displayed during the add hardware wizard.
// Once the operating system drivers have been installed it will usually display
// the name from the drivers .INF.
char const USB_STRING_DESC[]={
    //string 0
        4, //length of string index
        USB_DESC_STRING_TYPE, //descriptor type 0x03 (STRING)
        0x09,0x04,   //Microsoft Defined for US-English
    //string 1
        8, //length of string index
        USB_DESC_STRING_TYPE, //descriptor type 0x03 (STRING)
        'C',0,
        'C',0,
        'S',0,
    //string 2
```

```
    20, //length of string index
    USB_DESC_STRING_TYPE, //descriptor type 0x03 (STRING)
    'C',0,
    'i',0,
    'n',0,
    'v',0,
    'e',0,
    's',0,
    't',0,
    'a',0,
    'v',0

};


#ENDIF
```

RIPPLES

```
global AVen V  Vrms t desv Udes VLC RMSt LHFO C1 C2 Upicos picos fm ...
    b vf t  VPH tbhfo Fca Fcb Ord  hf ff fc memoria dircre umb1 umb2
cla;
tbhfo=hObject;
 set(hObject,'ForegroundColor',[0 1 0])
 pause(1);
 tic
 s=memory;
memoria=[memoria,(s.MemAvailableAllArrays)/1000000];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Calcular Filtro%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if isstruct(V)
Rp=1;
Rs=str2double(get(Ord,'String'));
fsamp =1/getfield(V,'interval');
fcuts = [Fcb-50 Fcb Fca Fca+50];
mags = [0 1 0];
devs = [10^((-Rs/20)) 1-10^(-(Rp/20)) 10^((-Rs/20))];
[n,Wn,beta,ftype] = kaiserord(fcuts,mags,devs,fsamp);
n = n + rem(n,2);
b = fir1(n,Wn,ftype,kaiser(n+1,beta),'noscale');

%b = fir1(Ord,[(Fcb/(fsamp/2)) (Fca/(fsamp/2))]);
if get(fc,'Value')==1
 dguar=strcat(dircre,'\foldertemp','\valores.mat');
 va=open(dguar);va=va.v;
 xx=va;
 xxf=filter(b,1,xx);
 clear xx
 clear va
 RMSt=(sum(xxf.^2)/length(xxf))^(1/2)
 clear xxf
end

end
s=memory;
memoria=[memoria,(s.MemAvailableAllArrays)/1000000];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%filtrado%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

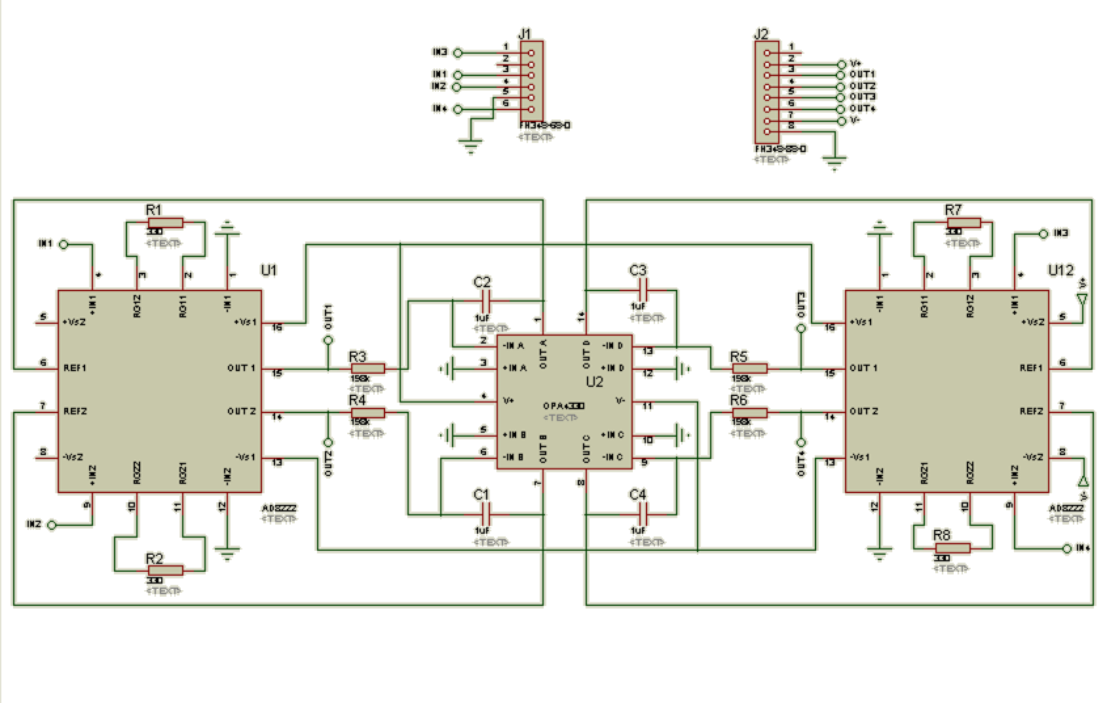
```
if C1==0
    C1=1;
end
if isstruct(V)
dguar=strcat(dircre,'\foldertemp','\valores.mat');
v=open(dguar);v=v.v;
v=v(C1:C2);
t=(C1:C2)*getfield(V,'interval');
t=t';
vf=filter(b,1,v);
%hist(vf,-1500:1500,'FaceColor','r')
clear v
v=0;
dguar=strcat(dircre,'\foldertemp','\valoresf');
save(dguar,'vf')
if get(fc,'Value')==0
 RMSt=(sum(vf.^2)/length(vf))^(1/2)
end
end
dguar=strcat(dircre,'\foldertemp','\valorest');
save(dguar,'t')
t=0;
clear t
set(fc,'String',strcat('RMS=',num2str(RMSt)))
pause(0.5)
s=memory;
memoria=[memoria,(s.MemAvailableAllArrays)/1000000];
%%%%%%%%%%%%%%%%%%%%%%%ENCONTRAR HFO%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
set(hObject,'ForegroundColor',[1 0 0 ])
if isstruct(V)
    tv=fix((str2double(get(AVen,'String'))*0.001)/getfield(V,'interval'));
    if tv==0
        tv=1;
    end
    if ~isempty(vf)
    Vrms(ceil(length(vf)/tv))=0;desv=0;
    end
    u=str2double(get(Udes,'String'));
    s=memory;
memoria=[memoria,(s.MemAvailableAllArrays)/1000000];
if ~isempty(vf)
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%VENTANEO%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    for i=1:length(Vrms)
     if (i*tv)< length(vf)
     Vrms(i)=((sum(vf((i*tv)-(tv-1):(i*tv)).^2))/length(vf((i*tv)-(tv-
1):(i*tv))))^(1/2);
     %Vrms=[Vrms;zeros(length(vf(i:(i+(tv-1)))),1)+((sum(vf(i:(i+(tv-
1)))).^2)/length(vf(i:(i+(tv-1))))))^(1/2)];
     %desv=[desv;((sum(vf(i:(i+(tv-1)))).^2)/length(vf(i:(i+(tv-1))))))^(1/2)];
     else
         if i<length(vf)
     Vrms(i)=((sum(vf((i*tv)-(tv-1):length(vf)).^2))/length(vf((i*tv)-(tv-
1):length(vf))))^(1/2);

%Vrms=[Vrms;zeros(length(vf(i:length(vf))),1)+(sum(vf(i:length(vf)).^2)/length(vf(i
:length(vf))))^(1/2)];
```

```
      %desv=[desv;(sum(vf(i:length(vf)).^2)/length(vf(i:length(vf))))^(1/2)];

          end
      end
     end
      desv=std(Vrms);
dguar=strcat(dircre,'\foldertemp','\valvrms');
save(dguar,'Vrms')
     s=memory;
memoria=[memoria,(s.MemAvailableAllArrays)/1000000];
      %%%%%%%%%%%%%%%%%%%%%%SELECCION DE CANDIDATOS HOF (primer
filtro)%%%%%%%%%%%%%%%%%%%%%%%%
      set(umb1,'String',strcat('U1=',num2str((RMSt+(desv*u)))))
      VLC=Vrms>(RMSt+(desv*u));VLC=double(VLC);
      primero=0;
      clear vf
      clear Vrms
      vf=0;Vrms=0;
      while (length(primero))~= 0
         primero=find(VLC(primero+1:length(VLC))==1,1)+primero;
         ultimo=find(VLC(primero:length(VLC))==0,1)+primero-1;
         tamano=(ultimo-primero)+1;
         if
tamano>=(ceil(fix((str2double(get(LHFO,'String'))*0.001)/getfield(V,'interval'))/tv
));
             VLC(primero:ultimo-1)=2;
         end
         primero=ultimo;
      end
      dguar=strcat(dircre,'\foldertemp','\valVLC');
save(dguar,'VLC')
  s=memory;
memoria=[memoria,(s.MemAvailableAllArrays)/1000000];
      %%%%%%%%%%%%%%%%%%%%%%SELECCION DE CANDIDATOS HOF (segundo
filtro)%%%%%%%%%%%%%%%%%%%%%%%%
      dguar=strcat(dircre,'\foldertemp','\valoresf.mat');
      vf=open(dguar);vf=vf.vf;

VPH=[];vfr=abs(vf);medvrf=mean(vfr);devrf=std(vfr);tv=fix((str2double(get(AVen,'Str
ing'))*0.001)/getfield(V,'interval'));
      VLC=double(VLC);npikos=0;
       primero=0;
       clear vf
       vf=0;

set(umb2,'String',strcat('U2=',num2str(medvrf+(devrf*str2double(get(Upicos,'String'
))))))
      while (length(primero))~= 0
         primero=find(VLC(primero+1:length(VLC))==2,1)+primero;
         ultimo=find(VLC(primero:length(VLC))~=2,1)+primero-2;
         if (length(primero))~= 0
             warning('off','signal:findpeaks:noPeaks')
         npikos=length(findpeaks(vfr((primero*tv)-(tv-
1):(ultimo*tv)),'minpeakheight',medvrf+(devrf*str2double(get(Upicos,'String')))));
         end
         if npikos>=str2double(get(picos,'String'));
```

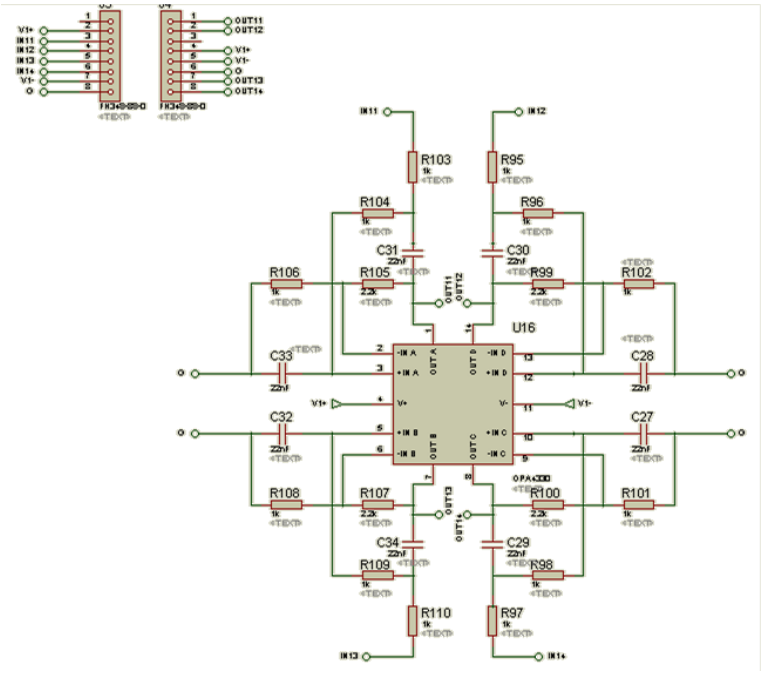
```
            VLC(primero:ultimo-1)=3;
            VPH=[VPH;[(primero*tv)-(tv-1),(ultimo*tv)]];
        end
        primero=ultimo;
    end
      clear VLC
      clear vfr
      VLC=0;vfr=0;
     dguar=strcat(dircre,'\foldertemp','\valVPH');
save(dguar,'VPH')
 VPH=0;vf=0;
  s=memory;
memoria=[memoria,(s.MemAvailableAllArrays)/1000000] ;
end
end
set(hObject,'ForegroundColor',[0 0 0])
toc
```

10.2 Esquemas

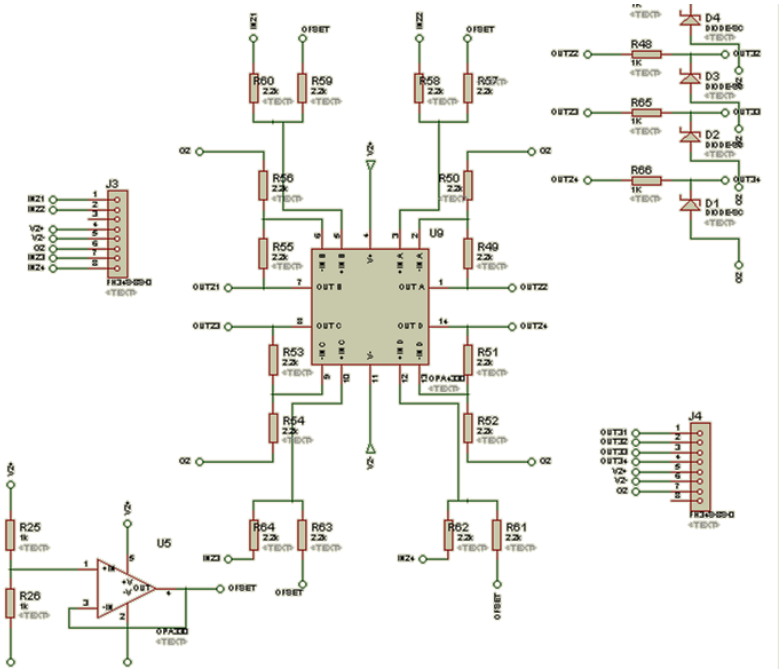
Circuito de preamplificador



Circuito de filtrado



Circuito de acoplamiento



11 Referencias

[1] Kleanthis C. Neokleous, Marios N. Avraamides,Costas K. Neocleous, and Christos N. Schizas. Cognitive Modeling of Dilution Effects in Visual Search. Department of Psychology, University of Cyprus, Department of Mechanical Engineering, Cyprus University of Technology, Department of Computer Science, University of Cyprus.2012

[2] Wajid Mumtaz1, Likun Xia, Aamir Saeed Malik, and Mohd Azhar Mohd Yasin, Complexity Analysis of EEG Data during Rest Stateand Visual Stimulus, Electrical and Electronic Engineering, Universiti Teknologi PETRONAS, 31750 Tronoh, Perak, Malaysia,Department of psychiatry, Hospital Universiti Sains Malaysia, Kelantan, Malaysia.2012.

[3] Ahmed Al-Ani, Bram Van Dun, Harvey Dillon, and Alaleh Rabie1. Analysis of Alertness Statusof Subjects Undergoing the Cortical Auditory Evoked Potential Hearing Test. Faculty of Engineering and Information Technology, University of Technology, Sydney, Ultimo NSW 2007 Australia,National Acoustic Laboratories, Chatswood, Sydney NSW 2067 Australia. 2012.

[4] Takahiro Imai, Takanori Sato, Isao Nambu, and Yasuhiro Wada. Estimating Brain Activity of Motor Learning by Using fNIRS-GLM Analysis. Nagaoka University of Technology, Nagaoka, Japan.2012

[5] Cameron S. Carter • Jeffrey W. Dalley, Brain Imaging in Behavioral Neuroscience, Springer, Current Topics in Behavioral NeurosciencesVolume 11 .2012

[6] Xiaoxu Kang, Marc Schieber,and Nitish V. Thakor. Decoding Cognitive States from Neural Activities of Somatosensory Cortex**.** Department of Biomedical Engineering, Johns Hopkins University, Baltimore, MD, Department of Neurology, Cognitive Behavioral Neurology, University of Rochester, Rochester, NY.2012

[7] Triangle BioSystems International. TBSI W5 System Manual Version 3.0 [pdf]. Volume(issue). http://www.trianglebiosystems.com/assets/tbsi_w5_manual.pdf

[8] http://microrespuestas.com/partes-de-una-neurona

[9] **Neuroscience and Behavior** - McGraw-Hill, **Understanding Psychology, 8/e** Capitulo 3

[10] **Neuroscience and Behavior** - McGraw-Hill, **Understanding Psychology, 8/e** Capitulo 3

[11] http://www.unizar.es/departamentos/bioquimica_biologia/docencia/ELFISICABIOL/PM/PotMemFB.htm

[12] http://vitae.ucv.ve/?module=articulo&rv=108&n=1528&m=3&e=1611

[13] Thompson, Richard F. Fundamentos de psicología fisiológica.México :Trillas,1973 (reimp 2009), 15p.

[14] Thompson, Richard F. Fundamentos de psicología fisiológica.México :Trillas,1973 (reimp 2009), 585p.

[15] Thompson, Richard F. Fundamentos de psicología fisiológica.México :Trillas,1973 (reimp 2009), 590p.

[16] Thompson, Richard F. Fundamentos de psicología fisiológica.México :Trillas,1973 (reimp 2009), 606p.

[17] *Paul Horowitz and Winfield Hill (1989), The Art of Electronics (Second ed.), Cambridge University Press.*

[18] Microchip. PIC18F2455/2550/4455/4550 Data Sheet [pdf]. http://ww1.microchip.com/downloads/en/devicedoc/39632c.pdf

[19] Custom Computer Services, Inc. 2015, CCS C Compiler Manual PCB / PCM / PCH [pdf] , https://www.ccsinfo.com/downloads/ccs_c_manual.pdf

[20] **Richard J. Staba, Charles L. Wilson, Anatol Bragin, Itzhak Fried and Jerome, Quantitative Analysis of High-Frequency Oscillations (80-500 Hz) Recorded in Human Epileptic Hippocampus and Entorhinal Cortex, *Journal of Neurophysiology,*** 88:1743-1752, 2002.