



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

Facultad de Ingeniería.

Estimulador Acústico para la Evaluación
de Tareas Cognitivas en Sujetos de Prueba.

Tesis Profesional

Para obtener el título de :
INGENIERO ELÉCTRICO-ELECTRÓNICO.

Presenta:

Luis Carlos Hernández Montoya.

Víctor Hugo Rodríguez Espíndola.

Director de Tesis:

Dr. Rogelio Alcántara Silva.



Agradecimientos.

Primero quiero agradecer a toda mi familia, en especial a mis padres y hermanas por brindarme todo el apoyo que necesite a lo largo de esta experiencia en la Facultad de Ingeniería de la UNAM.

Quiero agradecer especialmente al Dr. Rogelio Alcántara Silva que me aconsejó y apoyó sinceramente en la mayor parte de mi carrera.

Y por último a todas las personas que me acompañaron en esta travesía.

Atte: Luis Carlos.

A mis padres, familia y amigos, que me han visto crecer, muchas gracias por su apoyo, consejos, enseñanzas y amor.

Al Dr. Rogelio Alcántara Silva que me adoptó en su laboratorio, brindándome conocimientos.

Atte: Víctor Hugo.

Queremos agradecer al apoyo de todos nuestros sinodales Dr. Francisco Javier García Ugalde, M.I. Sergio Quintana Thierry, Ing. Rafael Flores García y M.I. Ricardo Mota Marzano, por dedicarnos parte de su tiempo para alcanzar nuestra meta.

También agradecer al grupo de psicología Dr. Oscar Zamora y Dr. Vladimir por creer en nosotros, además del apoyo para la realización de este proyecto.

Al proyecto PAPIME No. PE305511.

A Huesos, nuestra colaboradora.

Y por último a nuestra querida *alma mater*,

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

que nos dio las herramientas necesarias para poder desarrollar exitosamente nuestra vida profesional.

1. Resumen.

Los usuarios de los laboratorios de experimentación (LABEX) de la Facultad de Psicología de la Universidad Nacional Autónoma de México (FacPsiUNAM)[26], requieren generar sonidos a diferentes frecuencias pre-establecidas, controladas por medio de un sistema interfaz PCI (SI-PCI MED), para estudios de psicoacústica.

La psicoacústica es el estudio psicológico de la audición, cuyo objetivo principal es determinar las relaciones cuantitativas entre las variables físicas y perceptuales del sonido. Los parámetros físicos del sonido son: amplitud, frecuencia y forma de onda, que están relacionadas con su parámetro subjetivo correspondiente: sonoridad, tono y timbre.

Para poder realizar dichos estudios la FacPsiUNAM ha adquirido estimuladores acústicos por parte de proveedores en este ramo. Debido al costo de dicho instrumental, no se ha podido equipar adecuadamente los laboratorios con estimuladores acústicos.

Estos sistemas tienen un principio de operación semejante al de un sintetizador de frecuencias, puesto que estos aparatos generan señales de uno o diferentes tipos de onda a diferentes frecuencias.

Existen comercialmente sintetizadores de frecuencia que satisfacen las necesidades de la aplicación, sin embargo, puesto que no requerimos un instrumento de alta precisión, decidimos diseñar un circuito de desempeño equiparable.

Este trabajo consiste en diseñar e implementar un dispositivo de bajo costo para la generación de formas de onda sinusoidal, rectangular y triangular de diferentes frecuencias programables y pre-establecidas por el usuario quien las utiliza para la estimulación acústica en sujetos de prueba.

Este nuevo dispositivo cuenta con ventajas con respecto a los dispositivos que actualmente ocupan los laboratorios, debido a que puede generar 3 tipos de ondas diferentes, muestra en pantalla el estado del dispositivo; ya sea en modo de espera o las características de la señal, y por último, es reprogramable con la finalidad de cambiar las frecuencias de las señales a necesidad de los laboratoristas.

Por otra parte, este dispositivo será replicado físicamente 20 veces para su implementación en FacPsiUNAM, haciendo de este, un aporte a la UNAM.

Índice general

1. Resumen.	3
2. Introducción	1
2.1. Antecedentes.	1
2.1.1. Sistema interfaz PCI.	2
2.1.2. Estimulador acústico comercial.	3
2.1.3. Psicoacústica.	3
2.2. Generación de señales.	5
2.3. Configuración de la arquitectura para el generador de ondas propuesto.	5
2.4. Acoplamientos eléctricos del dispositivo.	6
2.5. Estudio de las señales discretizadas por el microcontrolador.	6
2.6. Alcances del proyecto.	6
3. Generación de señales continuas y discretas.	9
3.1. Sintetizadores de Frecuencia.	9
3.1.1. Método de generación.	9
3.1.2. Clasificación.	9
3.2. Muestreo.	10
3.2.1. Uso del tren de impulsos unitarios.	10
3.3. Reconstrucción de la señal usando retén de orden cero.	12
3.3.1. Interpolación lineal.	12
3.3.2. Interpolación de banda limitada.	13
3.4. Conversión de señales Digital-Analógico.	14
3.5. Cambio de tasa de muestreo en señales discretas	15
3.5.1. Interpolación en tiempo discreto.	18
3.6. Generadores de tonos.	19
3.7. Conclusiones.	20
4. Diseño del módulo eléctrico del estimulador acústico.	21
4.1. Características eléctricas de los dispositivos.	21
4.1.1. Sistema MED.	21
4.1.2. Microcontrolador PIC16F887.	21
4.1.3. Pantalla LCD.	22

4.2.	Tecnologías de montaje.	23
4.2.1.	THT (<i>Through-Hole Technology</i>)	23
4.2.2.	SMT (<i>Surface Mount Technology</i>)	23
4.3.	Reguladores de tensión.	23
4.3.1.	En derivación.	24
4.3.2.	En serie.	25
4.3.3.	Lineales integrados.	27
4.3.4.	Elección de configuración.	28
4.4.	Alimentación con regulador.	30
4.4.1.	Diodo zener	30
4.4.2.	Regulador zener de tensión ideal.	30
4.4.3.	Coeficiente de temperatura.	31
4.5.	Generación de entradas lógicas ($V_{I/O}$).	33
4.5.1.	Metodología de trabajo.	33
4.5.2.	Tablas de resultados.	40
4.5.3.	Elección de los elementos.	42
4.6.	Polarización PIC16F887 (V_{DD})	43
4.6.1.	Tablas de resultados.	43
4.6.2.	Elección de los elementos.	45
5.	Análisis de la arquitectura del generador de ondas programable.	47
5.1.	Elección para la configuración física del dispositivo.	47
5.2.	Elección del microcontrolador.	48
5.2.1.	Elección de lenguaje de programación.	48
5.2.2.	Comparación de los programas realizados en diferentes lenguajes.	50
5.3.	Generación de formas de onda.	51
5.3.1.	Cálculo y almacenamiento de tabla "lookup" para generar la señal sinusoidal.	51
5.3.2.	Modos de reloj externo.	52
5.3.3.	Tiempo de ejecución para las instrucciones en el microcontrolador.	54
5.4.	Reconstrucción de la señal continua a partir de retenes.	59
5.5.	Configuración para el filtrado controlado por modulación de ancho de pulso.	63
5.6.	Interfaz hombre máquina.	64
5.7.	Acople con la salida acústica.	64
5.8.	Tweeter piezo eléctrico.	65
5.9.	Diagrama esquemático y simulación.	66
5.9.1.	Repartición de los puertos del microcontrolador para la configuración del sistema.	66
5.10.	Conclusiones.	69

6. Comprobación de las señales en sus diferentes frecuencias.	71
6.1. Formas de onda generadas	71
6.1.1. Sinusoidal.	72
6.1.2. Triangular.	73
6.1.3. Rectangular.	74
6.2. Análisis espectral de las señales sinusoidales generadas.	74
6.2.1. Gráficas para el análisis de las señales sinusoidales.	75
6.3. Comprobación de señales acústicas producidas.	83
6.4. Comprobación de las señales eléctricas y acústicas generadas.	87
6.5. Conclusiones.	90
7. Conclusiones y Perspectivas.	91
A. Guía de usuario.	95
B. Curvas características eléctricas.	105
C. Códigos fuente.	111
D. Diseño del filtro paso-bajas para la interpolación.	135
E. Tarjeta fenólica PCB.	143
Bibliografía	147

2. Introducción

2.1. Antecedentes.

Los requerimientos de los usuarios de los LABEX¹ de la Facultad de Psicología, son los de poder generar y pre-establecer, por medio del sistema interfaz PCI fabricado por una compañía norteamericana denominada "MED Associates" [17], sonidos a diferentes frecuencias para la estimulación acústica en sujetos de prueba, que les permitan medir los cambios en su comportamiento, **Figura 2.1.**

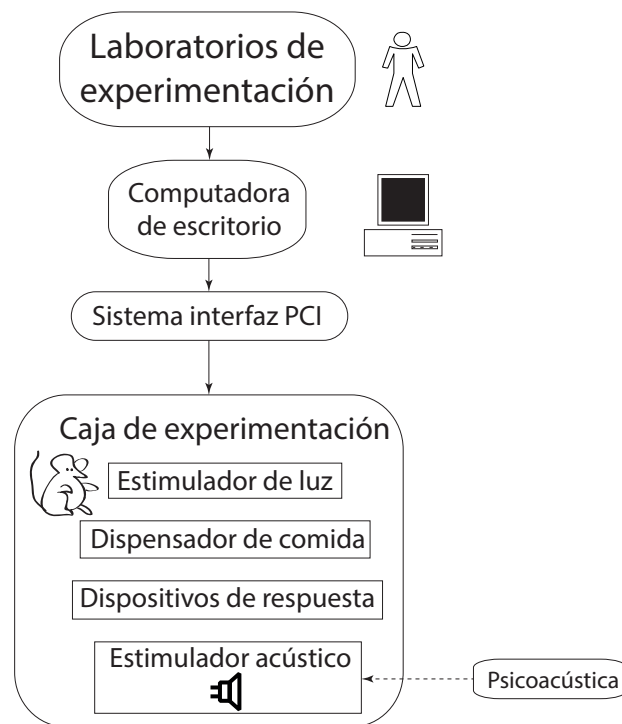


Figura 2.1.: Diagrama a bloques del sistema propuesto.

¹Laboratorios de experimentación.

2.1.1. Sistema interfaz PCI.

Los laboratorios de experimentación, que realizan estudios en la conducta de roedores, están conformados básicamente por un sistema interfaz, donde pueden estar contenidas de 1 a 16 tarjetas PCI, para ser controladas a través de una computadora de escritorio común. Cada tarjeta controla una sola caja de experimentación para diferentes transductores como son: dispensadores de comida, estimuladores de luz, estimuladores de sonido, dispositivos de respuesta como apagadores, entre otros. Ver [Figura 2.2](#) y [Figura 2.3](#).



Figura 2.2.: Sistema Interfaz PCI,[17].

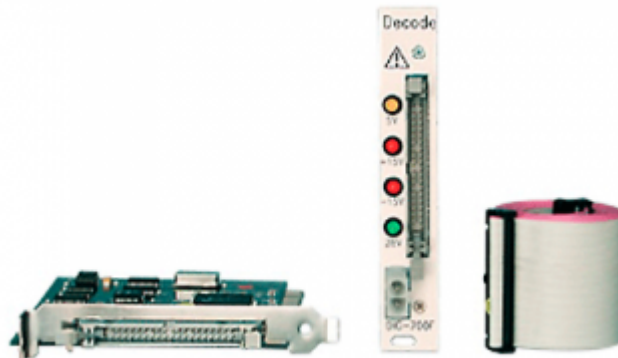


Figura 2.3.: Tarjeta Interfaz PCI,[17].

En la actualidad, los laboratorios utilizan un OOR² constituido por un circuito integrado NE555³, en configuración astable, que produce un tren de pulsos rectangulares que es una señal a una frecuencia fija y que al conectarse a un alta voz, produce un sonido. Para tener sonidos a diferentes frecuencias, fabricaron varios de estos osciladores,

²Oscilador de Onda Rectangular

³Es un circuito integrado cuya función es generar señales rectangulares con intervalos de tiempo configurables.

variando el valor de los dispositivos pasivos. Por cada OOR conectado se necesita un puerto de salida de la interfaz PCI de MED, en caso de aumentar la gama de sonidos, se debe prestar atención en los puertos de salida, ya que disminuye la posibilidad de poder conectar otros transductores indispensables.

2.1.2. Estimulador acústico comercial.

La compañía *MED Associates* tiene sus propios generadores acústicos, de los cuales solo mencionaremos uno para usarlo como referencia. Dicho dispositivo es el generador de tonos múltiple ENV-223, [Figura 2.4](#).



Figura 2.4.: Generador de tonos ENV-223,[\[17\]](#).

Este generador de tonos, con una sola forma de onda, es capaz de producir 15 sonidos a diferentes frecuencias pre-establecidas. Dicho rango de frecuencias tiene como frecuencia mínima 1 [KHz], y una frecuencia máxima de 20 [KHz]. Tales sonidos son controlados por cuatro entradas que posee, las cuales son comandadas por el SI-PCI MED a través de la decodificación de sus entradas, del orden de $2^4 - 1 = 15$ salidas acústicas, debido a que el estado faltante es de apagado.

El costo comercial del ENV-223 es de 150 dólares por lo que su adquisición para todos los equipos es muy elevada,[\[17\]](#).

2.1.3. Psicoacústica.

El sonido se produce por las vibraciones de un cuerpo o elemento elástico, cuyo fenómeno físico se define como la alteración mecánica que provoca un movimiento ondu-

Introducción

laborio a través de medios elásticos (sólidos, líquidos o gaseosos) en todas direcciones, en forma de ondas longitudinales de presión sonora,[15].

La audición es un proceso complejo que inicia en el oído, en el momento que la onda sonora (variaciones de presión del aire) golpea el tímpano, generando en el cerebro una serie de impulsos nerviosos, y culmina en las reacciones que provocará en los seres vivos; por lo tanto el sonido no es sólo físico, sino también mental dada la interpretación que el cerebro hace del sonido y las reacciones ante él,[15].

La psicoacústica es el estudio psicológico de la audición, cuyos objetivos son los de descubrir cómo procesan los sonidos el oído y el cerebro, las sensaciones que producen los sonidos en los seres vivos, y la forma en que aportan información del medio; a diferencia de la acústica que se ocupa de la señal física u onda sonora. En concreto, la psicoacústica consiste en determinar las relaciones cuantitativas y cualitativas entre las variables físicas y perceptibles del sonido como se muestra en [Tabla 2.1](#),[17].

Parámetros físicos	Parámetros subjetivos
Amplitud	Sonoridad (Volumen)
Frecuencia	Tono (Graves o Bajos)
Forma de onda	Timbre

[Tabla 2.1.](#): Relaciones cuantitativas entre las variables físicas y perceptuales del sonido.

Sonoridad: es la sensación subjetiva de la intensidad o amplitud sonora, dependiente de la frecuencia y la duración del sonido. Esta percepción acústica, también llamada volumen, no es una medida absoluta como el metro, el gramo o el segundo, sino proporcional, ya que debe describir únicamente la relación que existe entre dos presiones acústicas, por lo que obedece a una graduación logarítmica de la excitación sonora. La escala para la medición de umbrales auditivos es el belio, no obstante se utiliza el decibel.

Tono: esta determinado por la frecuencia de la señal sonora. Los sonidos graves son los que corresponden a frecuencias bajas, mientras que los agudos a frecuencias altas. La frecuencia es el número de oscilaciones que un elemento realiza entre unidad de tiempo. El oído humano percibe sonidos cuyas frecuencias oscilan entre los 16Hz y los 20KHz. Las vibraciones inferiores a los 16Hz se llaman infrasonidos y las mayores a los 20KHz, ultrasonidos,[15].

Timbre o colorido sonoro: es la cualidad que nos permite diferenciar sonidos de igual frecuencia e intensidad transmitidos por dos fuentes diferentes. La forma de onda, determinada por los armónicos, es la característica que nos permite distinguir un sonido

de cierta frecuencia producida por medios distintos. Los armónicos son vibraciones secundarias que acompañan a una vibración primaria. Normalmente cuando un cuerpo vibra no se obtiene un sonido puro, sino un compuesto de frecuencias diferentes,[15].

2.2. Generación de señales.

A lo largo del documento se hablará de como poder generar, adquirir y filtrar señales espurias, debido a que nuestro propósito es diseñar un sistema que hace uso de los procesos anteriores.

La etapa de muestreo en un dispositivo que trabaja con la generación de señales es indispensable, porque de esta forma se obtienen los parámetros de una señal discretizada para hacer uso de ella en procesos posteriores.

Entender los conceptos que abarca este tema es indispensable, para adquirir adecuadamente señales analógicas.

2.3. Configuración de la arquitectura para el generador de ondas propuesto.

Es de suma importancia elegir una configuración para el sistema que logre el mejor desempeño en todas las etapas de acoplamiento al SI-PCI MED.

Las tareas más importantes que realizará el microcontrolador son:

- Reconocer los valores lógicos que se le envíen y codificarlos en un tabla de direccionamiento de orden 2^4 , a través de la cual el microcontrolador decidirá que señal se generará.
- Reconstruir las señales digitales haciendo uso de tablas de consulta "*lookup*", en las cuales está almacenado un periodo de cada señal disponible en el dispositivo. Estas tablas de consulta serán leídas recursivamente por el microcontrolador para generar las señales continuas periódicas.
- Establecer la comunicación con una pantalla LCD⁴ para poder imprimir en ella la información del estado actual del dispositivo.

Por lo tanto, es indispensable elegir un microcontrolador que cuente con los suficientes puertos de propósito general para cumplir con estas tareas adecuadamente.

⁴Por sus siglas en inglés *Liquid Crystal Display*

2.4. Acoplamientos eléctricos del dispositivo.

Con el fin de polarizar correctamente los diferentes dispositivos y generar las entradas lógicas a partir de la tensión y corriente del SI-PCI, se eligió el regulador zener en derivación para el acoplamiento eléctrico; en base a sus características eléctricas y su precio comparado con otros circuitos reguladores de tensión.

Se calculó el valor ideal de resistencia para el resistor limitador de corriente y la tensión de ruptura en el diodo zener, para evitar alcanzar la potencia máxima en cada uno de los dispositivos reguladores.

Después de obtener esos valores ideales, se volvieron a calcular las potencias, corrientes y tensiones, pero con valores de resistores comerciales, obteniendo los porcentajes de error entre las magnitudes ideales y las reales.

2.5. Estudio de las señales discretizadas por el microcontrolador.

Se realizaron las tablas de consulta "*lookup*" de forma que se pudiera comprobar la resolución de la señal en tiempo continuo, y haciendo uso de la transformada rápida de Fourier, mostrar el análisis espectral de cada caso de la señal sinusoidal y de esta forma validar las tablas.

2.6. Alcances del proyecto.

Este trabajo consiste en diseñar e implementar un dispositivo de bajo costo para la generación de formas de onda sinusoidal, rectangular y triangular de diferentes frecuencias programables y pre-establecidas por el usuario, quien las utiliza para la estimulación acústica en sujetos de prueba.

Este nuevo dispositivo cuenta con ventajas con respecto a los generadores acústicos de los que actualmente se usan en los LABEX, debido a que puede generar tres tipos de ondas diferentes y muestra en pantalla el estado del dispositivo, además es un sistema reprogramable con la finalidad de cambiar las frecuencias de las señales de salida según la necesidad de los laboratoristas y se muestra en la [Figura 2.5](#).

Por otra parte, este dispositivo será replicado físicamente 20 veces, para integrarse al instrumental actualmente utilizado en los laboratorios de experimentación de la facultad de psicología, siendo este un aporte a la UNAM.

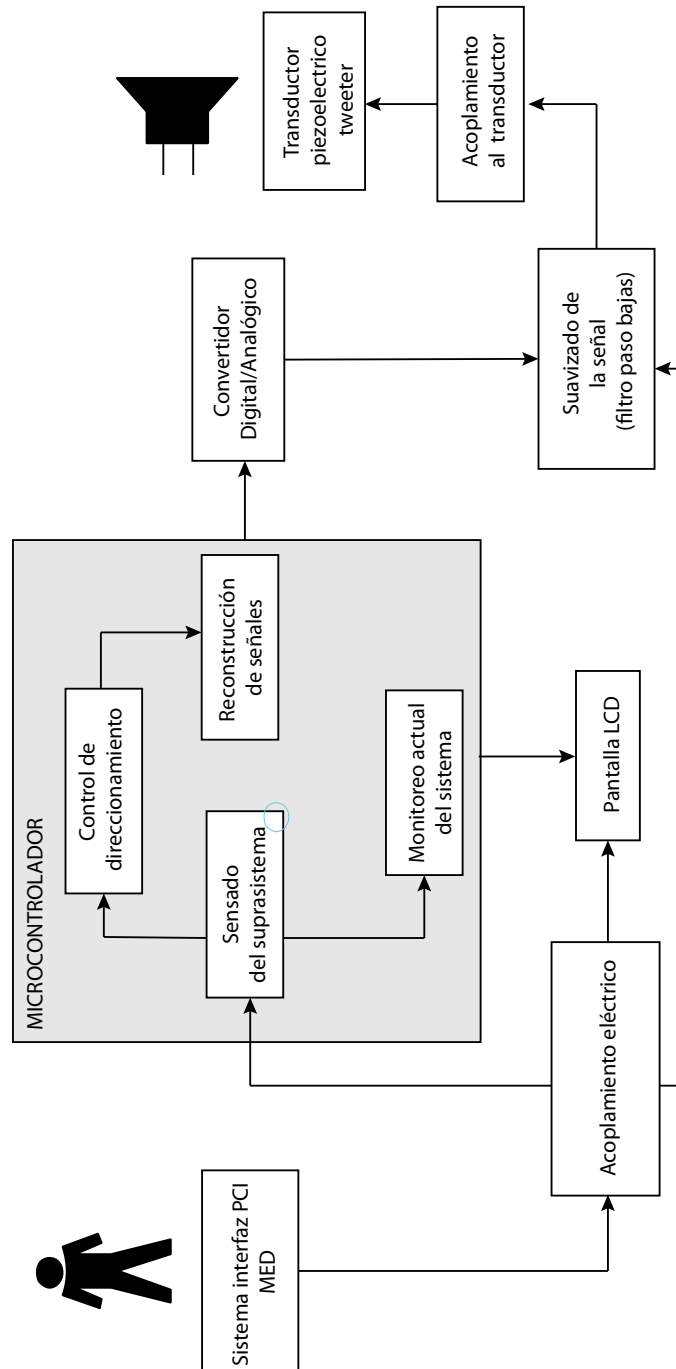


Figura 2.5.: Diagrama a bloques del estimulador acústico.

3. Generación de señales continuas y discretas.

3.1. Sintetizadores de Frecuencia.

3.1.1. Método de generación.

Las diferentes formas de diseñar un sintetizador de frecuencias son:

- Sintetizadores Directos: Realizan operaciones matemáticas sobre la frecuencia de referencia para obtener la salida. Estas operaciones son multiplicaciones o divisiones entre una señal de un oscilador local del sistema y otra señal externa al sistema. El producto resultante entre ambas es filtrado para obtener la frecuencia deseada,[28]. Este procedimiento es caro y sólo se usa en instrumentos de propósito específico, los cuales pueden alcanzar frecuencias del orden de tera hertz (THz).
- Sintetizadores Indirectos: Hacen uso del circuito integrado PLL¹. Este tipo de sintetizadores generan una alta pureza espectral para lograr la generación de la frecuencia deseada,[27]. Estos dispositivos pueden trabajar en frecuencias del orden de los giga hertz (GHz).
- Sintetizadores Digitales Directos DDS²: La mayor ventaja de estos dispositivos es poder de manipular con gran rapidez la precisión, amplitud, fase y frecuencia de una señal, a partir de un procesador de control (Microcontrolador, DSP, FPGA, etc.).

3.1.2. Clasificación.

Estos sistemas pueden ser clasificados como:

- Coherentes
- Incoherentes

¹Por sus siglas en inglés *Phase Lock Loop*.

²Por sus siglas en inglés *Direct Digital Synthesizer*.

Esta clasificación se hará con respecto al número de osciladores necesarios para generar las señales deseadas.

En los sintetizadores coherentes, solo debe haber una fuente de generación de la señal; tienen la ventaja de ser pequeños y baratos para su diseño. Por otro, lado solo puede ser usada una señal de salida a la vez de todas las disponibles.

Los sintetizadores incoherentes hacen uso de dos o más fuentes de generación de señales, esto quiere decir que hacer el diseño físico de uno de estos sistemas puede ser muy voluminoso y costoso. La característica primordial es que las señales que generan estos dispositivos, pueden ser usadas al mismo tiempo si así se requiere,[27, 28, 29].

3.2. Muestreo.

A partir de ciertas condiciones, una señal continua puede representarse o reconstruirse completamente a partir de sus muestras con intervalos del mismo tiempo; este proceso se le conoce como teorema de muestreo.

Este teorema es muy importante para el procesamiento de señales, ya que cuando una señal que ha sido muestreada a una frecuencia adecuada, puede ser vista como una señal continua debido a la cercanía entre cada muestra. Mostrando de esta forma una señal continua, pero sabiendo que con un acercamiento a dicha señal se podrá distinguir la separación entre muestras.

El papel más importante del teorema de muestreo es la forma de entrelazar las señales continuas con las discretas. Por eso el concepto de muestreo es un proceso extremadamente útil y ampliamente utilizado en la tecnología de sistemas o señales discretas y a partir de estos generar su contraparte en tiempo continuo,[18].

3.2.1. Uso del tren de impulsos unitarios.

Para desarrollar el teorema de muestreo con un método eficiente, se puede hacer uso de un tren de impulsos unitarios periódicos $p(t)$ que al multiplicar a la señal continua $x(t)$ obtendremos nuestra señal discretizada $x_p(t)$, que se muestra en la **Figura 3.1**,[18, 19].

$$x_p(t) = x(t) \cdot p(t).$$

donde $p(t)$

$$\sum_{n=-\infty}^{\infty} \delta(t - nT).$$

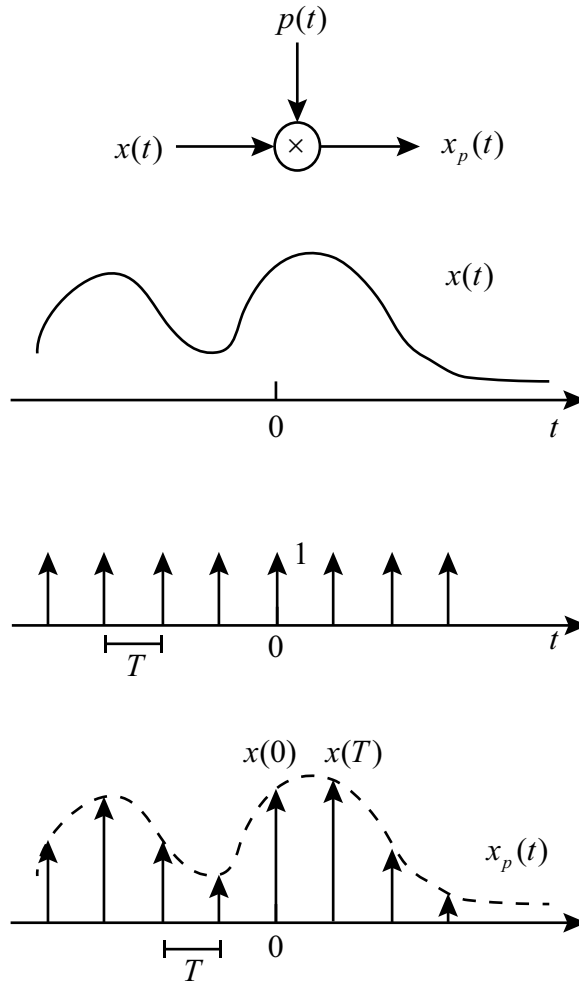


Figura 3.1.: Muestreo a partir de un tren de impulsos unitarios.

Es importante señalar que la frecuencia de muestreo ω_M , debe exceder la frecuencia máxima de la señal deseada ω_0 por lo menos al doble de esta³, lo que podemos interpretar como:

$$\omega_0 \leq 2\omega_M$$

³Esta condición es comúnmente conocida como: Teorema de muestreo de Nyquist.

3.3. Reconstrucción de la señal usando retén de orden cero.

Este tipo de sistema es comúnmente utilizado, debido a que generar un tren de impulsos de gran amplitud, es difícil de producir y transmitir. Por lo que resulta conveniente generar la señal de muestreo en una forma conocida como retén de orden cero.

Dicho sistema muestrea la señal $x(t)$ en un instante determinado y retiene ese valor hasta el siguiente instante de tiempo, donde se tomará la siguiente muestra como se aprecia en la [Figura 3.2](#).

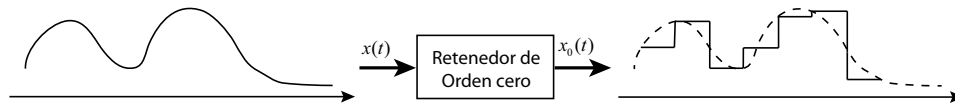


Figura 3.2.: Muestreo utilizando un retén de orden cero.

La reconstrucción de $x(t)$ a partir de la salida del retén de orden cero se lleva a cabo mediante un filtrado de bajas frecuencias⁴.

Sin embargo dicho filtro requerido no tendrá ganancia constante en la banda de paso. Para lograr obtener la característica necesaria del filtro, se debe observar que la señal de salida $x_o(t)$ del retén de orden cero, puede ser generada por el proceso de muestreo con un tren de impulsos seguido de un sistema lineal invariante en el tiempo con respuesta rectangular al impulso, como se ve en la [Figura 3.3](#).

3.3.1. Interpolación lineal.

La interpolación es el ajuste de una señal continua a un conjunto de valores de muestras. Es un proceso comúnmente utilizado para la reconstrucción de la señal $x(t)$ para obtener mayor precisión con la misma cantidad de muestras tomadas por el retén de orden cero.

$$\sum_{n=-\infty}^{\infty} x(nT)\delta(t - nT).$$

La ecuación anterior muestra como debe ser ajustada una curva continua entre los puntos de muestra $x(nT)$ por lo que representa una fórmula de interpolación lineal,[18, 19].

⁴Filtro Paso Bajas.

3.3 Reconstrucción de la señal usando retén de orden cero.

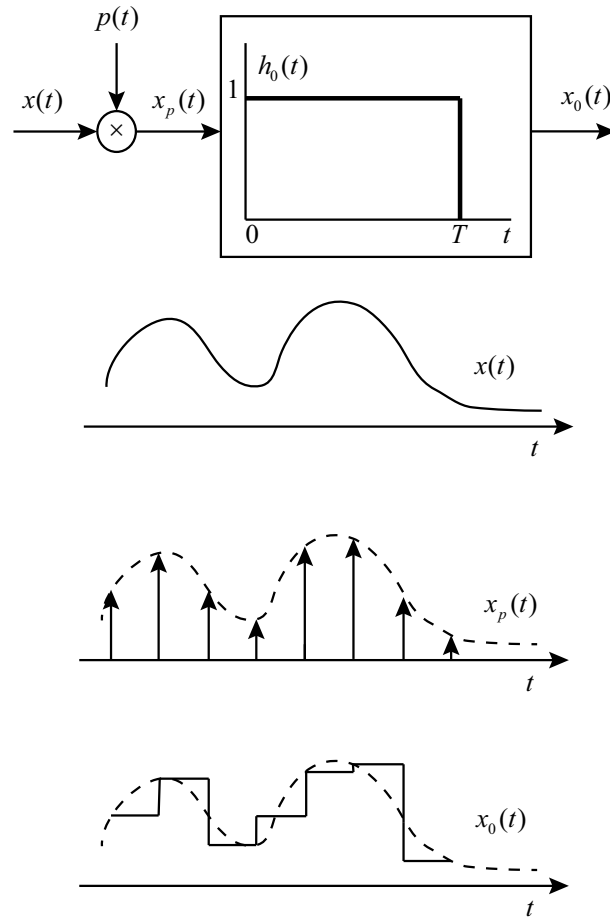


Figura 3.3.: Retén de orden cero como muestreador de tren de impulsos seguido de un sistema lineal invariante en el tiempo con respuesta rectangular al impulso

3.3.2. Interpolación de banda limitada.

Esta interpolación se realiza utilizando la respuesta al impulso de un filtro paso bajas ideal, debido a que realiza la reconstrucción mas exacta de la señal $x(t)$, si esta es en banda limitada y la frecuencia de muestro satisface las condiciones necesarias para cumplir con el teorema de muestreo de Nyquist.

Por lo tanto es una interpolación ideal, tomando en cuenta que las señal de salida del retén de orden cero ha pasado por un filtro ideal paso bajas como se observa en la [Figura 3.4](#).

Este proceso esta descrito por la ecuación siguiente:

$$x(t) = \sum_{n=-\infty}^{\infty} x(nT) \frac{\text{sen}(\pi/T)(t - nT)}{(\pi/T)(t - nT)}.$$

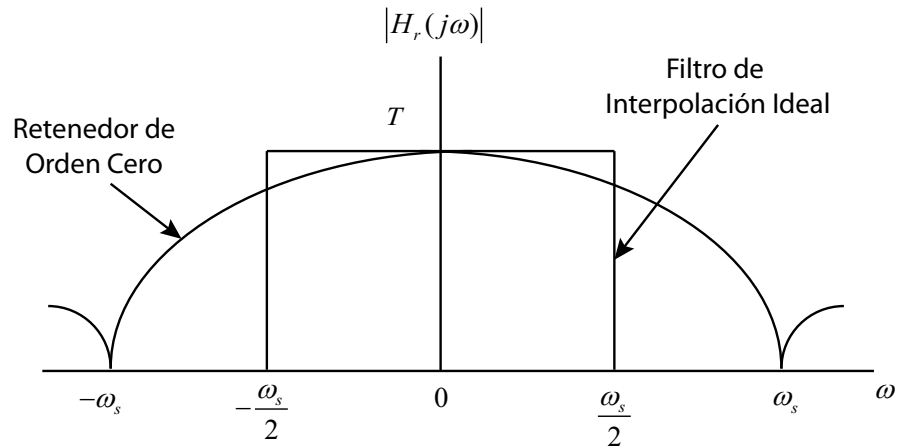


Figura 3.4.: Función de transferencia para el retén de orden cero y para el filtro de interpolación ideal.

3.4. Conversión de señales Digital-Analógico.

La base teórica para la conversión de una señal continua a una señal discreta y la reconstrucción de la señal discreta para la obtención de la señal continua, reside en el teorema de muestreo que debe cumplir al menos con el teorema de muestreo de Nyquist para lograr la reconstrucción de dicha señal analógica continua.

La señal continua y la señal discreta se relacionan de la siguiente forma:

$$x_d[n] = x_c[nT]$$

y

$$y_d[n] = y_c[nT]$$

donde T es el periodo de muestreo, la variable x denota la señal de entrada ya sea continua o discreta, mientras que la variable y hace referencia a las señales continua y discreta después de haber sido procesadas por el SLIT⁵. Esta secuencia se muestra en el diagrama de la [Figura 3.5](#).

La conversión Digital/Analógico se realiza por medio de una interpolación entre los valores de muestras proporcionadas a la salida del SLIT y que están relacionados por la ecuación antes mencionada:

$$y_d[n] = y_c[nT]$$

⁵Sistema Lineal Invariante en el Tiempo

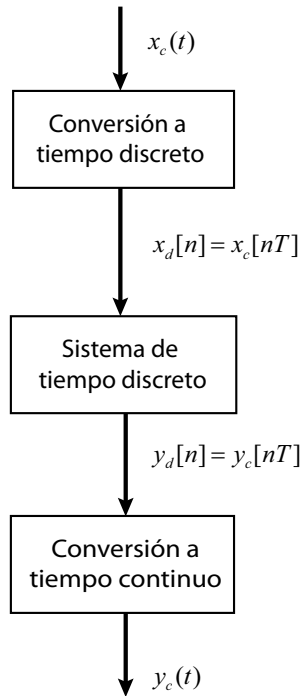


Figura 3.5.: Notación para la conversión de tiempo continuo a discreto y la conversión de tiempo discreto a tiempo continuo.

Después del procesamiento con el SLIT, la secuencia resultante se convertirá de nuevo en una señal continua.

Para aterrizar el concepto, a partir de la secuencia $y[n]$ se puede generar el tren de impulsos continuo $y_p(t)$.

La reconstrucción de la señal continua $y_c(t)$ a partir de un tren de impulsos se logra haciendo que esta pase por un filtro paso bajas como muestra la [Figura 3.6](#).

3.5. Cambio de tasa de muestreo en señales discretas

El muestreo de señales discretas es semejante al muestreo de señales continuas ya que la secuencia $x_p[n]$ que se obtiene del muestreo de la señal $x_c(t)$ es igual a la secuencia $x[n]$ en múltiplos enteros del periodo de muestreo N y es cero en las muestras intermedias, de forma que se puede expresar de la siguiente forma:

$$x_p[n] = \begin{cases} x_n[n] & n = \text{múltiplo de } N \\ 0 & \text{con otro valor} \end{cases}$$

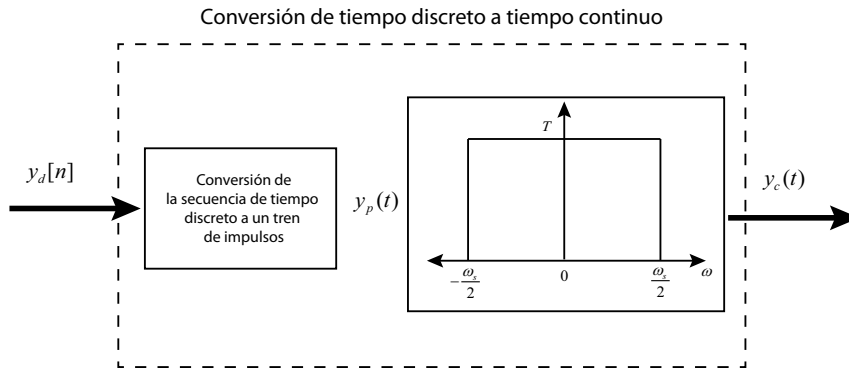


Figura 3.6.: Conversión de una secuencia de tiempo discreto a una señal continua.

Al igual que en el caso continuo se hace uso de la ecuación:

$$x_p[n] = x[n] \cdot p[n] = \sum_{k=-\infty}^{\infty} x[kN] \cdot \delta[n - kN].$$

y se muestra en la **Figura 3.7** en donde la transformada de Fourier para la secuencia de muestreo $p[n]$ es:

$$P(e^{j\omega}) = \frac{2\pi}{N} \sum_{n=-\infty}^{\infty} \delta(\omega - k\omega_s)$$

donde ω_s , es la frecuencia de muestreo y es igual a $2\pi/N$, y ω es la frecuencia máxima de salida de la señal continua.

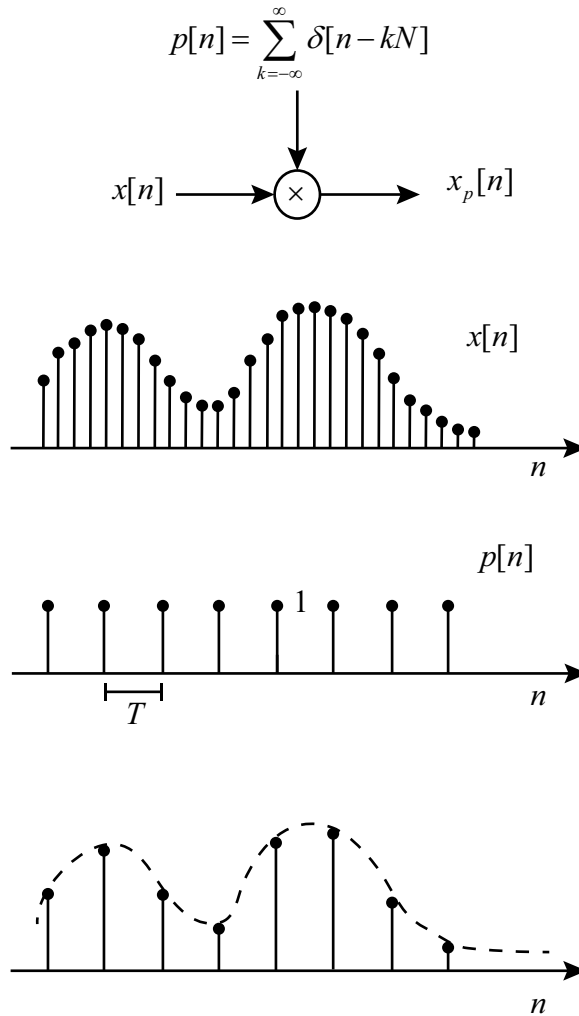


Figura 3.7.: Muestreo discreto.

Por lo que la ecuación para el muestreo discreto queda de la siguiente manera:

$$X_p(e^{j\omega}) = \frac{1}{N} \sum_{n=-\infty}^{\infty} X(e^{j(\omega - k\omega_s)}).$$

donde se debe cumplir la condición $\omega_s - \omega > \omega$ para evitar el efecto *aliasing*⁶.

A partir de esto, $x[n]$ se puede reconstruir a partir de $x_p[n]$ por medio de un filtro paso bajas con ganancia N y una frecuencia de corte mayor que ω y menor que $\omega_s - \omega$.

La reconstrucción de $x[n]$ mediante el uso de un filtro paso bajas aplicado a $x_p[n]$ se puede interpretar en el dominio del tiempo como una fórmula de interpolación similar

⁶Traslape de señales en el dominio de la frecuencia

a la de banda limitada que se vio en la [Sección 3.4](#), cuando la respuesta al impulso del filtro paso bajas es denotada por $h[n]$ obtenemos

$$h[n] = \frac{N\omega_c}{\pi} \cdot \frac{\text{sen}(\omega_c n)}{\omega_c n}$$

Por lo que la señal reconstruida estaría dada por:

$$x_r[n] = x_p[n] * h[n]$$

y que puede ser vista de una manera equivalente como:

$$x_r[n] = x[kN] \cdot \frac{N\omega_c}{\pi} \cdot \frac{\text{sen}(\omega_c)(n - kN)}{(\omega_c)(n - kN)}$$

Esta última ecuación representa la interpolación ideal de banda limitada y requiere de la construcción de un filtro paso bajas ideal.

3.5.1. Interpolación en tiempo discreto.

Este tema explica una parte muy importante en el desarrollo del sistema que se expone en este documento, debido a que los principios de muestreo son muy utilizados.

Para este nuevo tema sabemos que los instantes de muestreo de $x_p[n]$ es cero. Así que esta secuencia muestreada se reemplazara por una nueva denominada $x_b[n]$, ya que cada muestra que se encontrará en cada uno de los valores de N en $x_p[n]$ de tal manera que:

$$x_b[n] = x_p[nN]$$

La operación de extracción de cada muestra N , comúnmente es conocida como decimación y se puede apreciar en la [Figura 3.8](#)

Para poder determinar el efecto de decimación en el dominio de la frecuencia, debemos determinar la relación entre $X_b(e^{j\omega})$ que es la transformada de Fourier de: $x_b[n]$ y $X(e^{j\omega})$.

Y que queda de la forma:

$$X_b(e^{j\omega}) = \sum_{k=-\infty}^{\infty} x_p[kN] \cdot e^{-j\omega k}$$

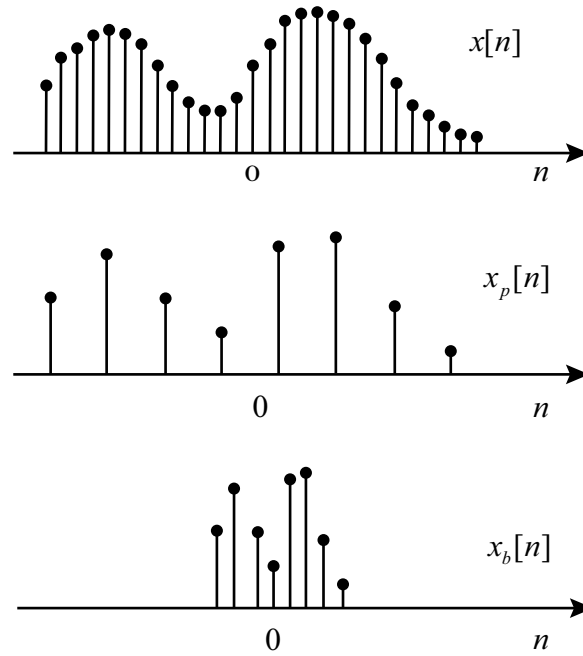


Figura 3.8.: Relación entre $x_p[n]$ correspondiente al muestreo y $x_b[n]$ correspondiente a la decimación.

3.6. Generadores de tonos.

Como se ha visto existen diferentes formas de poder generar tonos, ya sea analógicamente o digitalmente por ejemplo:

- Cristales, multivibradores, etc. La mayor desventaja de implementar esta solución, es que el dispositivo final es muy voluminoso.
- Puede hacerse uso de un DSP⁷. Gracias a la velocidad de procesamiento de estos dispositivos, es posible programar un código fuente con algún algoritmo que sea capaz de que este dispositivo cumpla las características de un oscilador digital en tiempo real.
- Otra forma de lograrlo es implementar un microcontrolador o DSP para gobernar mediante algún bus de datos tales como *SPI*⁸ o *I²C*⁹, un DDS¹⁰ el cual es un dispositivo que ya está optimizado para poder generar las señales deseadas a la salida después de leer la configuración que le es enviada.

⁷Procesador Digital de Señales DSP por sus siglas en inglés

⁸Del inglés *Serial Peripheral Interface*.

⁹Del inglés *Inter-Integrated Circuit*

¹⁰Sintetizador Directo Digital DDS por sus siglas en inglés

- También se puede hacer uso de tablas "*lookup*" almacenadas en la memoria del microcontrolador o DSP para que éste sólo consulte los valores previamente calculados y escritos en memoria, respetando las características como la longitud de palabra del dispositivo utilizado.

Un detalle que no se puede perder de vista es que las opciones anteriores que generan la señal digital desde un algoritmo o una tabla "*lookup*", necesitarán acoplarse a un convertidor digital/analógico para la reconstrucción de la señal continua.

3.7. Conclusiones.

La configuración que se seleccionó fue un sintetizador digital directo coherente para que pueda ser controlado con mayor facilidad a través del SI-PCI MED. Además el ancho de banda que caracteriza a esta clase de dispositivos es muy amplia; propiedad que nos sirve para solucionar la problemática.

Usar la teoría de muestreo y cumplir con sus criterios como el teorema de muestreo de Nyquist es imprescindible para el desarrollo de sistemas que realizan adquisición de señales continuas o discretas.

Con el fin de lograr su procesamiento y así poder generar o reconstruir una señal $x[n]$ lo mas parecida a $x(t)$. De tal forma que se evite tener problemas de traslape (*aliasing*) en el dominio de la frecuencia y no tener una señal semejante a la original.

4. Diseño del módulo eléctrico del estimulador acústico.

El diseño comprende los siguientes puntos: Circuito eléctrico en montaje THT¹ y SMT²; y la tarjeta fenólica PCB³.

Tomando en cuenta las características eléctricas de los dispositivos: PIC16F887, pantalla LCD, etc., debemos de integrar los elementos necesarios para regular la tensión y la corriente de estos, a fin de polarizar los dispositivos correctamente y generar las entradas lógicas, a partir de la tensión y corriente que nos brinda el SI-PCI MED [Figura 2.2](#).

4.1. Características eléctricas de los dispositivos.

4.1.1. Sistema MED.

Con el uso de un multímetro sabemos que el SI-PCI MED entrega tensiones de $V_{cd} = 28[V]$ para polarizar los circuitos externos que se le integren. Del gabinete sabemos que la corriente máxima es de $5[A]$, como se muestra [Figura 4.1](#)

4.1.2. Microcontrolador PIC16F887.

De las hojas de especificaciones del microcontrolador[10], sabemos que la tensión de polarización V_{DD} esta en el rango de $-0.3[V]$ a $+6.5[V]$ con respecto a la referencia denotada como V_{SS} . La tensión en los demás pines ($V_{I/O}$) debe estar entre a $-0.3[V]$ y $V_{DD} + 0.3[V]$ respecto a V_{SS} .

La corriente máxima que obtendremos del pin de la referencia V_{SS} es $300[mA]$, mientras que la máxima corriente que entra en el pin de polarización V_{DD} es $250[mA]$ [5]. La máxima corriente que puede entrar y salir de cualquier otro pin (I/O) es de $25[mA]$.

La disipación total de potencia es de $800[mW]$

¹Through-Hole Technology

²Surface Mount Technology

³Printed Circuit Board

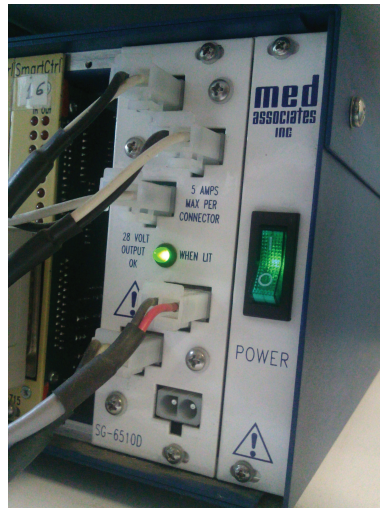


Figura 4.1.: Gabinete del sistema interfaz PCI de MED.

4.1.3. Pantalla LCD.

La tensión de polarización V_{LCD} esta en el rango de $4.5[V]$ a $5.5[V]$ con respecto a V_{SS} , [12].

La corriente esta en el rango de $1.5[mA]$ a $3.0[mA]$ con la condición de que $V_{CC} = 5.0[V]$.

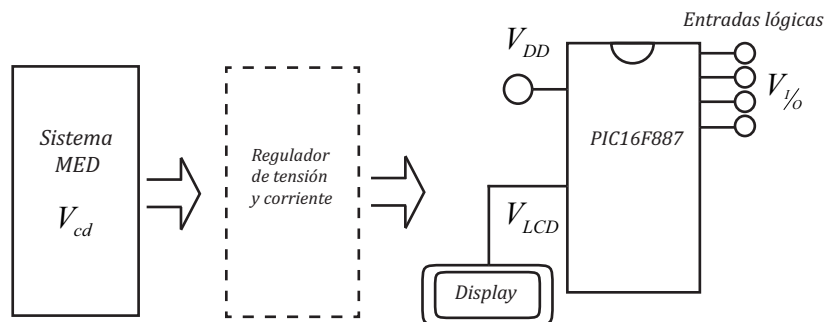


Figura 4.2.: Bloque de regulación de tensión y corriente.

El rango de tensión del microcontrolador y del multiplexor es bastante amplio comparado con el rango de la pantalla LCD, que es $5[V] \pm 10\%$; por lo tanto proponemos que la tensión de polarización estándar para las entradas lógicas y todos los dispositivos sea de $V_{STD} = 5.0[V]$, fijando nuestra referencia a $V_{SS} = 0[V]$ y un rango de error de $\pm 10\%$.

Analizando las características eléctricas del LCD, podemos polarizar el dispositivo correctamente mediante el uso del PIC16F887, ya que cubre todos sus requerimientos

eléctricos para un buen funcionamiento. De tal forma que nuestro sistema eléctrico se muestra en [Figura 4.2](#)

4.2. Tecnologías de montaje.

Gracias al apoyo del grupo académico de electrónica del departamento de instrumentación y medición, del CCADET⁴, la placa de circuito impreso o PCB, se pudo fabricar para los dos tipos de montaje que existe: THT y SMT.

4.2.1. THT (*Through-Hole Technology*)

THT es la técnica de montaje de dispositivos electrónicos de inserción en una PCB, cuyas pistas conductoras poseen barrenos, a través de los cuales las terminales del componente son insertadas, para ser fijadas con soldadura logrando la unión eléctrica,[7]. Una vez unido se recortan los sobrantes de las terminales a fin de evitar algún tipo de antena donde pueda introducir "ruido" o una señal no deseada. Los dispositivos se encuentran en encapsulados radiales, axiales y los circuitos integrados en encapsulado DIP⁵.

4.2.2. SMT (*Surface Mount Technology*)

SMT es la técnica de montaje que utiliza dispositivos activos y pasivos miniaturizados llamados SMD⁶, cuyas terminales cortas se sueldan en las zonas conductoras del PCB llamadas *lands*, de esta manera se evitan las perforaciones[6], el beneficio de las terminales cortas es no generar antenas y menor resistencia parásita.

4.3. Reguladores de tensión.

Para regular la tensión y corriente en los dispositivos usaremos reguladores lineales, estos circuitos poseen dispositivos que trabajan en su zona lineal a fin de mantener constante la tensión en la carga,[1]. Estos reguladores se dividen en dos grupos: en derivación y en serie. Los puntos que tomamos en cuenta para descartar los circuitos de regulación son:

- Las especificaciones mínimas y máximas de operación de los dispositivos.
- El uso mínimo de componentes para regular, lo cual implica un bajo costo.

⁴Centro de Ciencias Aplicadas y Desarrollo Tecnológico UNAM

⁵Por sus siglas en ingles: *Dual In-line Package*

⁶Por sus siglas en ingles: *Surface Mount Device*

4.3.1. En derivación.

La característica principal de este grupo es tener los dispositivos reguladores en paralelo con la carga, debido a esto, cuando la tensión a la entrada es constante (V_{in}), la corriente de entrada (I_S) es casi constante ante los cambios en la corriente por la carga (I_L). La corriente máxima sobre la carga (I_{LMAX}) con tensión regulada (V_Z) es igual a la corriente de entrada (I_S).

Algunos de los circuitos mas representativos son: regulador zener, regulador zener con transistor, regulador zener con transistor y arreglo de resistores, regulador zener con transistor, amplificador operacional y arreglo de resistores, etc.

Una ventaja de los reguladores en derivación es que su diseño tiene protección contra cortos circuitos. Si la carga se llega a cortocircuitar, ningún elemento del circuito se afecta. La corriente que se presenta es:

$$I_S = \frac{V_{in}}{R_S}$$

Cuyo valor no es lo suficientemente grande para dañar algún componente (este punto se detallará más adelante).

El defecto de estos reguladores es el rendimiento en la potencia, ya que la mayor parte de la potencia se disipa en los resistores en serie y si es el caso, en el transistor. El rendimiento se define como:

$$Rd = \frac{P_{out}}{P_{in}} \times 100 \%$$

Donde $P_{out} = V_{out} \cdot I_L$ es la potencia en la carga y $P_{in} = V_{in} \cdot I_{in}$ es la potencia de entrada.

La particularidad del diodo zener es que al funcionar en la zona de ruptura, produce una tensión a la salida casi constante a su tensión zener, como el mostrado en la **Figura 4.3**

Tenemos el siguiente comportamiento: si la corriente sobre la carga cambia, ya sea en incremento o decremento, la corriente sobre el diodo zener se comportará inversamente, a fin de mantener la corriente en la carga lo más constante posible. La máxima corriente que se puede presentar sobre la carga con tensión regulada, es igual a la corriente de entrada y se produce cuando la corriente sobre el diodo zener es casi cero. A este circuito se le llama regulador zener en derivación.

Si la carga demanda mucha corriente, se corre el riesgo de que el diodo zener salga de la zona de ruptura. Con el fin de mejorar la regulación en la carga, se incorpora un transistor al circuito como se aprecia en la **Figura 4.4**:

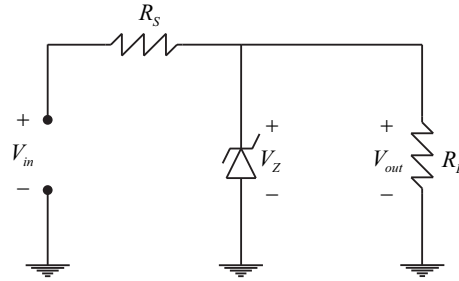


Figura 4.3.: Regulador en derivación, regulador zener.

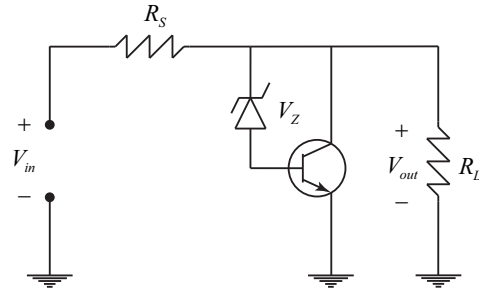


Figura 4.4.: Regulador en derivación, regulador zener con transistor.

La tensión en la carga será casi constante e igual a:

$$V_{out} = V_Z + V_{BE}$$

Esto se debe a que si se presenta un cambio de tensión en la carga, el cambio es transmitido a la base del transistor a través del diodo zener. La tensión que alimenta la base varía la corriente de colector sobre el resistor en serie, lo que implica un cambio opuesto en la tensión del resistor en serie, compensando la tensión en la carga.

Al añadir más componentes como arreglos de resistores o amplificadores operacionales, vamos a garantizar una mejor regulación debida a la retroalimentación, reducción de los efectos de la tensión base-emisor del transistor, así como minimizar el efecto llamado "coeficiente de temperatura" del diodo zener, etc.

4.3.2. En serie.

La característica de este grupo es que la corriente en la carga es aproximadamente igual a la corriente de entrada, esto implica que al variar la corriente en la carga, la corriente en la entrada variará con la misma razón de cambio,[1].

Los reguladores más simples son: seguidor zener, regulador serie con dos transistores, regulador serie con retroalimentación negativa; cuyo defecto es no tener protección

contra cortos circuitos. Si la carga se pone en cortocircuito, la corriente sobre ella será tan grande que dañará al transistor de paso y podría dañar algún elemento de la fuente que alimenta el regulador en serie.

El transistor usado en estos reguladores se le llama transistor de paso, ya que toda la corriente que requiere la carga lo atraviesa. La corriente sobre las otras ramas es tan pequeña que es despreciable. Por lo que la mayor parte de la potencia se disipa en el transistor de paso. Es aquí donde vemos una desventaja, si la corriente en el transistor de paso es grande para eliminar el exceso de calor, se deben de usar disipadores o ventiladores.

La **Figura 4.5** muestra el esquema del circuito seguidor zener.

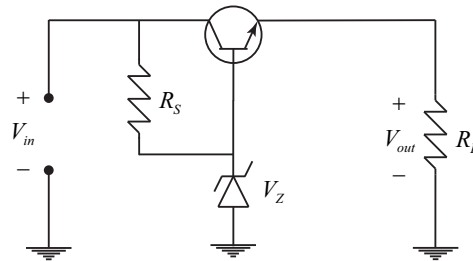


Figura 4.5.: Regulador serie, seguidor zener.

Al trabajar el diodo zener en la zona de ruptura, genera la tensión zener en la base del transistor. Éste último esta conectado a un seguidor de emisor. De tal forma que a la salida tenemos:

$$V_{out} = V_Z + V_{BE}$$

Si la corriente en la carga cambia, la tensión del zener y base-emisor solo tendrán pequeños cambios.

En aplicaciones donde la potencia en la carga no sea muy alta, inferior a 10[W], los reguladores en serie son la mejor opción, ya que su rendimiento oscila entre el 50 y 70 por ciento,[1]. Rendimiento se define como:

$$R d = \frac{V_{out}}{V_{in}} \times 100 \%$$

Con lo que el rendimiento óptimo se produciría cuando la tensión de salida sea casi igual a la tensión de entrada, o visto de otra manera, la diferencia entre la tensión de entrada y la tensión regulada (tensión diferencial entrada/salida) tiene que ser un valor pequeño.

Para reducir la probabilidad de una falla por corto circuito, se añade un arreglo llamado "transistor de limitación de corriente", que sirve para limitar la corriente de corto circuito a cierto valor definido por el diseñador; esta configuración se le llama "regulador serie con limitador de corriente". Un arreglo más complejo, es aquel que al limitador de corriente se le agrega un "reductor de corriente", el cual sirve para disipar la potencia del transistor de paso cuando hay corto en la carga.

4.3.3. Lineales integrados.

Existe una gran cantidad de circuitos integrados para regular tensión, todos de tipo serie. Se encuentran en encapsulados de plástico o metálicos que van de 3 a 14 pines, siendo los más empleados aquellos que tienen tres, debido a su fácil uso. A estos circuitos integrados se les puede adicionar ciertas configuraciones, como: arreglos de resistores para limitar corriente, capacitores de desacoplo para evitar oscilaciones en el circuito integrado, etc.

Los reguladores integrados básicos se clasifican de varias maneras de acuerdo a ciertas características, por ejemplo: si la tensión de salida tiene un valor fijo, ya sea positivo, negativo, o si es variable la corriente que puede entregar a la carga.

Un parámetro de interés es el llamado *dropout*, el cual se define como la diferencia mínima entre la tensión de entrada y la tensión de salida, para lograr una óptima regulación dicho de otra manera, la tensión de entrada al regulador debe ser mayor que la salida, a cierto rango definido por el fabricante. Si no se cumple, el circuito no regulará. Asimismo, se debe tener cuidado con la máxima tensión de entrada, debido a la máxima disipación de potencia.

LA SERIE LM78XX, son reguladores fijos positivos de tres terminales, donde "XX" se sustituyen por dos números que indican la tensión que regula. Estos dispositivos están diseñados para entregar: 5, 6, 8, 10, 12, 15, 18 o 24[V], con tolerancias de tensión de $\pm 4\%$, [25].

El diseño de estos dispositivos integra los arreglos complejos de los reguladores en serie. A la entrada tenemos un transistor de paso que soporta corrientes en la carga de 1[A], usando un disipador de calor adecuado. Cuenta con un divisor de tensión, el cual mide la tensión de salida y genera una tensión de retroalimentación en la entrada inversora de un amplificador de alta ganancia. La entrada no inversora esta conectado a un voltaje de referencia, que sería equivalente a la tensión de un diodo zener. La salida del amplificador está calculada para conseguir la tensión regulada. A esto se incluye protección térmica y limitación de corriente, con las cuales el dispositivo deja de operar cuando la temperatura interna alcanza a los 175[°C]. Esto se puede explicar más claramente en la **Figura 4.6**.

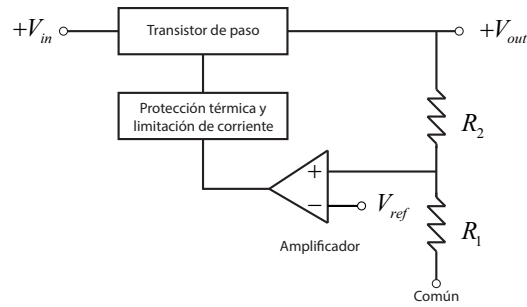


Figura 4.6.: Diagrama a bloques, regulador integrado

4.3.4. Elección de configuración.

La primera opción que se debe de descartar, es el uso de un solo regulador integrado LM7805. Aunque este regulador tiene características que solo se pueden conseguir integrando un gran arreglo de elementos, el inconveniente es la tensión máxima de entrada. De la hoja de datos sabemos que para obtener una tensión de salida regulada a $V_{out} = 5[V]$ a una potencia $P_O \leq 15[W]$, la tensión máxima a la entrada debe ser $V_{inMAX} = 20[V]$, dato por el cual ésta opción queda descartada.

Una alternativa es usar dos reguladores integrados en serie: Un LM7815 cuya tensión máxima de entrada es $V_{inMAX} = 30[V]$, y su salida $V_{out} = 15[V]$ usarla de entrada en un LM7805. Para prevenir alguna falla debida a la disipación térmica, se necesitaría usar disipadores de calor. De esta manera queda resuelto el problema para regular solo una de las varias entradas. El costo aproximado (Abril 2012,[9].) es \$3.961 de dolar:

- Un LM7805: \$0.761 de dolar
- Un LM7815: \$0.46 de dolar
- Dos disipadores de calor \$2.74 de dolar

En cada circuito tenemos ocho entradas, por lo cual el costo es muy elevado. Por lo tanto, la regulación con circuitos integrados no es factible.

La segunda opción que se descarta es el uso de configuraciones complejas ya sea en reguladores en derivación o en serie, con el fin de mejorar la regulación de tensión y corriente. Al aumentar el número de componentes se aumentaría el tamaño del PCB, igualmente el costo. Por otra parte, no mejorar la regulación es justificable debido a que las características eléctricas de los dispositivos nos permiten cierto margen de error, teniendo en cuenta que un diseño de regulador de tensión simple debe ser óptimo para obtener la tensión de polarización estándar propuesta.

Si implementamos para cada entrada un regulador en derivación con retroalimentación, el costo aproximado (Abril 2012,[9].) es \$2.56 de dolar

- Un transistor: \$0.79 de dolar

- Un amplificador operacional: \$0.973 de dolar
- Cuatro resistores: \$0.348 de dolar
- Un diodo zener: \$0.455 de dolar

Y la cantidad total multiplicada por las entradas nos generan un costo elevado.

Si implementamos un regulador serie con limitación y reducción de corriente, el costo aproximado (Abril 2012,[9].) es \$3.617 de dolar

- Un amplificador operacional: \$0.973 de dolar
- Dos transistores: \$1.58 de dolar
- Siete resistores: \$0.609 de dolar
- Un diodo zener: \$0.455 de dolar

Por la cantidad de entradas, se tiene un costo alto. Debido a esto, solo quedan dos configuraciones: seguidor zener o regulador zener.

Haciendo cálculos para obtener la potencia y el rendimiento en dos casos donde la corriente es variable, pero sabemos que la tensión del SI-PCI MED [Subsección 2.1.1](#) es $V_{in} = 28[V]$, la cual debemos de regular a $V_{out} = 5[V]$, tenemos:

Caso-A: Estableciendo que el microcontrolador PIC16F887 requiere $50[mA]$ para su alimentación, la potencia que tendremos que disipar aproximadamente será de: $P_1 = (28[V] - 5[V]) \cdot 50[mA] = 1[W]$. Si usamos un factor de seguridad de 2, necesitamos un elemento que disipe $2[W]$.

Caso-B: Para las entradas lógicas, si establecemos una corriente de $15[mA]$, la potencia que disiparemos es de: $P_2 = (28[V] - 5[V]) \cdot 15[mA] = 345[mW]$. Usando un factor de seguridad de 2, necesitamos un elemento que disipe $1[W]$

Rendimiento aproximado con seguidor zener. Para los dos casos será el mismo. $Rd = \frac{28}{5} \times 100 \% = 17.86 \%$

Rendimiento aproximado con regulador zener. Para los dos casos será el mismo, debido a que el zener consume el 10% de la corriente que entrega el resistor limitador de corriente.

$$\text{Caso-A: } Rd_1 = \frac{5[V] \cdot 45[mA]}{28[V] \cdot 50[mA]} \times 100 \% = 16.07 \%$$

$$\text{Caso-B: } Rd_2 = \frac{5[V] \cdot 13.5[mA]}{28[V] \cdot 15[mA]} \times 100 \% = 16.07 \%$$

Teóricamente el seguidor zener, en cuanto a rendimiento, es mejor opción que el regulador zener, para nuestro caso no lo es.

En cuanto al costo (Abril 2012,[9].) y de acuerdo a su disipación de potencia, tenemos la siguiente tabla comparativa:

	2[W]	1[W]
Transistor	\$1.17	\$0.79
Resistor	\$0.741	\$0.087

Tabla A.0.: Costo en dolares, según la disipación de potencia.

Por lo tanto, el uso de resistores es la elección más económica.

Un motivo para no usar el seguidor zener es que no cuenta con protección contra cortocircuito, poniendo en riesgo el SI-PCI MED [Subsección 2.1.1](#).

Tomando en cuenta todos los puntos anteriores, la regulación de tensión y corriente se realizará con un regulador zener en derivación.

4.4. Alimentación con regulador.

4.4.1. Diodo zener

El diodo zener al igual que un diodo rectificador de silicio, si se polariza en zona directa, tiene una barrera de potencial mejor conocida como tensión umbral de $0.7[V]$ a temperatura ambiente de $25[^\circ C]$, tensión a la cual la corriente empieza a incrementarse rápidamente. Al polarizarse en inversa se tiene una pequeña corriente de fuga entre los $0[V]$ y la zona de ruptura, la cual se puede ignorar para frecuencias de bajo de los $10[MHz]$,[\[2\]](#).

La diferencia radica en que el diodo zener está fabricado con el propósito de trabajar eficientemente en la zona de ruptura. En esta zona, se tiene un aumento muy pronunciado de corriente inversa (I_z) a una tensión casi constante, aproximadamente a la tensión zener (V_z). Es por este motivo que el diodo zener es utilizado como regulador de tensión, ya que se obtiene una tensión casi constante (V_z) aun con variaciones en la tensión de la fuente (V_{cd}) y de la resistencia de carga(R_L). La [Figura 4.7](#) muestra la gráfica del comportamiento del diodo zener.

4.4.2. Regulador zener de tensión ideal.

Para que el diodo zener funcione como regulador, se debe de polarizar en inversa como lo muestra la [Figura 4.8](#) y la tensión de la fuente de alimentación tiene que ser mayor que la tensión de ruptura, de tal modo que la tensión en la carga será igual al valor de la tensión de ruptura del diodo zener.

En las hojas de especificaciones aparece un dato que se denomina I_{ZM} ; Es de vital importancia, debido a que nos indica el punto para dos posibles estados del diodo zener. Si la corriente inversa es menor a I_{ZM} , este funcionará en su zona de seguridad.

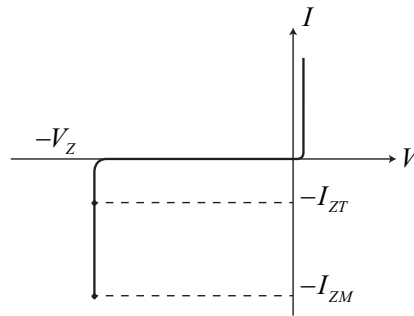


Figura 4.7.: Gráfica de I vs V del zener

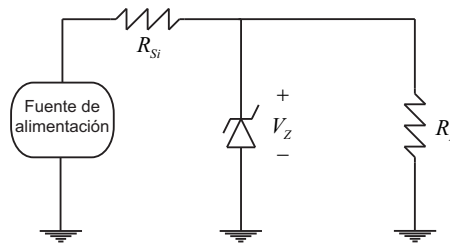


Figura 4.8.: Esquema del regulador de tensión.

Por otra parte, si la corriente inversa es mayor a I_{ZM} , este se quemará debido al exceso en la disipación de potencia, por tal motivo es necesario utilizar un resistor limitador de corriente (R_s).

4.4.3. Coeficiente de temperatura.

De acuerdo a la tensión de ruptura del diodo zener y la corriente utilizada, se presenta un cambio en la "tensión regulada" debido al incremento en la temperatura por cada grado centígrado ($mV/^\circ C$), este efecto se conoce como coeficiente de temperatura.

Estos cambios de tensión pueden ser decrementos o incrementos y están relacionados con la tensión de ruptura. Para los diodos zener con tensión de ruptura menor a $4[V]$, el coeficiente es negativo, por otra parte, si la tensión de ruptura es mayor a $6[V]$, el coeficiente de temperatura es positivo. Entre $4[V]$ y $6[V]$ el coeficiente de temperatura cambia de negativo a positivo, de tal manera que podemos encontrar un punto de funcionamiento con coeficiente de temperatura igual a cero,[1].

Se propuso un estándar de tensión a $5[V]$, usando tecnología THT de la serie 1N4728A de Motorola Semiconductor a $1[W]$, con ayuda de las hojas de especificaciones, encontramos dos diodos zener aproximados al valor de ruptura que necesitamos: $4.7[V]$ y $5.1[V]$,[14].

Con la ayuda de la gráfica de coeficiente de temperatura zener **Figura 4.9.** ($mV/^\circ C$) contra voltaje zener (V_Z), de la hoja de especificaciones del fabricante,[13]. Aproximamos lo siguiente:

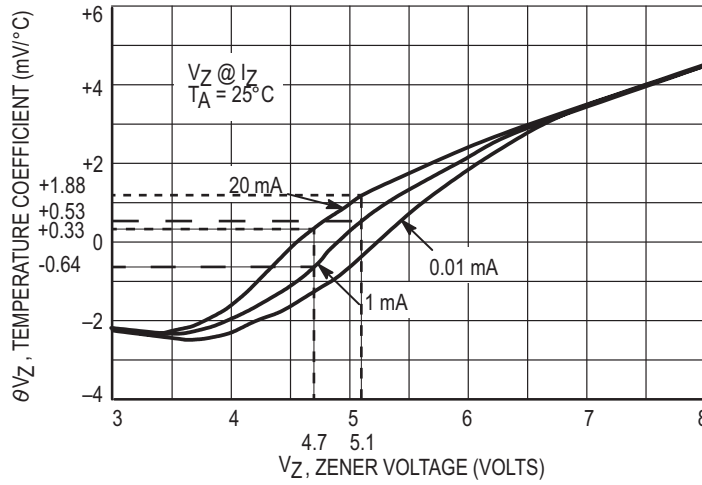


Figura 4.9.: Gráfica de θV_Z vs V_Z

Tensión zener	Corriente zener	Coficiente de temperatura
4.7[V]	1[mA]	-0.638[mV/°C]
5.1[V]	1[mA]	+0.53[mV/°C]
4.7[V]	20[mA]	+0.329[mV/°C]
5.1[V]	20[mA]	+1.188[mV/°C]

Tabla 4.1

Si en cierta situación ocupamos una corriente sobre el diodo zener de 20[mA] y el diodo de 5.1[V] de ruptura, el coeficiente de temperatura es de +1.188[mV/°C]. Si la tensión aumenta a más de 5.5[V], nuestro estándar de tensión no se cumple. Por lo tanto la tolerancia máxima será de:

$$TolMax = 5.5[V] - 5.1[V] = 0.4[V] = 400[mV]$$

De tal forma que la temperatura debería de aumentar

$$Temp = 400[mV]/1.188[mV/^\circ C] = 336.7[^\circ C]$$

Para alcanzar la tensión límite máxima.

Por otra parte, si trabajamos con una corriente sobre el diodo zener de 1[mA] y el diodo de 4.7[V] de ruptura, el coeficiente de temperatura será de -0.638[mV/°C]. Si la tensión baja a menos de 4.5[V], nuestro estándar no se cumple. Por lo tanto la tolerancia mínima será de:

4.5 Generación de entradas lógicas ($V_{I/O}$).

$$Tol_{min} = 4.7[V] - 4.5[V] = 0.2[V] = 200[mV]$$

De tal manera que la temperatura debería de aumentar

$$Temp = 200[mV]/0.638[mV/^{\circ}C] = 313.48^{\circ}C$$

Para alcanzar la tensión límite mínima.

Los casos anteriores serían los peores, puesto que otras situaciones como se muestra en la **Tabla 4.2**, para el caso A el coeficiente es positivo y nos resulta conveniente que se incremente la temperatura a fin de llegar a la polarización deseada, mientras que en el caso B el coeficiente es tan pequeño que debe de aumentar demasiado la temperatura para alcanzar la tensión límite.

	Caso Máximo	Caso Mínimo	Caso A	Caso B
Tensión zener	5.1[V]	4.7[V]	4.7[V]	5.1[V]
Corriente zener	20[mA]	1[mA]	20[mA]	1[mA]
Tolerancia	400[mV]	200[mV]	200[mV]	400[mV]
Incremento de temperatura	336.7[$^{\circ}C$]	313.48[$^{\circ}C$]	607.9[$^{\circ}C$]	754.71[$^{\circ}C$]

Tabla 4.2

Los datos anteriores nos dan una idea sobre cual es el máximo incremento de temperatura que se puede tolerar, antes de que nuestro estándar de tensión regulada se desajuste en los diodos zener a las tensiones de ruptura que fija el fabricante.

4.5. Generación de entradas lógicas ($V_{I/O}$).

4.5.1. Metodología de trabajo.

A partir de la potencia real que puede disipar el elemento (P_{Lreal}) y un factor de seguridad (fs), se obtiene la potencia máxima con la que trabajaremos (P_{LMAX}). Usando el valor de tensión estándar al que queremos regular el circuito (V_{STD}) y la potencia máxima (P_{LMAX}), se propone el valor de corriente máxima que circulará por la carga (I_{LMAX}). A partir de los datos anteriores, analizando mallas y nodos por leyes de Kirchhoff y usando especificaciones del circuito regulador zener de tensión, obtenemos el valor mínimo de la corriente inversa en el diodo zener (I_{Zmin}), la corriente ideal en el resistor limitador ideal (I_{Si}), la resistencia del resistor limitador ideal (R_{Si}) y la potencia que debe disipar (P_{Si}). Se propone un caso de corriente en la carga igual a cero ($I_L = 0$), con la finalidad de obtener el valor máximo de corriente inversa en el diodo zener (I_{ZMAX}) y su potencia máxima (P_{ZMAX}).

A partir de la corriente mínima (I_{Zmin}) y máxima (I_{ZMAX}) inversa sobre el diodo zener, la tensión estándar (V_{STD}) y la gráfica "electrical characteristic curves" de las hojas

de especificaciones del fabricante mostradas en el [Curvas características eléctricas](#); se seleccionará la tensión de ruptura del diodo zener que se ocupará.

Selección del valor de ruptura zener y del resistor limitador, en función de la corriente sobre la carga.

Todos los elementos de cualquier circuito tienen especificaciones de funcionamiento (máxima tensión, máxima corriente, máxima disipación de potencia, etc.). Las cuales limitan su funcionamiento a esas condiciones para evitar daños. Concretamente en el caso de los resistores: circuito abierto o daño térmico (cambiar de forma permanente el valor de la resistencia); en el caso del diodo degradar sus propiedades o su destrucción.

Por tal motivo se usan los dispositivos a una fracción de su máxima especificación, lo cual garantizará que aun utilizando los elementos en cualquier condición de trabajo, no se dañarán. A esta fracción se le conoce como factor de seguridad (fs).

En el caso del PIC16F887, el fabricante nos indica que la corriente que puede entrar o salir de cualquier pin ($I_{I/Ofab}$) no debe ser mayor a $25[mA]$, [10]. Con esta corriente y nuestra tensión estándar ($V_{STD} = 5[V]$), la potencia que se tendrá en cada puerto será de:

$$P_{I/Oreal} = (V_{STD}) (I_{I/Ofab}) = (5[V])(25[mA]) = 125[mW] = 1/8[W]$$

Utilizando un factor de seguridad de 2 ($fs = 1/2$), la potencia con la que se trabajará será:

$$P_{I/OMAX} = (fs) (P_{I/Oreal}) = (1/2) (1/8[W]) = 1/16[mW] = 62.5[mW]$$

Por lo tanto la corriente máxima que puede fluir por la carga ($R_{I/O}$) será de:

$$I_{I/Oreal} = \frac{P_{I/OMAX}}{V_{STD}} = \frac{62.5[mW]}{5[V]} = 12.5[mA]$$

Con base en este dato, proponemos que la máxima corriente para generar las entradas lógicas al PIC16F887 sea de:

$$I_{I/OMAX} = 10[mA]$$

Con los datos anteriores calcularemos el valor de la resistencia del resistor limitador de corriente ideal (R_{Si}) en función de la corriente propuesta para el circuito ($I_{I/OMAX}$) en la [Figura 4.10](#).

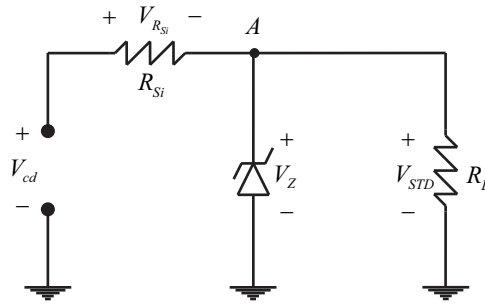


Figura 4.10.: Tensiones sobre elementos del circuito.

El valor mínimo de corriente que circulará por el diodo zener (I_{Zmin}) es igual al 10% de la corriente máxima en la carga ($I_{I/O MAX}$) [2].

$$I_{Z\text{mín}} = (10\%) (I_{I/O MAX}) = \frac{I_{I/O MAX}}{10} = \frac{10[mA]}{10} = 1[mA]$$

Por la ley de Kirchoff de corrientes tenemos en el nodo "A" de la [Figura 4.11](#) que:

$$I_{Si} - I_Z - I_{I/O} = 0$$

Por lo tanto:

$$I_{Si} = I_Z + I_{I/O}$$

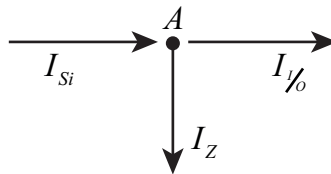


Figura 4.11.: Corrientes del nodo "A".

La corriente mínima en el diodo zener (I_{Zmin}) ocurre cuando es máxima la corriente en la carga ($I_{I/O MAX}$):

$$I_{Si} = I_{Z\text{mín}} + I_{I/O MAX}$$

$$I_{Si} = 1[mA] + 10[mA] = 11[mA]$$

Diseño del módulo eléctrico del estimulador acústico.

Siendo esta la corriente ideal (I_{Si}) que circulará sobre el resistor limitador ideal (R_{Si}).

Por otra parte, por la ley de Kirchhoff de tensión tenemos en la segunda malla que, el diodo zener se encuentra en paralelo con la carga, por lo que la tensión será la misma.

$$V_Z - V_{STD} = 0$$

Por lo tanto:

$$V_Z = V_{STD}$$

Del mismo modo, se aplica la ley de Kirchhoff de tensión en la primera malla, **Figura 4.12:**

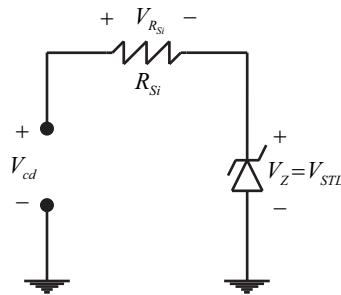


Figura 4.12.: Ley de Kirchhoff de tensión sobre la primera malla del circuito.

$$V_{cd} - V_{R_{Si}} - V_{STD} = 0$$

Despejando la tensión del resistor limitador de corriente ideal ($V_{R_{Si}}$)

$$V_{R_{Si}} = V_{cd} - V_{STD}$$

Aplicando ley de Ohm y estableciendo la ecuación en función de la resistencia del resistor limitador de corriente ideal (R_{Si})

$$I_{Si} \cdot R_{Si} = V_{cd} - V_{STD}$$

$$R_{Si} = \frac{V_{cd} - V_{STD}}{I_{Si}}$$

El valor de la resistencia del resistor limitador de corriente ideal es:

$$R_{Si} = \frac{28[V] - 5[V]}{11[mA]} = \frac{23[V]}{11[mA]} = 2090.909[\Omega] = 2.091[k\Omega]$$

La potencia que deberá ser capaz de disipar es de:

$$P_{Si} = (V_{cd} - V_{STD}) (I_{Si}) = (23[V]) (11[mA]) = 0.253[W] = 253[mW]$$

Con un factor de seguridad de 2, ocuparemos un elemento a $\frac{1}{2}[W] = 500[mW]$; con uno de 4, ocuparemos uno de $1[W]$.

Si por algún motivo la carga se desconecta (resistencia infinita $R_{I/O} \approx \infty$ en caso teórico), la corriente que fluye en la carga será igual a cero ($I_{I/O} = 0$) y la corriente en el diodo zener será la máxima (I_{ZMAX}):

$$I_{Si} = I_{ZMAX} + 0[mA] = 11[mA]$$

$$I_{ZMAX} = 11[mA]$$

El diodo zener tendrá que disipar una potencia máxima de:

$$P_{ZMAX} = (V_{STD}) (I_{ZMAX}) = (5[V])(11[mA]) = 0.055[W] = 55[mW]$$

Con un factor de seguridad de 2, ocuparemos un elemento a $\frac{1}{8}[W] = 125[mW]$; con uno de 4, ocuparemos uno de $\frac{1}{4}[W] = 250[mW]$.

La corriente inversa del diodo zener puede variar entre $1[mA]$ y $11[mA]$. Dependiendo de la tensión de ruptura de los diodos (V_Z) y la corriente sobre él (I_Z), la tensión regulada sobre la carga ($V_{I/O}$) varía unas cuantas milésimas de volt con respecto a la tensión estándar (V_{STD}), esto lo podemos ver en las hojas de especificaciones donde se muestran las curvas características eléctricas de cada diodo zener en el [Curvas características eléctricas](#).

Usando tecnología SMT de la serie PTZ5.1B de ROHM CO., a $1000[mW]$, con ayuda de la gráfica [11], de las hojas de especificaciones del fabricante aproximamos a la [Tabla 4.3](#).

Tensión de ruptura	Tensión regulada ($V_{I/O}$) con $I_{Zmin} = 1[mA]$	Tensión regulada ($V_{I/O}$) con $I_{ZMAX} = 11[mA]$
$V_Z = 4.7[V]$	3.778[V]	4.579[V]
$V_Z = 5.1[V]$	4.556[V]	5.189[V]
$V_Z = 5.6[V]$	5.611[V]	5.833[V]
$V_Z = 6.2[V]$	6.5[V]	6.5[V]

[Tabla 4.3](#)

De las tablas anteriores concluimos que:

El diodo zener que entra en el rango de tensión estándar que propusimos de $5[V] \pm 10\%$, a la corriente máxima y mínima calculadas es el de $V_Z = 5.1[V]$ de ruptura.

Para $1[mA]$ tenemos una diferencia con respecto al valor ideal de:

$$|5[V] - 4.556[V]| = 0.444[V]$$

Para $11[mA]$ tenemos una diferencia con respecto al valor ideal de:

$$|5[V] - 5.189[V]| = 0.189[V]$$

Corrientes, tensiones y potencias del sistema en función del valor de los resistores.

Hasta el punto anterior, los valores que obtuvimos para las resistencias, tensiones y potencias, son ideales, ya que están en función de la corriente propuesta. No obstante, tendremos que comprar valores de los resistores comerciales aproximados a los ideales, en consecuencia, los valores de las corrientes sobre los elementos al igual que las tensiones y las potencias, variarán, ya que estarán en función de las resistencias.

Si bien, los nuevos valores de los resistores tendrán efecto sobre el sistema, las resistencias ideales deben de generar los datos ideales, por este motivo desarrollaremos las ecuaciones en función de las resistencias ideales, posteriormente con valores comerciales, comparar datos y discriminar los mejores resultados.

1. Cálculo de corriente mínima y máxima sobre el diodo zener. Para calcular la corriente sobre el resistor limitador, haremos uso de la tensión estándar que habíamos definido como $V_{STD} = 5[V]$, obteniendo:

$$I_{Si} = \frac{V_{cd} - V_{STD}}{R_{Si}} = \frac{28[V] - 5[V]}{2090.909[\Omega]} = 11[mA]$$

Por consiguiente, la potencia en el resistor será de:

$$P_{Si} = (V_{cd} - V_{STD}) (I_{Si}) = (23[V]) (11[mA]) = 0.253[W] = 253[mW]$$

Al solo conocer el valor de la corriente en el resistor limitador, tenemos que resolver el sistema lineal de ecuaciones, a fin de encontrar las demás corrientes. El método que utilizaremos es el de sustitución.

4.5 Generación de entradas lógicas ($V_{I/O}$).

De las especificaciones del circuito sabemos que la corriente mínima inversa sobre el diodo zener (I_{Zmin}) es:

$$I_{Zmin} = \frac{I_{I/O MAX}}{10}$$

Despejando el valor máximo de corriente ($I_{I/O MAX}$) sobre la carga:

$$I_{I/O MAX} = 10 \cdot I_{Zmin}$$

Sustituyendo en la ecuación que obtuvimos por leyes de corrientes de Kirchoff sobre el nodo "A" **Figura 4.11**:

$$I_{Si} = I_{Zmin} + I_{I/O MAX}$$

Tenemos:

$$I_{Si} = I_{Zmin} + 10 \cdot I_{Zmin} = 11 \cdot I_{Zmin}$$

Dejando la corriente mínima inversa (I_{Zmin}) en función de la corriente sobre el resistor limitador (I_{Si}), obtenemos el valor:

$$I_{Zmin} = \frac{I_{Si}}{11} = \frac{11[mA]}{11} = 1[mA]$$

Por lo tanto, la corriente máxima sobre la carga es:

$$I_{I/O MAX} = 10 \cdot I_{Zmin} = 10 \cdot 1[mA] = 10[mA]$$

Sabemos que la corriente máxima inversa del diodo zener (I_{ZMAX}) se presentará cuando la carga se desconecta ($I_{I/O} = 0$), de tal manera que será igual a la corriente sobre el resistor limitador (I_{Si}).

$$I_{ZMAX} = I_{Si} = 11[mA]$$

De las curvas características eléctricas **Curvas características eléctricas**, sabemos que con una corriente inversa en el diodo zener de $I_Z = 11[mA]$, la tensión regulada en la carga es $V_{I/O} = 5.189[V]$. La potencia máxima del diodo zener será:

$$P_{ZMAX} = (V_{STD})(I_{ZMAX}) = (5.189[V])(11[mA]) = 0.055[W] = 57.08[mW]$$

2. Cálculo de corriente en el resistor limitador y en la carga, dentro de la zona de ruptura del diodo zener.

De la misma manera, sabemos que con una corriente inversa en el diodo zener de $I_Z = 1[mA]$, la tensión regulada en la carga es $V_{I/O} = 4.556[V]$. Estos cálculos son importantes, ya que nos muestran cual será la corriente que se tendrá en la carga, en el caso que la impedancia de entrada del dispositivo sea tan pequeña que demande la máxima corriente para la cual está diseñado. Mientras mas impedancia tenga el dispositivo, la corriente sobre la carga disminuirá y la corriente inversa del diodo se incrementará, por lo tanto la tensión en la carga tenderá a la tensión estándar.

La corriente sobre el resistor limitador es:

$$I_{Sau} = \frac{V_{cd} - V_{I/O}}{R_{Si}} = \frac{28[V] - 4.556[V]}{2090.909[\Omega]} = 11.21[mA]$$

Y la potencia que tendrá que disipar es:

$$P_{Sau} = (V_{cd} - V_{I/O}) (I_{Sau}) = (23.444[V]) (11.21[mA]) = 262.862[mW]$$

La corriente sobre la carga será:

$$I_{I/O} = I_{Sau} - I_Z = 11.21[mA] - 1[mA] = 10.21[mA]$$

La potencia que deberá ser capaz de disipar es de:

$$P_{R_{Fi}} = (V_{I/O}) (I_{I/O}) = (4.556[V]) (10.21[mA]) = 46.53[mW]$$

4.5.2. Tablas de resultados.

A modo de resumen, en la [Tabla 4.4](#) se muestra el valor del elemento, la tensión sobre él, sus corrientes (máxima y mínima) y su potencia.

Entradas lógicas ($V_{I/O}$) ideales		
$V_{STD} = 5[V]$; $fs = 1/2$		
R. limitadora	Diodo zener $V_Z = 5.1[V]$	Carga
$R_{Si} = 2090.909[\Omega]$	$I_{Zmin} = 1[mA]$	$I_{Sau} = 11.21[mA]$
$I_{Si} = 11[mA]$	$I_{ZMAX} = 11[mA]$	$P_{Sau} = 262.86[mW]$
$P_{Si} = 253[mW]$	$P_{ZMAX} = 57.1[mW]$	$V_{I/Omin} = 4.556[V]$
$P_{Si2} = \frac{1}{2}[W]$	$P_{Z2} = \frac{1}{8}[W]$	$I_{I/OMAX} = 10.21[mA]$
$P_{Si4} = 1[W]$	$P_{Z4} = \frac{1}{4}[W]$	$P_{I/OMAX} = 46.53[mW]$

Tabla 4.4

4.5 Generación de entradas lógicas ($V_{I/O}$).

Ahora bien, se presentan los cálculos que se hicieron entre dos resistores comerciales diferentes. Uno con valor superior al ideal, y otro con valor inferior.

Este procedimiento se realiza dos veces, debido a que los resistores SMT los podemos encontrar con una variación de valor menor (tolerancia), comparado con los resistores THT.

Los valores comerciales para THT: $R_{S1} = 2.2[k\Omega]$ [Tabla 4.5](#), $R_{S2} = 1.8[k\Omega]$ [Tabla 4.6](#)

Entradas lógicas ($V_{I/O}$)		
R. limitadora 1	Diodo zener	Carga
$R_{S1} = 2200[\Omega]$	$V_Z = 5.1[V]$	$I_{Sau} = 10.659[mA]$
$I_S = 10.455[mA]$	$I_{Zmin} = 0.950[mA]$	$V_{I/Omin} = 4.55[V]$
$P_S = 240.45[mW]$	$I_{ZMAX} = 10.455[mA]$	$I_{I/OMAX} = 9.709[mA]$
	$P_{ZMAX} = 54.23[mW]$	$P_{I/OMAX} = 44.174[mW]$

[Tabla 4.5](#)

Entradas lógicas ($V_{I/O}$)		
R. limitadora 2	Diodo zener	Carga
$R_{S2} = 1800[\Omega]$	$V_Z = 5.1[V]$	$I_{Sau} = 13.024[mA]$
$I_S = 12.778[mA]$	$I_{Zmin} = 1.162[mA]$	$V_{I/Omin} = 4.557[V]$
$P_S = 293.89[mW]$	$I_{ZMAX} = 12.778[mA]$	$I_{I/OMAX} = 11.862[mA]$
	$P_{ZMAX} = 66.35[mW]$	$P_{I/OMAX} = 54.056[mW]$

Tabla 4.6

Los valores comerciales para SMT: $R_{S1} = 2.15[k\Omega]$ [Tabla 4.7](#), $R_{S2} = 2[k\Omega]$ [Tabla 4.8](#)

Entradas lógicas ($V_{I/O}$)		
R. limitadora 1	Diodo zener	Carga
$R_{S1} = 2150[\Omega]$	$V_Z = 5.1[V]$	$I_{Sau} = 10.907[mA]$
$I_S = 10.698mA$	$I_{Zmin} = 0.973[mA]$	$V_{I/Omin} = 4.55[V]$
$P_S = 246.047[mW]$	$I_{ZMAX} = 10.698[mA]$	$I_{I/OMAX} = 9.934[mA]$
	$P_{ZMAX} = 55.505[mW]$	$P_{I/OMAX} = 45.2[mW]$

Tabla 4.7

Entradas lógicas ($V_{I/O}$)		
R. limitadora 2	Diodo zener	Carga
$R_{S2} = 2000[\Omega]$	$V_Z = 5.1[V]$	$I_{Sau} = 11.722[mA]$
$I_S = 11.5[mA]$	$I_{Zmin} = 1.045[mA]$	$V_{I/Omin} = 4.557[V]$
$P_S = 264.5[mW]$	$I_{ZMAX} = 11.5[mA]$	$I_{I/OMAX} = 10.676[mA]$
	$P_{ZMAX} = 59.685[mW]$	$P_{I/OMAX} = 48.64[mW]$

Tabla 4.8

4.5.3. Elección de los elementos.

Para THT, el mejor valor de resistor limitador de corriente es $R_{S1} = 2.2[k\Omega]$. Los porcentajes de error de exactitud son:

Para la corriente sobre el resistor limitador:

$$error I_S = \left| \frac{11[mA] - 10.455[mA]}{11[mA]} \right| \times 100 = 4.959\%$$

La corriente mínima inversa del zener:

$$errorI_{Zmin} = \left| \frac{1[mA] - 0.950[mA]}{10[mA]} \right| \times 100 = 4.958 \%$$

La corriente máxima inversa del zener:

$$errorI_{ZMAX} = \left| \frac{11[mA] - 10.455[mA]}{10[mA]} \right| \times 100 = 4.958 \%$$

La corriente máxima sobre la carga:

$$errorI_{I/O MAX} = \left| \frac{10.21[mA] - 9.709[mA]}{9.135[mA]} \right| \times 100 = 4.932 \%$$

Para SMT, el mejor valor de resistor limitador de corriente es $R_{S1} = 2.15[k\Omega]$. Los porcentajes de error de exactitud se muestran en la Tabla A.1.

Los datos anteriores comprueban que al usar elementos de SMT el error es menor, en comparación a los de THT.

$errorI_S$	$errorI_{Zmin}$	$errorI_{ZMAX}$	$errorI_{I/O MAX}$
2.748 %	2.748 %	2.748 %	2.748 %

Tabla A.1

4.6. Polarización PIC16F887 (V_{DD})

Usando la metodología para calcular el valor del diodo zener y la resistencia del resistor limitador, para la generación de las entradas lógicas ($V_{I/O}$), se calcularon los valores de los diferentes dispositivos para lograr la correcta regulación de tensión y corriente para las diferentes partes del sistema.

4.6.1. Tablas de resultados.

Observamos en la [Tabla 4.9](#) que la potencia que debe disipar el resistor limitador sobrepasa $1[W]$, y al agregar el factor de seguridad, los elementos que debemos de comprar deben de ser mínimo de $2[W]$ como se muestra en la columna "R. limitador A" de la [Tabla 4.9](#). Para evitar daños térmicos se optó por usar un arreglo en paralelo [Figura 4.13](#) como se aprecia en la columna "R. limitador B" de la [Tabla 4.9](#) para repartir la corriente y la potencia a disipar.

PIC16F887 (V_{DD}) ideal			
$V_{STD} = 5[V]$; $fs = 1/5$			
R. limitador A	R. limitador B	D. zener $V_Z = 5.1[V]$	Carga
$R_{Si} = 522.7273[\Omega]$	$R_{SiC/U} = 1045.4546[\Omega]$	$I_{Zmin} = 4[mA]$	$I_{Sau} = 44.067[mA]$
$I_{Si} = 44[mA]$	$I_{SiC/U} = 22[mA]$	$I_{ZMAX} = 44[mA]$	$P_{Sau} = 1015[mW]$
$P_{Si} = 1012[mW]$	$P_{SiC/U} = 506[mW]$	$P_{ZMAX} = 235.6[mW]$	$V_{DDmin} = 4.965[V]$
$P_{Si2} = 2[W]$	$P_{Si2C/U} = 1[W]$	$P_{Z2} = \frac{1}{2}[W]$	$I_{DDMAX} = 40[mA]$
$P_{Si4} = 4[W]$	$P_{Si4C/U} = 2[W]$	$P_{Z4} = 1[W]$	$P_{DDMAX} = 199[mW]$

Tabla 4.9

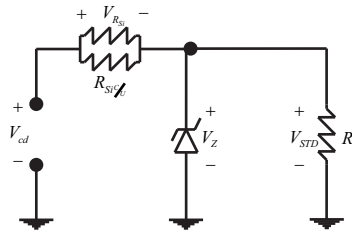


Figura 4.13.: Arreglo en paralelo equivalente al resistor limitador de corriente.

Los resistores en SMT para potencias mayores a $1[W]$ tienen un alto costo, por lo que optamos por usar THT.

Los valores comerciales para THT: $R_{S1} = 1.2[k\Omega]$, $R_{S2} = 1[k\Omega]$ **Tabla 4.10**

PIC16F887 (V_{DD}) Paralelo		
R. limitadora 2	Diodo zener	Carga
$R_{Sparalelo} = 500[\Omega]$	$V_Z = 5.1[V]$	$I_{Sau} = 45.903[mA]$
$I_S = 46[mA]$	$I_{Zmin} = 4.182[mA]$	$V_{DDmin} = 5.0484[V]$
$P_S = 1058[mW]$	$I_{ZMAX} = 46[mA]$	$I_{DDMAX} = 41.721[mA]$
	$P_{ZMAX} = 246.51[mW]$	$P_{DDMAX} = 210.626[mW]$

Tabla 4.10

4.6.2. Elección de los elementos.

Para THT, el mejor valor de resistor limitador de corriente es $R_S = 500[\Omega]$ (dos resistores de $R_{S2} = 1[k\Omega]$). Los porcentajes de error de exactitud se muestran en la Tabla A.2:

$error I_S$	$error I_{Zmin}$	$error I_{ZMAX}$	$error I_{I/O MAX}$
4.545 %	4.545 %	4.545 %	4.123 %

Tabla A.2

5. Análisis de la arquitectura del generador de ondas programable.

La arquitectura del generador de ondas está básicamente descrito en la [Figura 5.1](#) y se puede observar cada una de las etapas de acople entre cada sub-sistema.

En este capítulo nos enfocaremos principalmente al microcontrolador y el procesamiento que realiza, para generar los diferentes tipos de ondas.

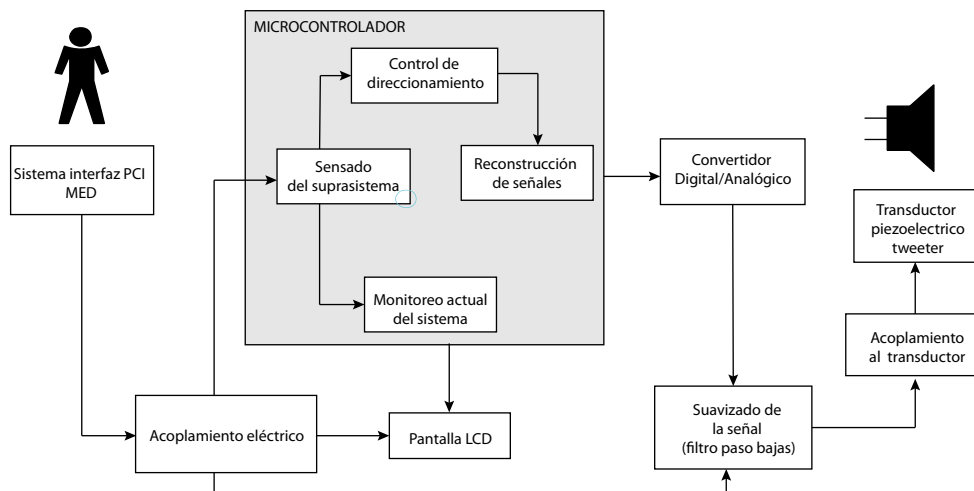


Figura 5.1.: Diagrama a bloques del generador de ondas propuesto.

5.1. Elección para la configuración física del dispositivo.

Para la toma de elección sobre la generación de señales se tomaron en cuenta las características mencionadas en el [Capítulo 3](#), descartando de esta manera los dispositivos analógicos para la obtención de la señal, ya que sería un dispositivo muy voluminoso y que es un punto importante en la implementación y acoplamiento final al SI-PCI MED ([Figura 2.2](#)).

Las formas de generar las señales a partir de un DSP¹ y/o con un DDS² resultó ser una opción muy buena debido a que supera bastante los requerimientos de la aplicación, pero por otro lado el costo del sistema se incrementa demasiado debido a que los DSP's y los DDS's son circuitos integrados caros para la aplicación.

El diseño que se escogió está basado en un microcontrolador el cual hará uso de una tabla de consulta "*lookup*" para generar las señales y que se comunicará con una etapa de conversión analógica/digital para la reconstrucción de la señal continua.

5.2. Elección del microcontrolador.

Esta etapa fue muy importante para el desarrollo e implementación del dispositivo propuesto, debido a que el microcontrolador será la base del sistema y controlará tanto las interfaces de salida como las señales de entrada.

Otro punto importante para este proyecto fue la reducción de costos como se ha mencionado anteriormente, por lo que no debió ser un microcontrolador muy caro y con el mejor desempeño posible cubriendo todos los requerimientos. De tal manera que se eligieron dos microcontroladores de uso común en la industria y fáciles de conseguir en el país.

Los microcontroladores propuestos fueron:

- PIC16F887,[10].
- ATMega328P,[20].

Ambos dispositivos son microcontroladores de 8 bits que soportan una frecuencia máxima de un cristal de 20 MHz como oscilador externo, que para fines prácticos son las características que más nos interesaran en el desarrollo del sistema.

Otra característica importante pero no crítica, es la cantidad de memoria flash que contienen ambos microcontroladores. Es importante debido a que todo el código que es necesario para cumplir con los requisitos de la aplicación es bastante amplio, y que como más adelante se mencionará, un ciclo de cada señal será almacenado en memoria para poder generar las señales a partir de estas.

5.2.1. Elección de lenguaje de programación.

En este punto es conveniente mencionar que tipo de compiladores se usaron y el porqué de la elección del microcontrolador.

La primera opción de compilador que se contempló usar, fue:

¹*Digital Signal Processor*

²*Digital Direct Synthesizer*

- MikroC,[21]. Ya que las exigencias del programa que se desarrolló, es básico y no se necesita una optimización crítica para el buen funcionamiento del sistema. Esta herramienta necesita de una licencia para su uso, esto podría ser una desventaja para esta opción de programación.
- Arduino,[22]. Esta herramienta es de dominio público, y se tomo en cuenta su uso, debido a que en la alternativa pasada era necesario comprar una licencia para usarla. Otra ventaja sobre esta herramienta es que es una placa de desarrollo que hace uso del microcontrolador ATmega328P que una vez que se ha programado, este puede ser desmontado de la placa y ser usado directamente en tu aplicación.
- Ensamblador. Esta opción se tomó en cuenta al final de la realización del programa hecho en lenguaje C, con el fin de ver si el programa hecho anteriormente necesitaría alguna optimización para migrar el código en C a lenguaje máquina (ensamblador). Pero como la aplicación no lo requería, puesto que tiene buena respuesta en tiempo a los usuarios finales, no se tomo como necesario o indispensable. Aparte de que si el proyecto se quisiera mejorar, un programa en lenguaje C, sería más fácil de entender que uno en ensamblador.

Se realizaron dos programas para la elección más adecuada entre ambos lenguajes comparando sus desempeños. Los compiladores que ser usaron para el desarrollo de estos programas fueron: MikroC y Arduino.

Estos programas serán mostrados en su totalidad en el [Códigos fuente](#).

5.2.2. Comparación de los programas realizados en diferentes lenguajes.

La prueba de comparación entre ambos lenguajes se realizó básicamente con la misma estructura. El microcontrolador responderá al cambio lógico que tenga en uno de sus puertos, de forma que se hará la función de la decodificación.

El puerto de entrada será de 4 bits de forma que se puedan obtener 16 respuestas a un puerto de salida del sistema, esto se puede entender mejor en la siguiente [Tabla 5.1](#).

Entrada A	Entrada B	Entrada C	Entrada D	Frecuencia de la señal de salida
1	1	1	1	Sistema en espera
1	1	1	0	1 [KHz]
1	1	0	1	1.5 [KHz]
1	1	0	0	2 [KHz]
1	0	1	1	2.5 [KHz]
1	0	1	0	3 [KHz]
1	0	0	1	3.5 [KHz]
1	0	0	0	4 [KHz]
0	1	1	1	5 [KHz]
0	1	1	0	6 [KHz]
0	1	0	1	8 [KHz]
0	1	0	0	10 [KHz]
0	0	1	1	12 [KHz]
0	0	1	0	15 [KHz]
0	0	0	1	18 [KHz]
0	0	0	0	20 [KHz]

[Tabla 5.1](#)

La señal que entregará cada prueba, será un tren de pulsos rectangulares con la frecuencia que se muestran en la ([Tabla 5.1](#)).

La razón de programar una señal de tren de pulsos rectangulares para hacer pruebas, se debe a que solo es necesario prender y apagar el puerto de salida en un intervalo de tiempo.

Con esto se observó la reacción de cada sistema, tanto en la salida de la señal (no exista retardos o cambios de frecuencia) como en tiempo de respuesta al cambio en las entradas. De forma que al comparar ambos códigos ya implementados en ambos prototipos, obtuvimos que el programa diseñado en C fue más eficiente en cuanto a tiempos de respuesta en el sistema y no hubo que calibrar las señales de salida como en arduino.

5.3. Generación de formas de onda.

Como se mencionó anteriormente, se decidió generar las señal de salida a partir de tablas de consulta "lookup", las cuales almacenarán los valores de las amplitudes y formas de onda de las señales,[23].

5.3.1. Cálculo y almacenamiento de tabla "lookup" para generar la señal sinusoidal.

Los valores para generar dicha tabla se obtuvieron con el uso de Matlab. Se generó un programa que calculará e imprimirá en un archivo de texto los valores de un ciclo de una señal sinusoidal, que pueda ser leída por el microcontrolador de 8 bits que se escogió.

Como se opto por la opción de usar el microcontrolador PIC16F887 con lenguaje C programado con MikroC, tenemos que el conjunto de valores más grande que usaremos como registro de almacenamiento puede contener hasta 2044 valores, siendo un dato importante, porque limitara al número de valores que se puedan generar en las señales y eso afecta la resolución de las ondas.

El procedimiento que se programó en Matlab fue el siguiente:

- Los 360° un ciclo de una señal sinusoidal son divididos entre el número máximo de valores del registro de almacenamiento para poder obtener la distancia entre muestras, que será denominada como: Δ . Esto quiere decir que:

$$\Delta = \frac{360^\circ}{2044}$$

que en radianes queda expresada por:

$$\Delta = \frac{2\pi}{2044}$$

Después de tener esto obtenemos la distancia Δ entre muestras en radianes.

$$\Delta = \frac{2\pi}{2044} = 0.00307 \text{ [rad]}$$

Una vez obtenido el valor de Δ podemos hacer uso de la siguiente ecuación para lograr obtener más parámetros que nos ayuden más adelante.

$$f = \frac{\Delta}{t \cdot N}$$

donde N es el número total de muestras que habrá en un ciclo, t es el tiempo que tardara el microcontrolador en pasar de leer una muestra a la siguiente y por último, f será la frecuencia a la que nosotros queramos obtener la señal sinusoidal.

Debido a que la única información que necesitaremos de la tabla de consulta "lookup" son los valores de las alturas de las muestras, se usará la siguiente ecuación.

$$\text{sen}\left(\frac{2\pi \cdot n}{N}\right)$$

donde n es el índice de avance de la función seno. $n = 0, 1, 2, \dots, N - 2, N - 1$

Esta ecuación nos arrojará valores de 1 a -1 durante el barrido del índice n . Gráficamente se muestra en la [Figura 5.2](#)

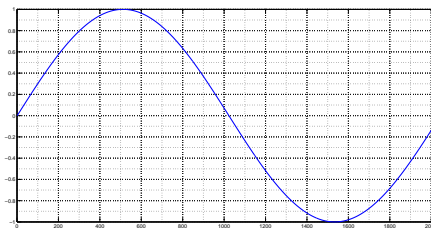


Figura 5.2.: Señal sinusoidal generada.

Estos valores se tienen que procesar para poderlos acoplar lo mejor posible al código fuente, para que el microcontrolador tenga un mejor desempeño. Esto se logró haciendo que la señal tuviera un *offset* con el fin de impedir que existan valores negativos. Sabiendo que se debe contar con el *offset*, todos los valores generados serán multiplicados por un valor constante de 128, con el fin de amplificar la señal y que los *Valores_{PP}*³ no excedan de -128 a 128. Después de obtener el resultado del producto anterior, se le sumará (*offset*) 127 para que los valores de la señal entren en el rango de 0 a 255 como se muestra en la [Figura 5.3](#).

Una vez habiendo generado los valores que estarán contenidos en la tabla de consulta "lookup", tenemos un ciclo de la señal sinusoidal a una frecuencia fija. Esta frecuencia fija esta dada por el intervalo de tiempo t y es dependiente de la frecuencia del cristal que funciona como oscilador externo del microcontrolador.

5.3.2. Modos de reloj externo.

El oscilador se puede configurar de diferentes formas para funcionar, lo que le da la posibilidad de manejar diferentes velocidades de ejecución y utilizar diferentes compo-

³Valores pico a pico

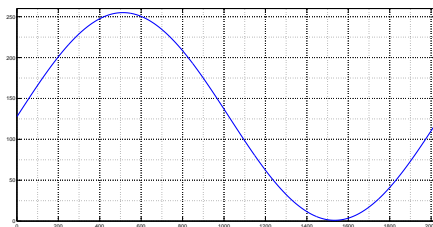


Figura 5.3.: Señal sinusoidal generada, amplificada y con *offset*.

mentos para estabilizar la frecuencia.

El modo en el que funcionará la señal de reloj, se debe escribir al principio del programa que irá en el microcontrolador.

Algunas de las configuraciones de osciladores que se pueden ocupar en un microcontrolador para que tenga una señal de reloj son:

- LP (Low power) - Es un oscilador constituido por un cristal de cuarzo de baja frecuencia, y por lo general su frecuencia de oscilación es de: 32.768 KHz.
 - XT (Xtal) - Este tipo de oscilador externo tiene un consumo medio de corriente a comparación del anterior. Las frecuencias para las que está diseñado, no superan los 8 MHz.
 - HS (High Speed) - Este modo es el que más energía consume de todos, pero es el que proporciona las frecuencias más altas, que van más allá de los 8 MHz.
- Figura 5.4**
- RC (Resistor-Capacitor) - En las aplicaciones en donde no es crítica la frecuencia de reloj, se puede ocupar esta configuración para abaratar el costo del proyecto y liberando un GPIO⁴.
 - Reloj Interno - Este modo para el PIC16F887 tiene la característica de poder usar un oscilador interno con frecuencias que van de los 31 KHz a los 8 MHz, haciendo uso de este oscilador se tiene la ventaja de no desperdiciar dos GPIO, pero la velocidad de ejecución está limitada a máximo 8 MHz, que para nuestro propósito es insuficiente.

Tomando en cuenta los modos anteriores, únicamente el HS logra alcanzar la frecuencia máxima soportada por el microcontrolador que es de 20 MHz, logrando así obtener una mayor velocidad de ejecución en el dispositivo. Por lo que es el modo seleccionado.

⁴Del inglés *General Purpose Input Output*

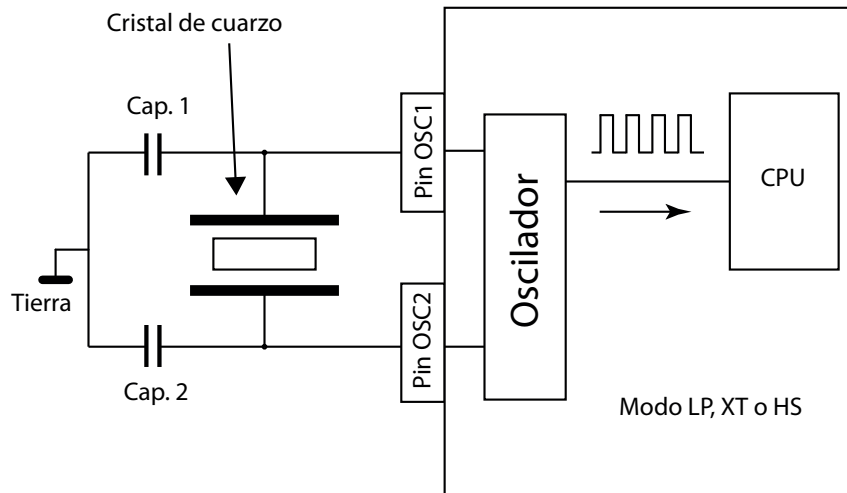


Figura 5.4.: Configuración de microcontrolador con cristal de cuarzo como oscilador externo.

5.3.3. Tiempo de ejecución para las instrucciones en el microcontrolador.

El tiempo de ejecución es semejante para la mayoría de las instrucciones, que son 4 ciclos de reloj.

Esta frecuencia está establecida por el cristal de cuarzo (Figura 5.4). Las instrucciones de salto y ramificaciones tardan 8 ciclos de reloj en ejecutarse.

Para ejemplo del cristal que se ocupó, que es un cristal de cuarzo de 20 [MHz] el tiempo de ejecución de este está dado por:

$$f = 4 \cdot \left(\frac{1}{f_{Xtal}} \right)$$

donde:

$$f = 4 \cdot \left(\frac{1}{20 [MHz]} \right) = 200 [ns]$$

Tomando en cuenta estos parámetros, podemos calcular de una forma aproximada las frecuencias de salida de las señales que se han almacenado en memoria.

Esto es debido a que el tiempo que tarda el microcontrolador en leer todos y cada uno de los valores de la tabla de consulta "lookup" (un ciclo de señal sinusoidal) será un valor fijo y lo usaremos como una constante de tiempo para poder realizar el cálculo de las demás frecuencias a partir de este.

Este valor de tiempo lo obtendremos de la siguiente manera haciendo uso de la ecuación

$$f = \frac{\Delta}{t \cdot N}$$

en donde Δ ya lo hemos obtenido (Subsección 5.3.1) y en donde N es constante siendo el número máximo de valores que puede contener el registro de almacenamiento auxiliar.

Con estos valores nos quedan dos incógnitas que son f y t . Como se dijo antes, el valor de t se obtendrá a partir de hacer la primer prueba con el dispositivo y obtener la frecuencia de salida de la señal sinusoidal cuando el microcontrolador lee todos los valores de la tabla "lookup" de uno en uno como en la Figura 5.5.

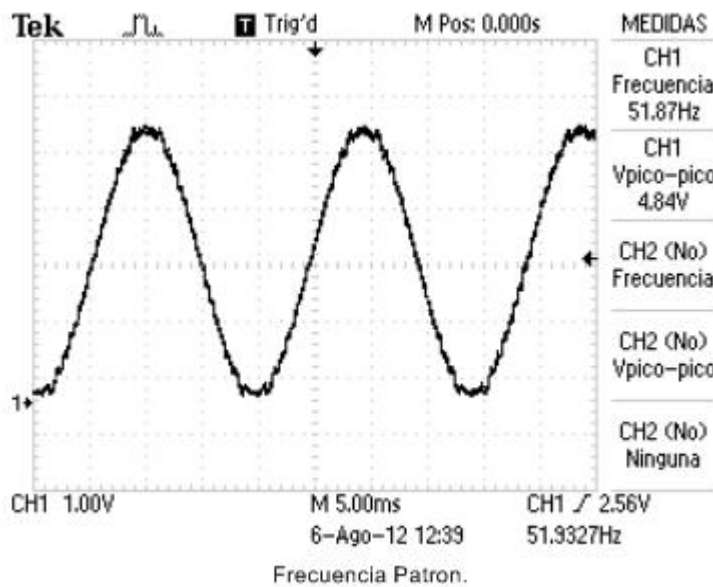


Figura 5.5.: Salto unitario - Frecuencia patrón.

Obtenemos que la frecuencia más baja y que contiene a todos los valores es de 51.87 [Hz] aproximadamente y haciendo uso de este dato podemos calcular t para todas las demás frecuencias, esto es porqué el intervalo de tiempo t es el tiempo que hay entre lecturas de muestras y por lo general.

$$t = \frac{\Delta}{f \cdot N} = \frac{0.003073}{51.87 \cdot 2044} = 29.03[nS]$$

Una vez obtenido el valor de t (tiempo de lectura entre muestra y muestra), podemos seguir con la obtención de más valores.

Las frecuencias son datos fijos que están dadas previamente por el usuario final. Lo que nos interesará obtener es el número de muestras que queramos que el microcontrolador lea en un ciclo de la señal sinusoidal, con la intención de que entre menos muestras lea este, el ciclo de la señal sinusoidal sea repetido un mayor número de veces en un mismo intervalo de tiempo. Esto quiere decir que como el ciclo de la señal será repetido mas rápido porqué lee un menor número de muestras, la frecuencia de la señal final aumentará y así obtendremos las frecuencias deseadas.

Cabe mencionar que para obtener la señal continua, debemos cumplir con el criterio de Nyquist, el cual nos dice que por lo menos nuestra frecuencia de muestreo debe ser el doble de la señal deseada. Esto quiere decir que por la longitud de nuestro registro de almacenamiento auxiliar que es 2044, no debemos superar saltos mayores a 1020 registros entre lectura y lectura de la tabla "lookup" siendo este el peor caso. Un mayor número de saltos entre muestras, no generará una señal sinusoidal, si no más bien una señal espuria y que en frecuencia tendría efecto de traslape o *aliasing*,[18].

Tomando en cuenta lo anterior, ahora obtendremos el número de saltos⁵ necesarios para reducir el tiempo de lectura de toda la tabla de consulta "lookup". Entre más grande sea el salto entre muestra y muestra leída, mayor será la frecuencia, pero conforme el salto aumente, la resolución de la señal sinusoidal decrecerá y eso es claramente razonable por el criterio de Nyquist (a menor número de muestras en una señal discreta, menor será la resolución de la señal reconstruida a partir de dicha señal discreta),[23].

Esto se puede explicar con los siguientes gráficos: [Figura 5.6](#), [Figura 5.7](#), [Figura 5.8](#), [Figura 5.9](#), [Figura 5.10](#) y [Figura 5.11](#).

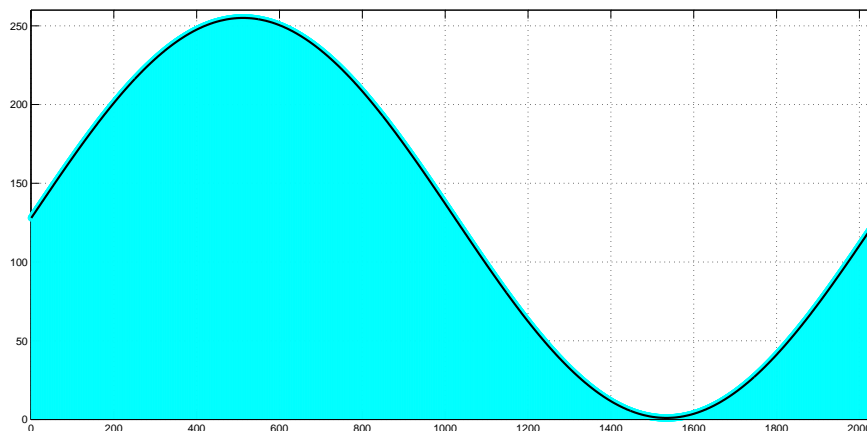


Figura 5.6.: Señal sinusoidal generada usando los 2044 valores disponibles en el registro auxiliar.

⁵Datos no leídos por el microcontrolador

En esta gráfica (Figura 5.6) se observa como el microcontrolador toma todas las muestras (señal discreta) de la tabla "lookup" (curva negra).

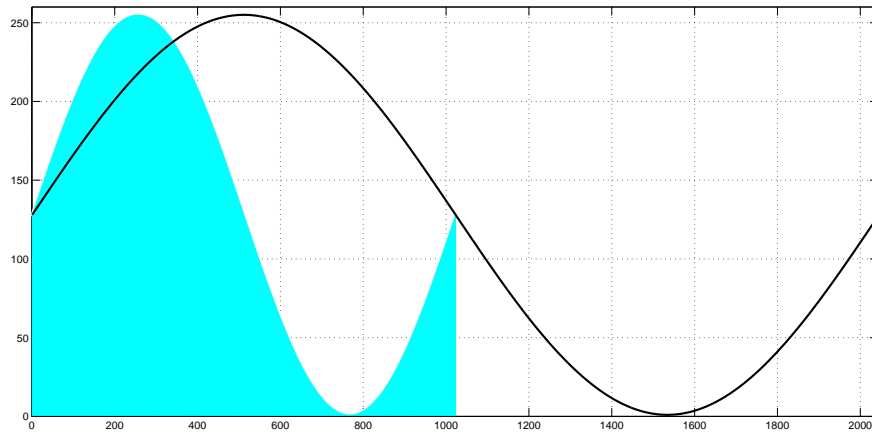


Figura 5.7.: Señal sinusoidal generada usando los 1022 valores disponibles en el registro auxiliar.

Ahora podemos observar en la Figura 5.7 que las muestras que toma el microcontrolador son menos que la cantidad total de muestras existentes en la tabla. Esto quiere decir que entre menos muestras tengamos, más rápido se repetirá el nuevo ciclo, haciendo aumentar la frecuencia de la señal.

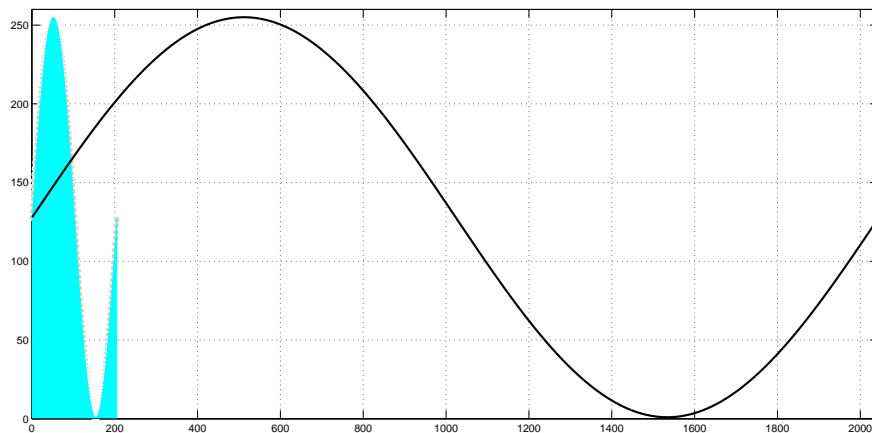


Figura 5.8.: Señal sinusoidal generada usando los 204 valores disponibles en el registro auxiliar.

En la [Figura 5.8](#) se aprecia como disminuye considerablemente el periodo de la señal de salida, si el microcontrolador toma 1 de cada 11 muestras.

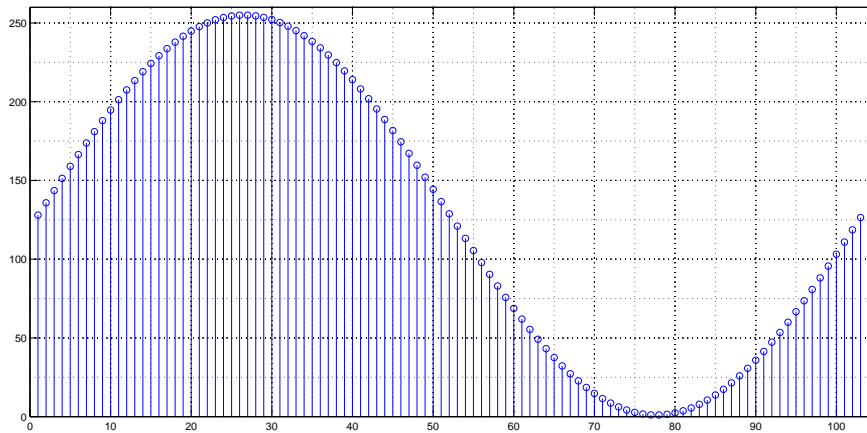


Figura 5.9.: Señal sinusoidal generada usando los 21 valores disponibles en el registro auxiliar.

Realizando un acercamiento a la señal que reconstruirá el microcontrolador ([Figura 5.9](#)) tomando 1 muestra cada 21 muestras, podemos afirmar que aún existe muy buena resolución en la señal.

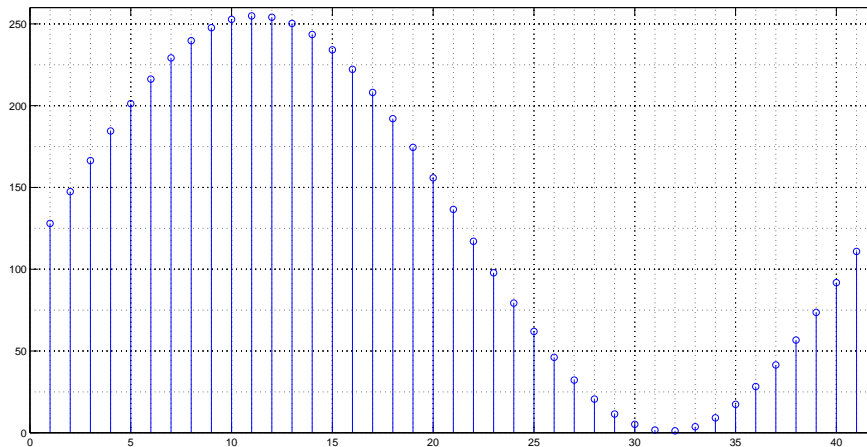


Figura 5.10.: Señal sinusoidal generada usando los 11 valores disponibles en el registro auxiliar.

Aún realizando 1 salto de 100 muestras entre cada muestra tomada para reconstruir la señal, se tiene una resolución aceptable, **Figura 5.10**.

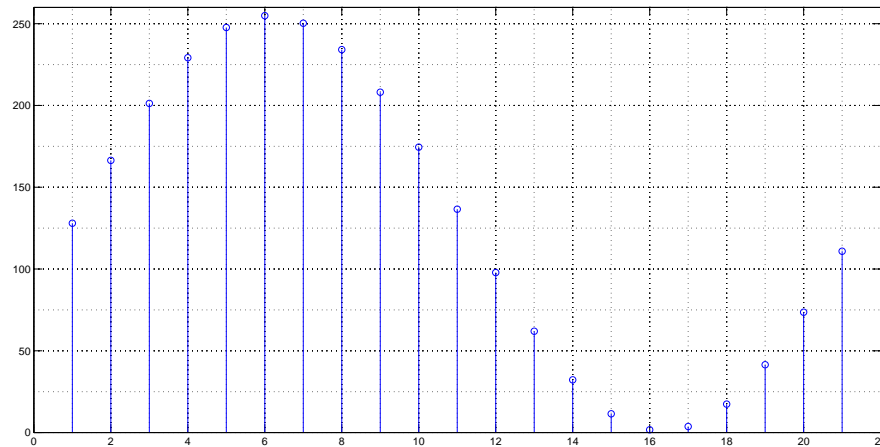


Figura 5.11.: Señal sinusoidal generada usando los 5 valores disponibles en el registro auxiliar.

En este ejemplo, **Figura 5.11** se ve la señal reconstruida a partir de tomar 1 muestra cada 1020 valores en la tabla de consulta. La resolución en la señal disminuye, pero aún mantiene la forma de un ciclo de una onda sinusoidal, lo que quiere decir que sí se cumple con el criterio de Nyquist.

5.4. Reconstrucción de la señal continua a partir de retenes.

Retén de orden cero. El retén de orden cero es un modelo matemático de la reconstrucción hecha por un convertidor digital/analógico.

Esto describe el efecto de conversión de una señal en tiempo discreto a una señal en tiempo continuo al retener cada valor de la muestra para un intervalo de muestreo.

Su modelo matemático en el dominio del tiempo esta dado por:

$$x_{ZOH}(t) = \sum_{n=0}^{\infty} x[n] \cdot \text{rect}\left(\frac{t - T/2 - nT}{T}\right)$$

donde la función $\text{rect}()$ es una función rectangular.

Retén de primer orden. El retén de primer orden es un modelo matemático para la reconstrucción de la señal hecha por un convertidor digital/analógico (Figura 5.12) y a la salida de este tiene un filtro paso bajas, con la función de realizar el suavizado de la señal, como se aprecia en la Figura 5.13.

Cuyas ecuaciones están dadas por:

$$x_d(t) = x(t)T \sum_{n=-\infty}^{\infty} \delta(t - nT)$$

$$x_{FOH}(t) = \sum_{n=-\infty}^{\infty} x(nT) \text{tri} \left(\frac{t - nT}{T} \right)$$

- Selección del circuito analógico (filtro paso bajas) para el suavizado de la señal.

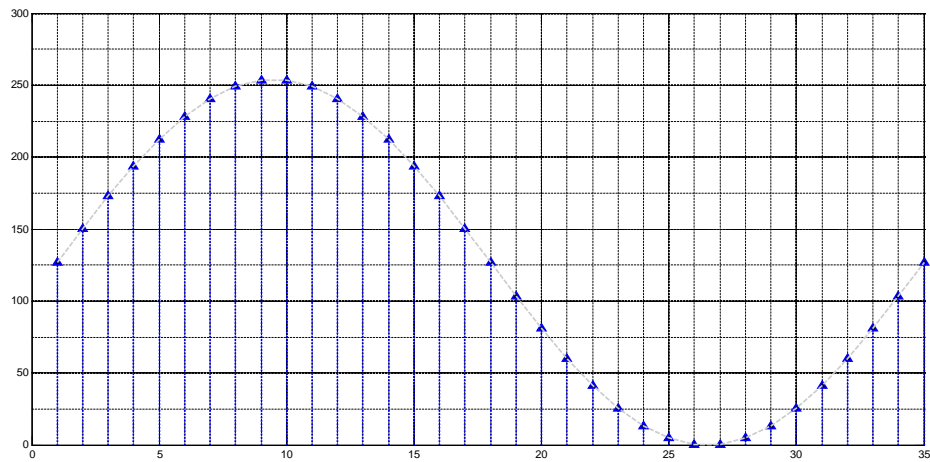


Figura 5.12.: Señal $x_d(t)$ muestreada idealmente.

Convertidor Digital-Analógico (R-2R).

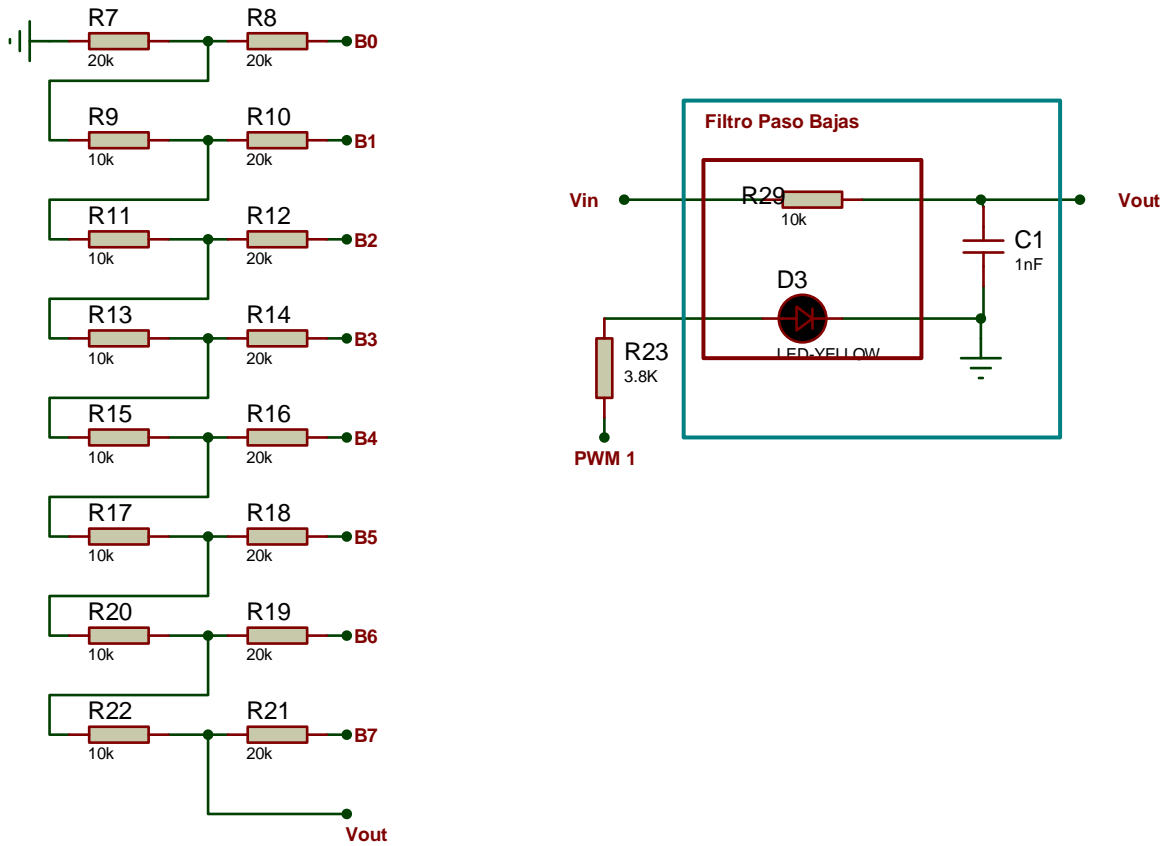


Figura 5.14.: Convertidor Digital-Analógico R-2R de 8 bits con filtro paso bajas a la salida.

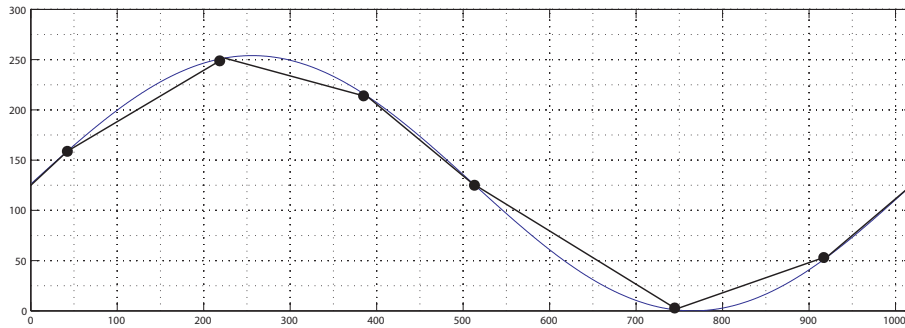


Figura 5.13.: Suavizado de la señal discreta.

Análisis de la arquitectura del generador de ondas programable.

Una red como esta logra el peso apropiado para cada bit, utilizando únicamente resistencias de 2 valores (R y $2R$),[34].

Aplicando el principio de superposición, podemos ver cual es la salida si la entrada es un 1 en el i -ésimo bit y un 0 lógico en las restantes.

Generalizando para cualquier entrada del arreglo obtenemos que:

$$V_{out} = -\frac{R_f \cdot V_{ref}}{3R} \left(\frac{1}{2^{(n-i)}} \right)$$

$$[b_{n-1}, b_{n-2}, \dots, b_0]$$

tenemos que la salida es:

$$[b_{n-1}, b_{n-2}, \dots, b_0]$$

y que se puede ver como:

$$V_{out} = -\frac{R_f \cdot V_{ref}}{3R (2^n)} \cdot \sum_{i=0}^{n-1} b_i 2^i$$

Si definimos

$$D = \sum_{i=0}^{n-1} b_i 2^i$$

(peso binario de la palabra) y $R_f = 3R^6$, por lo tanto tenemos que:

$$V_{out} = -\frac{D}{2^n} V_{ref}$$

Es importante resaltar nuevamente, que la tensión a la salida es proporcional al peso binario de la palabra de entrada.

⁶ R_f es la resistencia en el lazo de retroalimentación entre la salida del amplificador y la entrada inversora.

5.5. Configuración para el filtrado controlado por modulación de ancho de pulso.

Como se ha mencionado, la etapa del filtrado es importante por lo que implementar un filtro adaptativo analógico resultó conveniente para el desempeño y la disminución de componentes en el dispositivo propuesto.

La realización de este filtro adaptable se logró gracias al diseño propio de un dispositivo el cual designamos: "Opto-acoplador resistivo controlado por modulación de ancho de pulso" y esquemáticamente se observa su implementación en la [Figura 5.15](#) en el cuadro interior del filtro paso bajas.

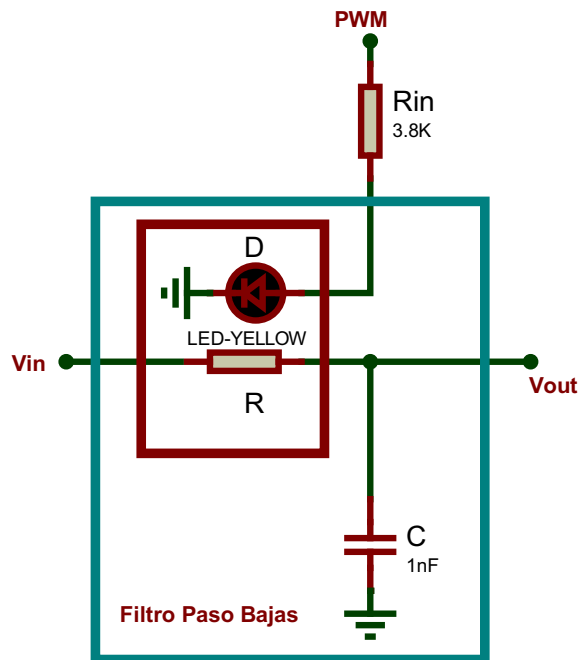


Figura 5.15.: Esquema de los opto-acopladores resistivos.

Estos dispositivos tendrán la función de variar la resistencia de un foto-resistor interno a través de un diodo emisor de luz o LED⁷, que será controlado a través de los módulos PWM del PIC16F887.

Dicho LED será de colores diferentes en cada caso, para que emitan una longitud de onda de diferentes valores,[36]. Esta longitud de onda será la que el foto-resistor necesite, para que tenga la respuesta deseada en su resistencia en cuanto a la intensidad lumínica,[37].

⁷Del inglés, *Light-Emitting Diode*.

Este dispositivo opto-acoplador resistivo, se caracterizó y obtuvieron las gráficas para conocer el rango de resistencias que obtiene en funcionamiento con un LED amarillo. Esta información se encuentra en el [Apéndice D](#).

5.6. Interfaz hombre máquina.

El dispositivo podrá mostrar la información de la señal generada en uso o en estado de espera, a través de una pantalla LCD (*Liquid Crystal Display*) 16X2,[\[12\]](#).

Esta pantalla mostrará al usuario las siguientes características de la señal:

- Frecuencia de la señal de salida.
- Tipo de onda generada (Sinusoidal, Triangular o Rectangular).

5.7. Acople con la salida acústica.

Si analizamos el acople como un diagrama a bloques ([Figura 5.16](#)), observaremos que del PIC16F887 tenemos dos puertos de salida para señales. Del primer puerto obtenemos la señal rectangular y del segundo puerto saldrán los valores de la señal triangular o sinusoidal, que son conectadas a un arreglo R-2R cuyo propósito es el de reconstruir la señal, seguido de un bloque de suavizado, que es un filtro paso bajas para eliminar la frecuencia de muestreo, seguido de un amplificador.

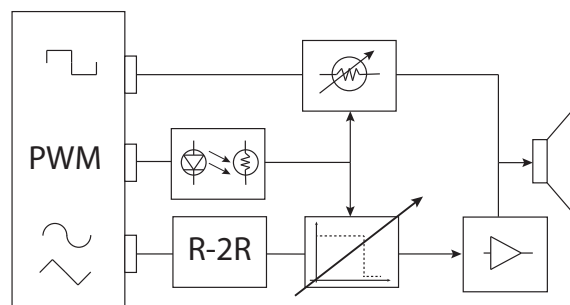


Figura 5.16.: Diagrama a bloques, acople con la salida acústica.

Esas dos terminales tienen que conectarse a una misma salida acústica, el tweeter. El bloque que ocuparemos para acoplar las dos señales es una configuración de opto-acopladores resistivos.

5.8. Tweeter piezo eléctrico.

Este transductor electro-acústico transforma la corriente eléctrica en ondas sonoras de altas frecuencias, que emite al medio. La gama de frecuencias a las que está optimizado es entre 2.5[KHz] a 20[KHz], como podemos ver en la [Figura 5.17](#).

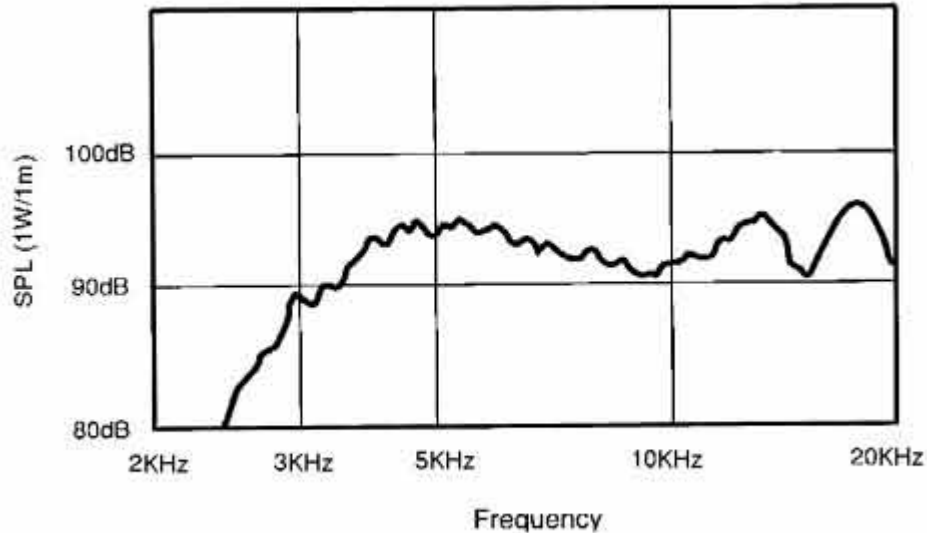


Fig. 6 Super Horn frequency response

Figura 5.17.: Respuesta en frecuencia del tweeter. Imagen de, [\[33\]](#).

El principio de operación se basa en la piezoelectricidad, es decir, se induce un campo eléctrico al aplicar fuerzas mecánicas a un material, al ser un fenómeno reversible, sabemos que se inducen fuerzas mecánicas al aplicar campos eléctricos en un material. El funcionamiento de un tweeter consiste en deformar ligeramente dos placas piezoeléctricas, que tienen una membrana entre ellas, al aplicar una corriente eléctrica. Al detener la corriente las placas vuelven a su forma original. Las placas están conectadas de tal manera que al deformarse una, la otra esta en su forma original. Este arreglo se llama "biamorfo". Al colocar un cono de en el centro del biamorfo y hacerlo vibrar, el cono seguirá su vibración.

Debido a su construcción mecánica, el tweeter se puede representar eléctricamente por un capacitor con una pequeña resistencia de disipación.

5.9. Diagrama esquemático y simulación.

Como se ha mencionado a lo largo del capítulo, todas las etapas han sido analizadas y configuradas para poder ser acopladas entre sí.

El acoplo de las etapas del sistema muestra que la configuración final estará compuesta por los siguientes elementos:

- Microcontrolador PIC16F887,[10].
- Arreglo de resistencias R2R (Conversión Digital Analógica).
- Pantalla LCD 16X2,[12].
- Configuración filtro paso-bajas.
- Opto acoplador resistivo controlado por modulación por ancho de pulso.

Estos componentes se muestran en la **Figura 5.18.**

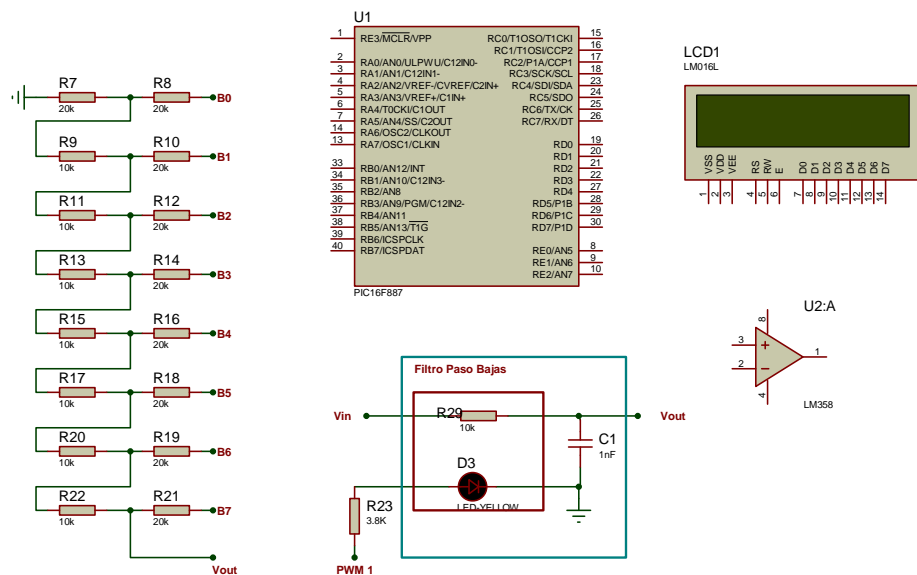


Figura 5.18.: Componentes necesarios para el sistema.

A partir de estos componentes se podrán generar todas las señales y resolver la problemática.

5.9.1. Repartición de los puertos del microcontrolador para la configuración del sistema.

El puerto A cumplirá con la función de operar la pantalla LCD y de esta forma, lograr el monitoreo de las acciones realizadas por el aparato.

El puerto B será ocupado en su totalidad como salidas que pasarán a través del arreglo de resistencias R-2R para obtener la señal deseada.

El puerto C se usará para producir la señal rectangular, polarizar la pantalla LCD, controlar las salidas de PWM⁸.

El puerto D estará dedicado únicamente a leer las señales de salida del sistema SI-PCI MED (Figura 2.2) para hacer la codificación de dichas señales, y a partir de esto poder seleccionar la frecuencia de la señal deseada.

El puerto E estará configurado para poder controlar la forma de onda que se requiera.

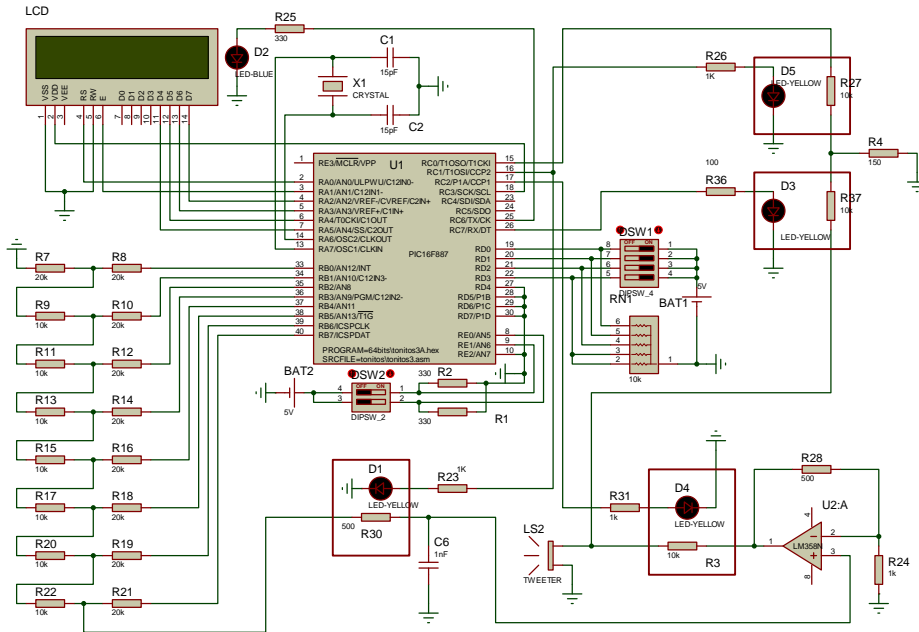


Figura 5.19.: Diagrama esquemático final para realizar pruebas en la simulación.

Como se muestra en la Figura 5.19, ese será el diseño para las simulaciones del sistema, con la finalidad de poder optimizar el programa que se cargará en el microcontrolador.

Como se ha mencionado, dicho programa contara con tres partes esenciales:

- Control. - A partir de las entradas del puerto D del microcontrolador, se direccionará a la señal según sea el caso.
- Procesamiento. - Generará una forma de onda diferente. Cada una de estas tendrá varias frecuencias fijas.
- Monitoreo. - Mostrará las características de la señal (forma de onda y frecuencia) o estado de espera, en la pantalla LCD.

⁸Pulse Width Modulation

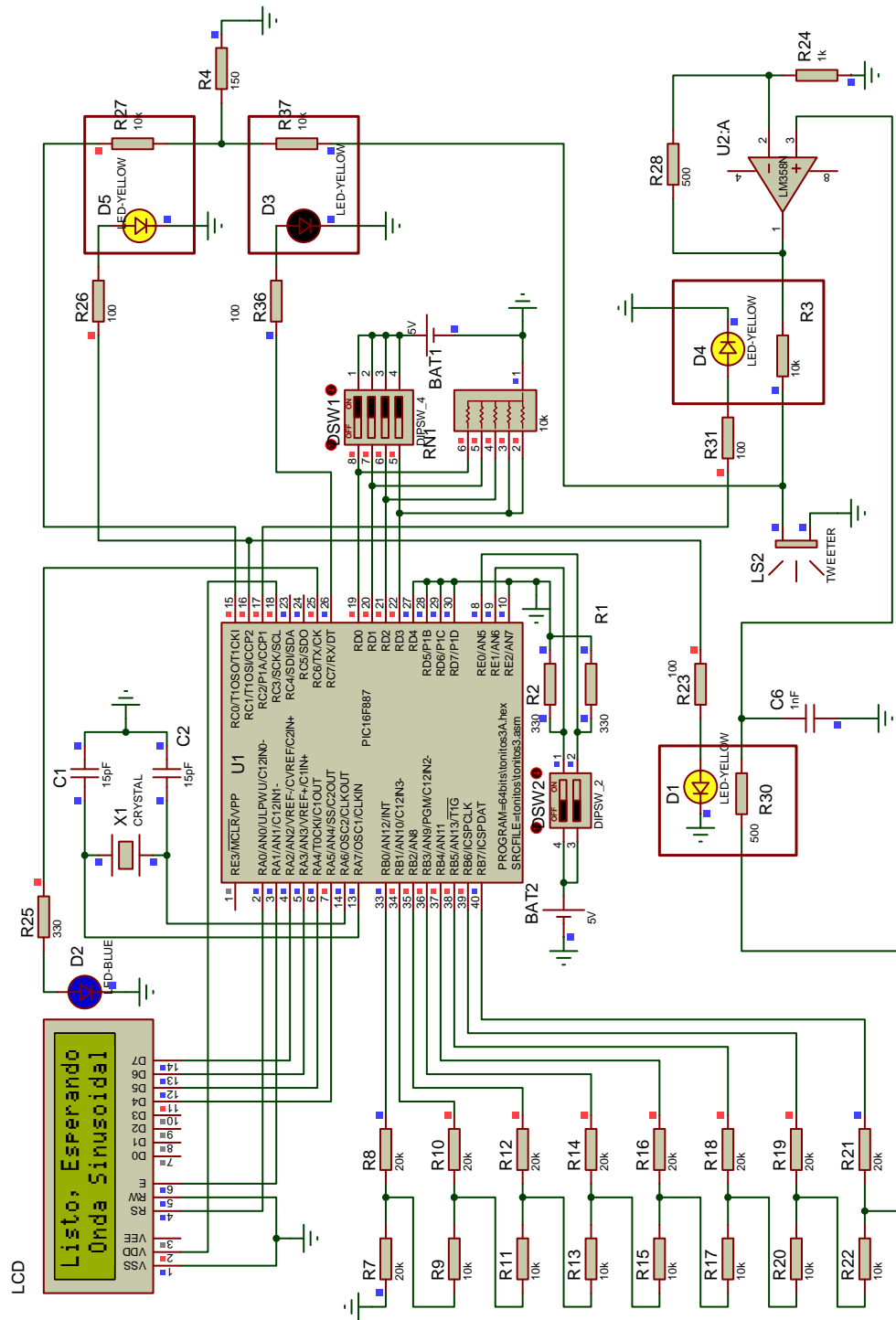


Figura 5.20.: Imagen con la simulación en ejecución en modo de espera.

La Figura 5.20 muestra un ejemplo de como se depuro la programación del sistema para

poder comparar los resultados simulados con los físicos y lograr una mayor optimización del sistema.

5.10. Conclusiones.

A lo largo del capítulo se explica la configuración de cada etapa del sistema y cada uno de sus objetivos.

De este punto se podrían observar algunas características importantes para poder mejorar el sistema en alguna etapa de actualización, si ese fuera el caso.

6. Comprobación de las señales en sus diferentes frecuencias.

6.1. Formas de onda generadas

El sistema se diseñó para poder generar tres tipos de ondas diferentes, dándole un valor agregado, debido a que el sistema empleado por MED carece de esta característica, esto hará que nuestro sistema sea más valioso.

Los tres tipos de ondas que generará nuestro sistema son los siguientes:

1. Onda sinusoidal.
2. Onda rectangular.
3. Onda triangular.

Dichas ondas fueron generadas e implementadas como se mencionó en la [Sección 5.3](#) de este documento, en donde se explico únicamente la generación de la señal sinusoidal.

En este capítulo se explicará como se generaron los tres tipos de onda y como son sus respuestas en tiempo continuo. Estas pruebas serán presentadas usando Matlab.

Haciendo un breve recordatorio, las señales discretizadas almacenadas en el microcontrolador, son valores que forman parte de un arreglo denominado tabla de consulta "*lookup*", que posteriormente son enviadas a la etapa de conversión digital/analógico.

6.1.1. Sinusoidal.

El programa que se diseñó para realizar la generación de la tabla de consulta "lookup" para un periodo de señal sinusoidal se muestra abajo.

Se nota como se hace la interpolación desde la generación de la señal (Figura 6.1), para que de esta forma podamos conseguir construir una señal sinusoidal con la mejor resolución posible. En este código también hay una parte en la cual los valores que generará el programa, serán almacenados en un archivo de texto con el nombre «Tabla Senos». Este archivo contendrá todos los valores de la señal ya separados con comas entre cada valor, para poderlos escribir directamente en el código fuente del microcontrolador.

```
clear all %Limpia pantalla

dt = 0.0000002903 % N = 2044
N = 2044; %Lóngitud Máxima de Registro de Almacenamiento
n = 0:1:N; %Índice
D = (2*pi)/N; %Delta en radianes
f = 1000 %Escribir la frecuencia de Salida deseada en Hertz%

s = D/(dt*f);
S = N/s %Número fraccionario decimal para insertar en el programa del uC
NumSalto = floor(S) %Número entero decimal para insertar en el programa del uC
HexSalto = dec2hex(NumSalto+1) %Número en hexadecimal para insertar en el programa del uC

Seno1 = 127*sin((2*pi*n)/N)+128; % Valores de Alturas de la Funcion Seno.
Seno2 = 127*sin((2*pi*(n+1))/N)+128; % Normalizados y con ofset para programar el uC
Seno3 = 255*(Seno2-Seno1)+128; % Valores para la interpolación de la funcion Seno

AltA=floor(Seno1); % Crea el archivo de texto con sus
Alt1 = fopen('TablaSenos.txt','w'); % Respetivos valores.
fprintf(Alt1,'%0.0f;',AltA);
fclose(Alt1);

plot(Seno3); %Imprime Señal sinusoidal.
axis([0 2044 0 260]) %Acota la gráfica.
grid on
grid minor
hold on %Detiene la primer gráfica.
plot(Seno1,'r');
```

Código de Matlab para generar un ciclo de onda sinusoidal.

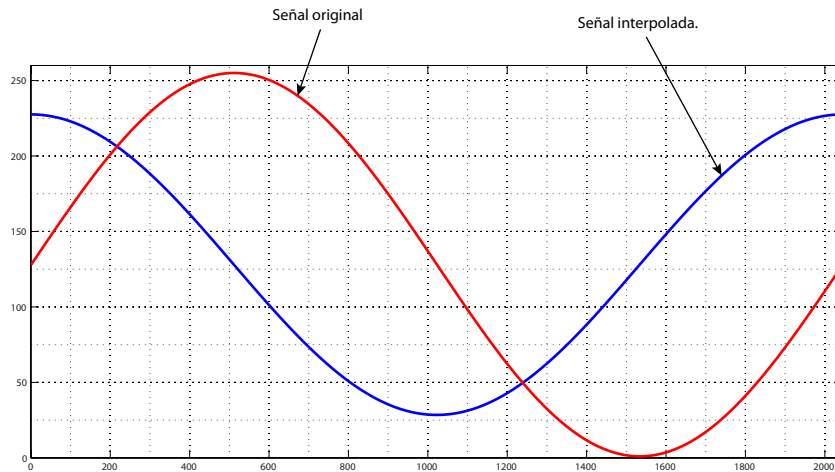


Figura 6.1.: Gráfica de los valores obtenidas a partir del código escrito en Matlab.

6.1.2. Triangular.

El código que se diseñó para hacer la generación de la señal triangular cambia ligeramente con respecto al ejercicio anterior, puesto que la función de la cual se hace uso es "tripuls()". Estos cambios se ven en la constante por la que se tiene que multiplicar la función, que es el doble de la pasada (Figura 6.1). Y que se debe generar un vector de n valores iniciando desde un número negativo, para que la función genere el periodo de nuestra señal con las características que deseamos. También es pertinente mencionar que para la generación de esta señal, no se hará uso de interpolación.

Como en el programa pasado, este también muestra la señal de salida (Figura 6.2) y almacena los 1020 valores en un archivo de texto llamado «TablaTriangulo».

```
clear all          %Limpia pantalla
grid on           %Activa cuadrícula
dt = 0.0000002903276011 %Delta de t tiempo que se tarda en leer cada data1 el uC
N = 2044;         %Lóngitud Máxima de Registro de Almacenamiento
n = -1021:1:1022; %Longitude del Vector

D = (2*pi)/N;    %Delta en radianes
f = 1000         %Escribir la frecuencia de Salida deseada en Hertz%

s = D/(dt*f);
S = N/s          %Número fraccionario decimal para insertar en el programa del uC
NumSalto = floor(S) %Número entero decimal para insertar en el programa del uC
HexSalto = dec2hex(NumSalto) %Número en hexadecimal para insertar en el programa del uC

Triangular = 245*TRIPULS(n,N)+5; % Valores de Alturas de la Funcion Seno,

AltA=floor(Triangular); % Crea el archivo de texto con sus
Alt1 = fopen('TablaTriangulo.txt','w'); % respectivos valores
fprintf(Alt1,'%0.0f\n',AltA);
fclose(Alt1);

plot(Triangular); %Imprime señal triangular discreta
AXIS([0 2060 0 260])
grid minor       %Activa Cuadrícula mas precisa.
```

Código fuente para la generación de un periodo de señal triangular.

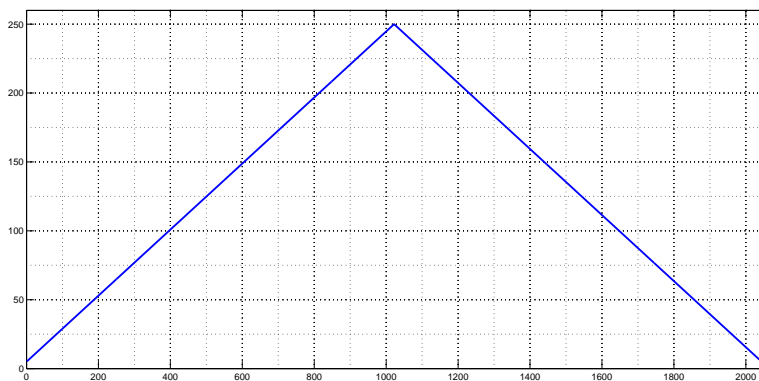


Figura 6.2.: Gráfica de la señal triangular generada con Matlab.

6.1.3. Rectangular.

El método utilizado para la generación de la onda rectangular, es diferente a la obtención de las señales anteriores. Esto es debido a que dicha señal puede formarse a partir del cambio del estado lógico de un pin configurado como salida en uno de los puertos disponibles.

Esto quiere decir que alternando el estado entre encendido y apagado de un pin de salida del puerto, se logrará obtener la señal de onda rectangular.

Para conocer los intervalos de tiempo que debe estar encendido y apagado dicho pin, se dividirá entre 2 el periodo de la señal deseada.

$$t = \frac{1}{2 \cdot f}$$

De forma que el valor obtenido será el tiempo que el pin necesite estar encendido y apagado. Estos valores de tiempo son retrasos que deben ocurrir entre el cambio de estado lógico del puerto.

6.2. Análisis espectral de las señales sinusoidales generadas.

En esta sección se mostrará como son los espectros de las señales generadas en Matlab. En específico las ondas sinusoidales, debido a que un tono es una señal sinusoidal de una frecuencia y amplitud fija.

Este análisis es puramente teórico puesto que la señal no tiene perturbaciones externas como las tiene el sistema físico. Tales perturbaciones pueden ser ruido térmico, retrasos por programación etc. Así que este análisis es ideal.

Para correr esta prueba, se volvió a diseñar un programa en Matlab el cual graficará la señal que es semejante a la continua y su espectro en frecuencias.

6.2 Análisis espectral de las señales sinusoidales generadas.

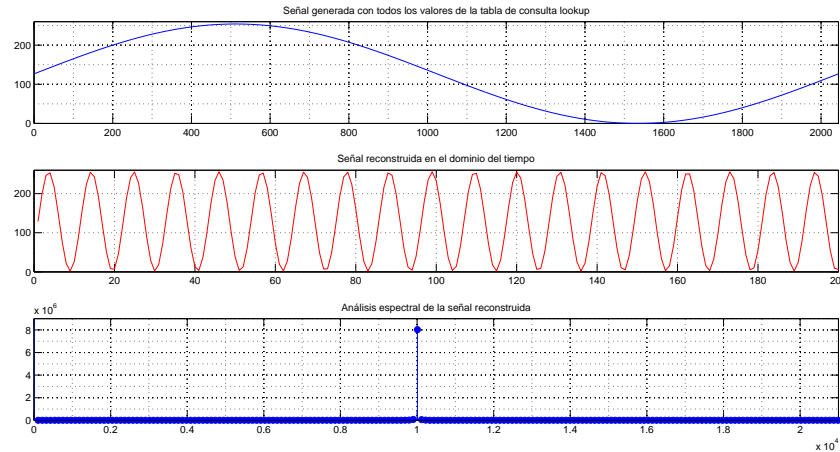


Figura 6.3.: Gráficas obtenidas para el análisis de las señales.

Como se observa en la [Figura 6.3](#), la primer gráfica de arriba hacia abajo, es la señal que se genera usando los 2044 valores disponibles en el registro auxiliar del programa. Podemos ver que usando todos los valores disponibles en el registro auxiliar, la señal generada tendrá un periodo mayor y un frecuencia mas baja.

En la segunda gráfica observamos una señal de salida de 10 KHz. Esta usa un menor número de muestras, por lo que leer un ciclo de la señal será más rápido y con esto ganaremos disminuir el periodo de la señal y aumentar la frecuencia, con la desventaja de perder resolución en la señal.

En la tercer gráfica se ve el espectro de la señal de 10 KHz. Como se puede apreciar, la espiga de esta señal aparece en 10 KHz, con lo que podemos validar la frecuencia de salida de la señal deseada.

6.2.1. Gráficas para el análisis de las señales sinusoidales.

Para fines prácticos, las siguientes figuras no mostrarán la primer gráfica de la [Figura 6.3](#), debido a que esa señal es el patrón de todas y es la misma para cada frecuencia.

Comprobación de las señales en sus diferentes frecuencias.

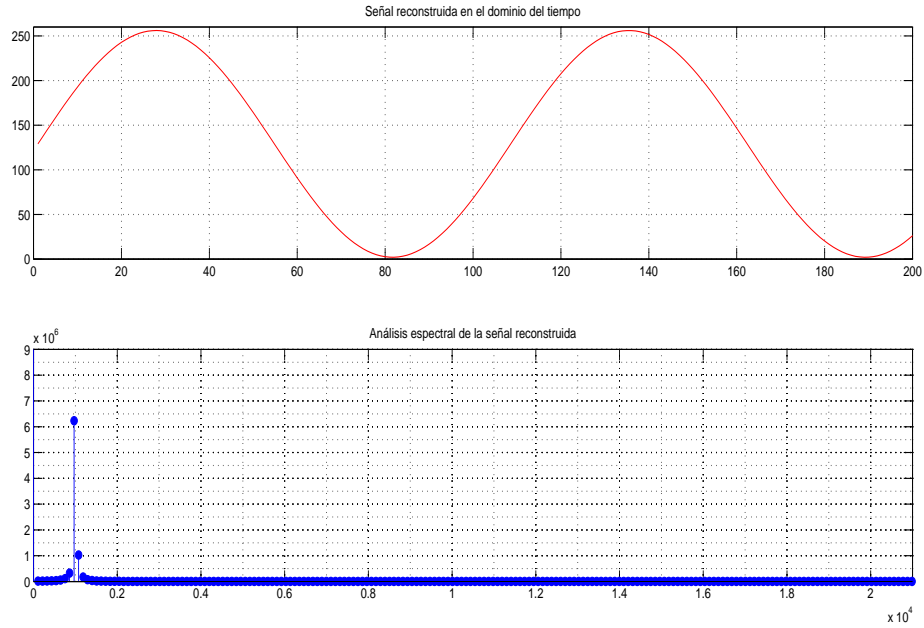


Figura 6.4.: Gráfica para el análisis de la señal a 1 [KHz].

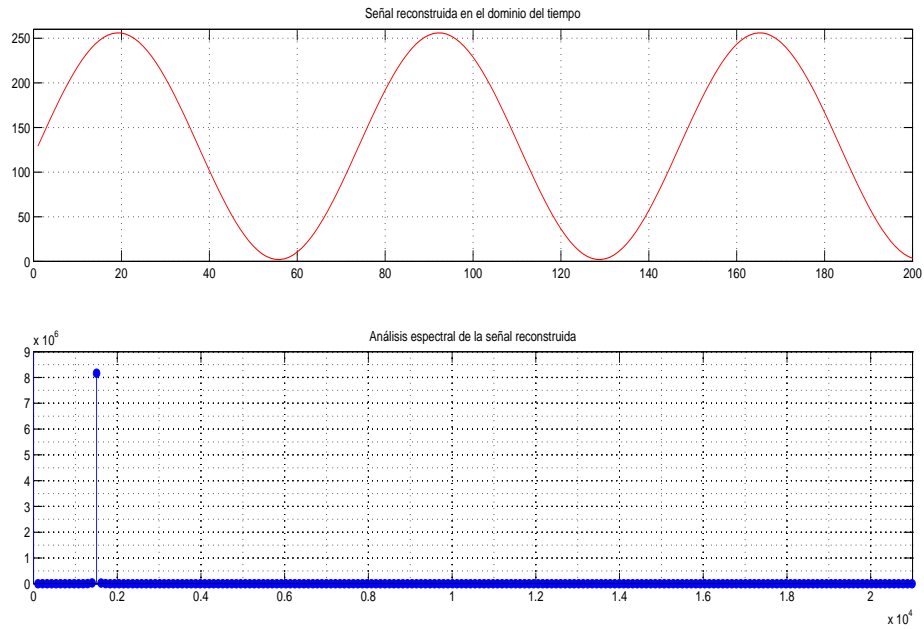


Figura 6.5.: Gráfica para el análisis de la señal a 1.5 [KHz].

6.2 Análisis espectral de las señales sinusoidales generadas.

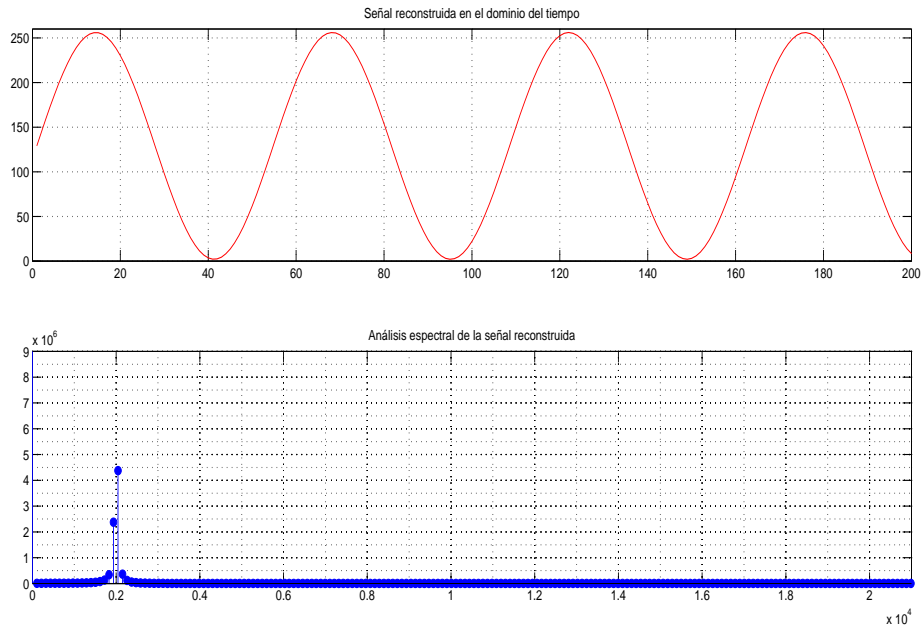


Figura 6.6.: Gráfica para el análisis de la señal sinusoidal a 2 [KHz].

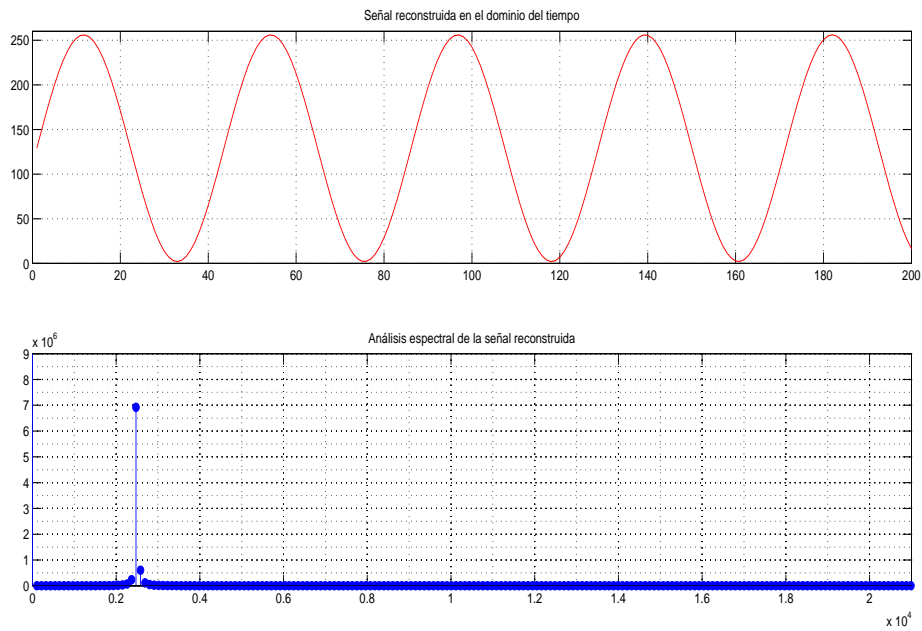


Figura 6.7.: Gráfica para el análisis de la señal sinusoidal a 2.5 [KHz].

Comprobación de las señales en sus diferentes frecuencias.

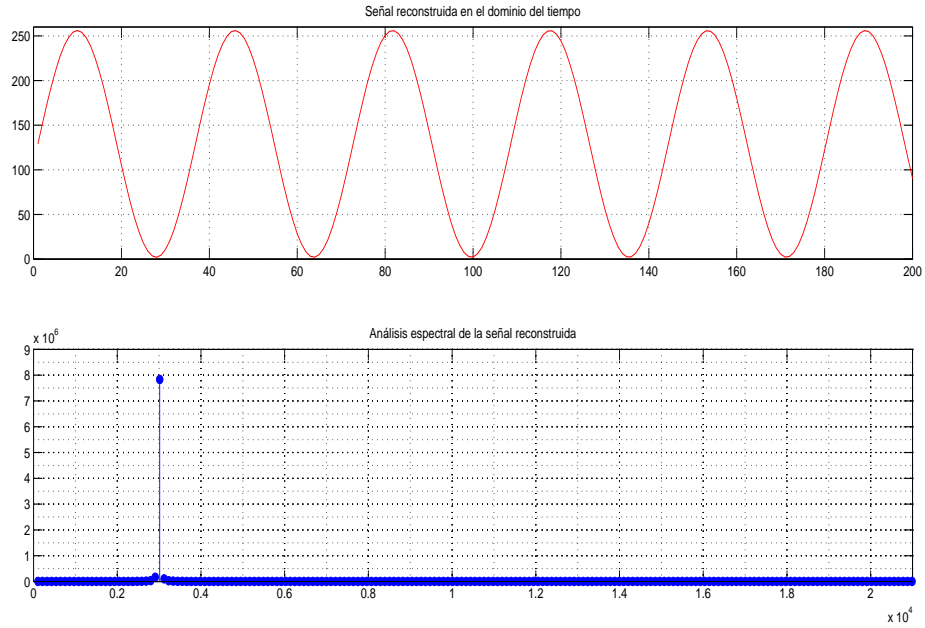


Figura 6.8.: Gráfica para el análisis de la señal sinusoidal a 3 [KHz].

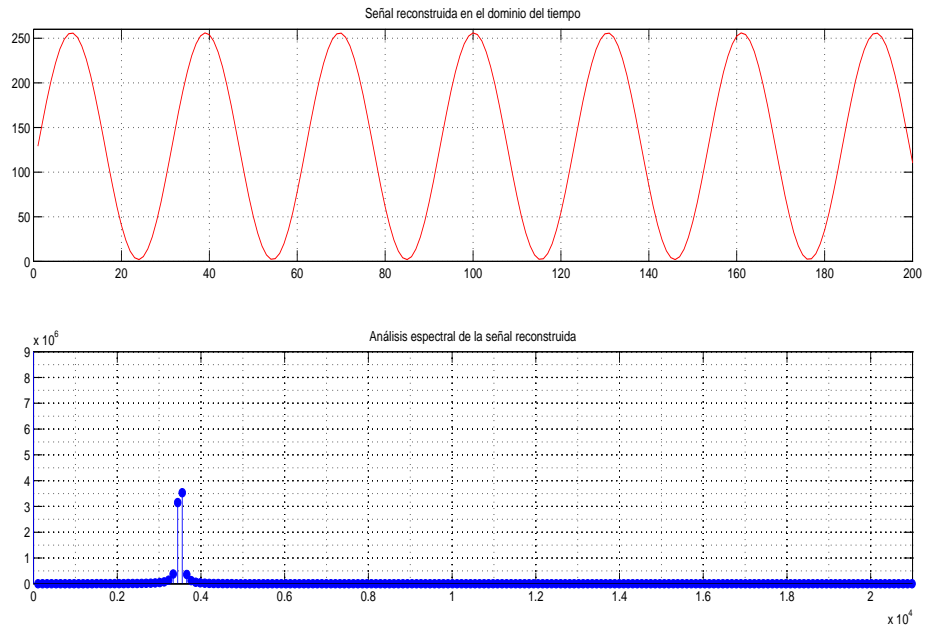


Figura 6.9.: Gráfica para el análisis de la señal sinusoidal a 3.5 [KHz].

6.2 Análisis espectral de las señales sinusoidales generadas.

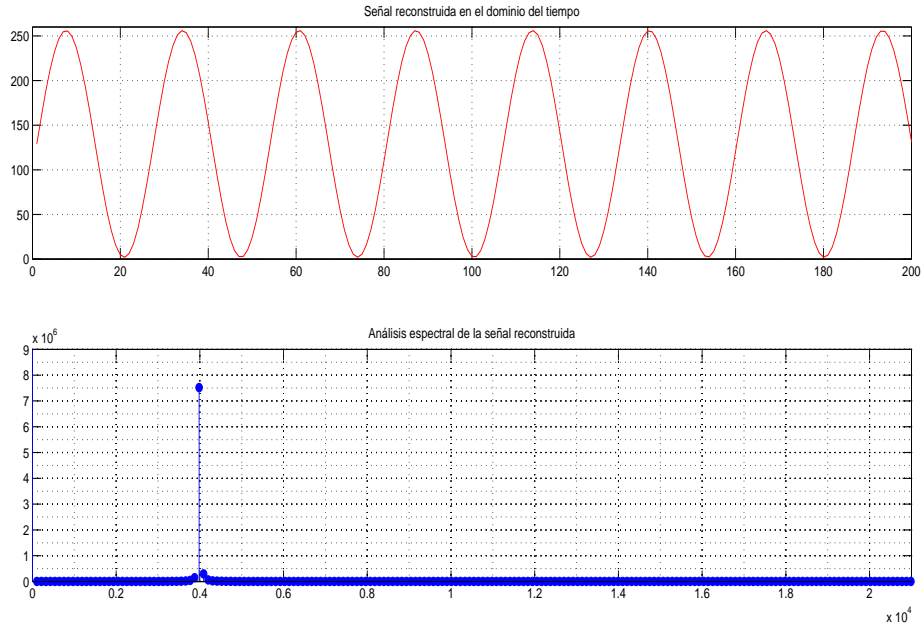


Figura 6.10.: Gráfica para el análisis de la señal sinusoidal a 4 [KHz].

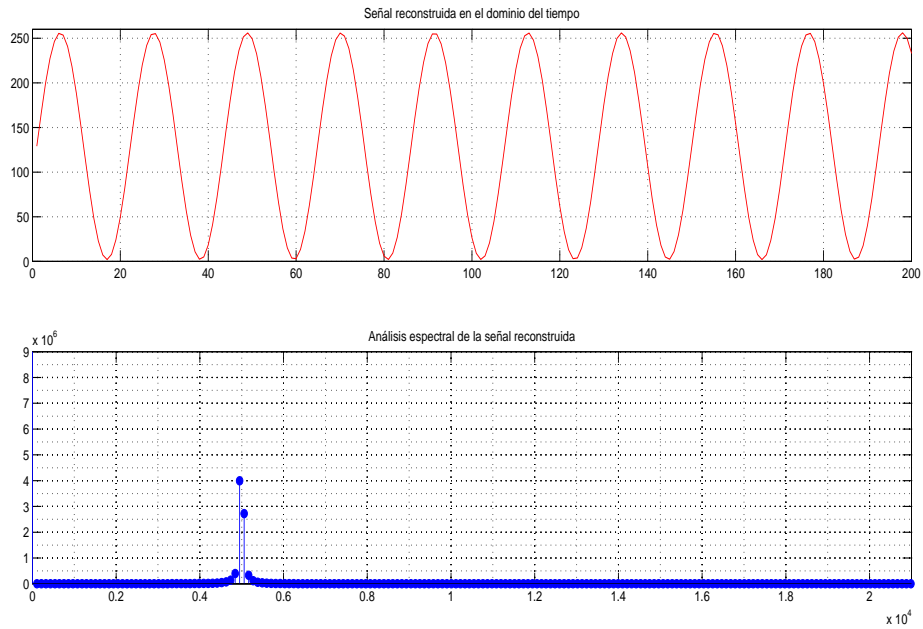


Figura 6.11.: Gráfica para el análisis de la señal sinusoidal a 5 [KHz].

Comprobación de las señales en sus diferentes frecuencias.

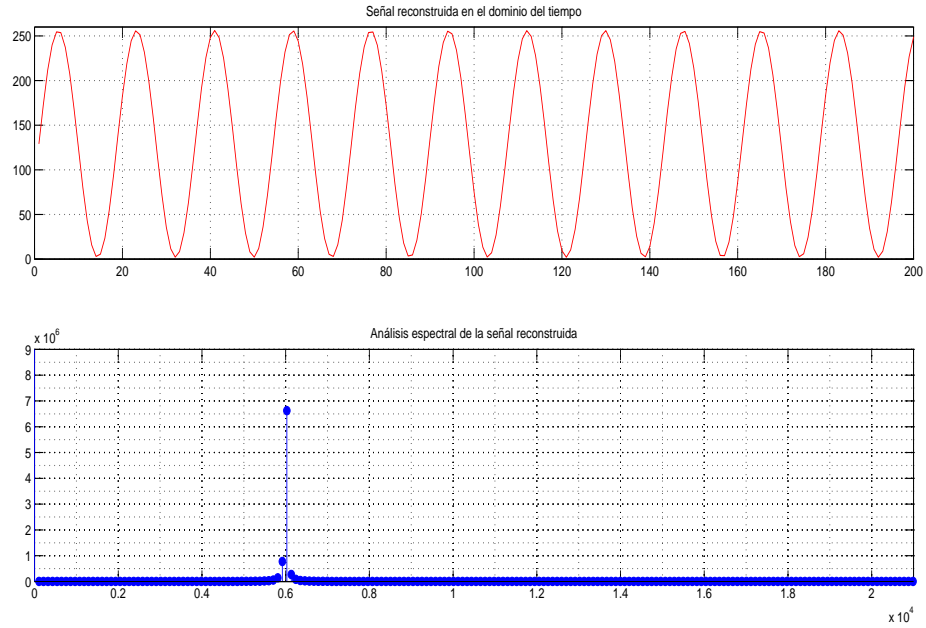


Figura 6.12.: Gráfica para el análisis de la señal sinusoidal a 6 [KHz].

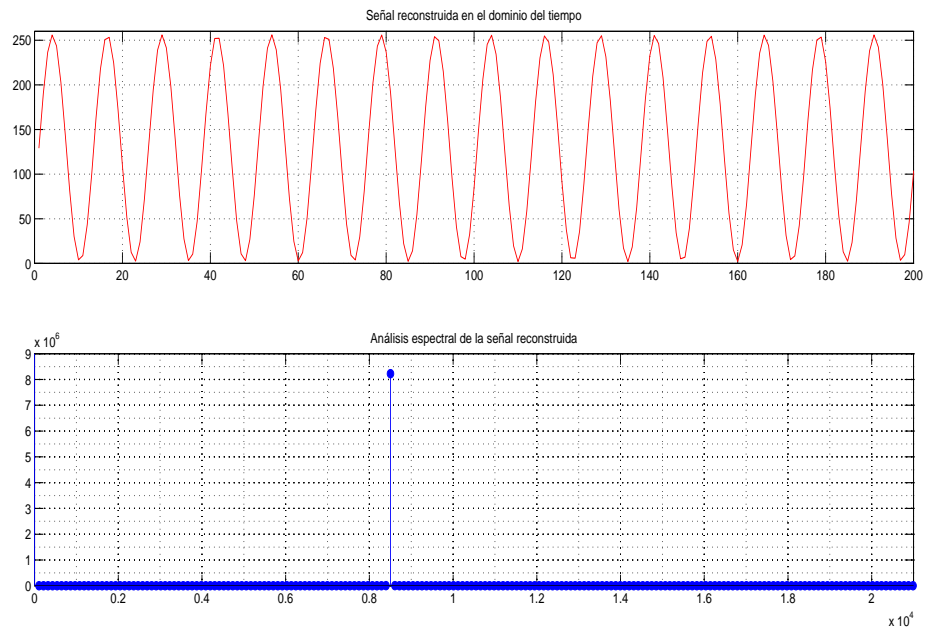


Figura 6.13.: Gráfica para el análisis de la señal sinusoidal a 8.5 [KHz].

6.2 Análisis espectral de las señales sinusoidales generadas.

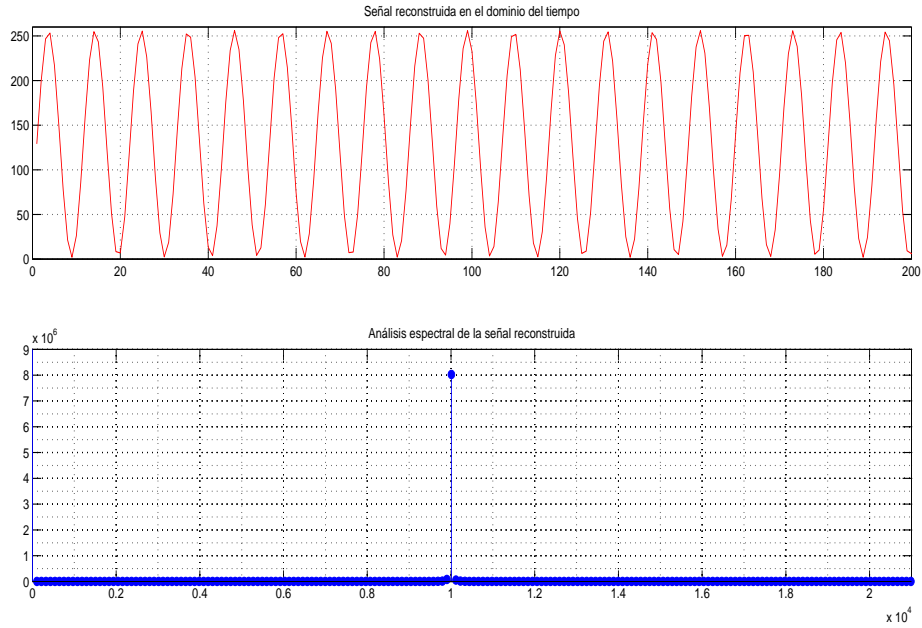


Figura 6.14.: Gráfica para el análisis de la señal sinusoidal a 10 [KHz].

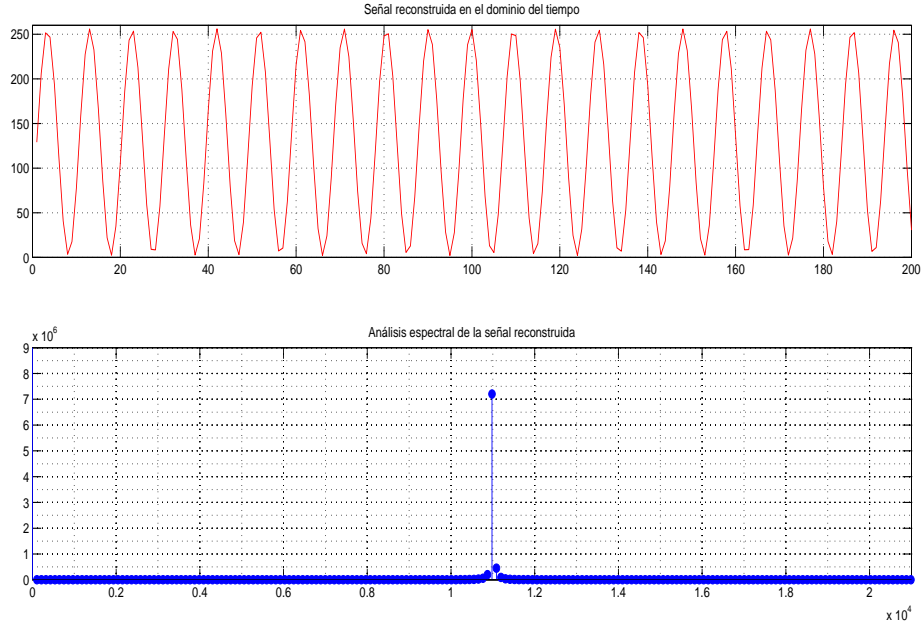


Figura 6.15.: Gráfica para el análisis de la señal sinusoidal a 11 [KHz].

Comprobación de las señales en sus diferentes frecuencias.

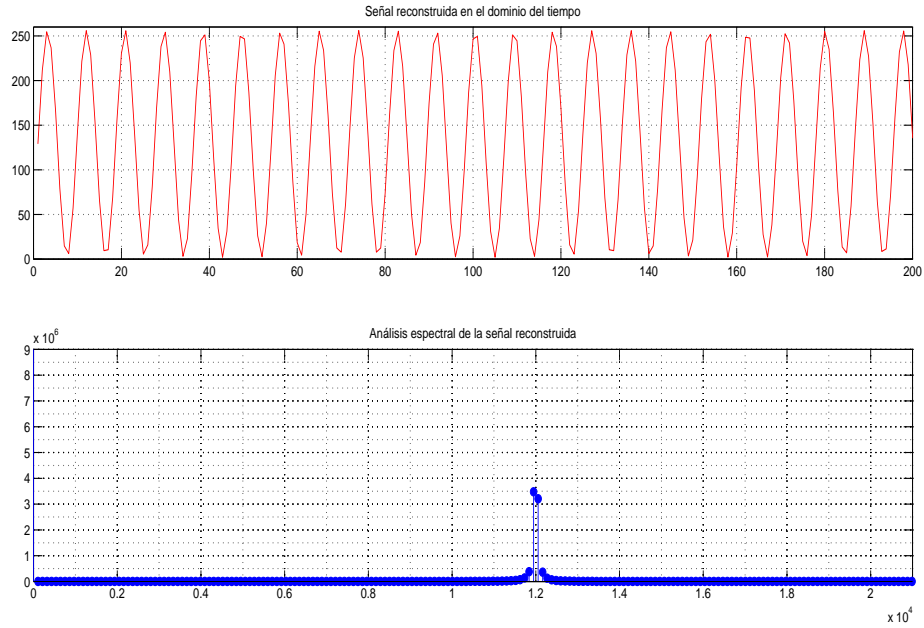


Figura 6.16.: Gráfica para el análisis de la señal sinusoidal a 12 [KHz].

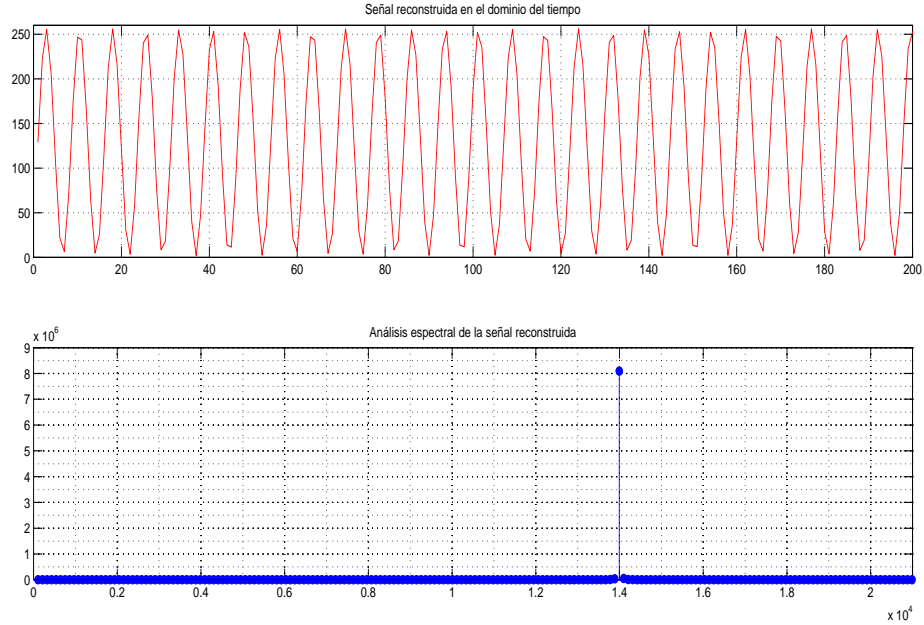


Figura 6.17.: Gráfica para el análisis de la señal sinusoidal a 14 [KHz].

6.3 Comprobación de señales acústicas producidas.

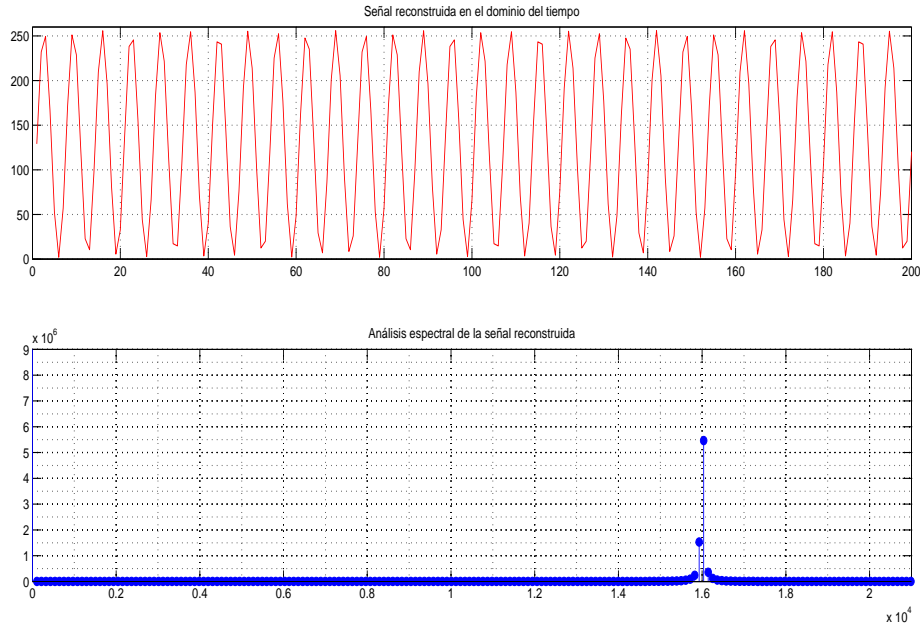


Figura 6.18.: Gráfica para el análisis de la señal sinusoidal a 16 [KHz].

6.3. Comprobación de señales acústicas producidas.

En esta sub-sección se mostrarán los análisis espectrales de los diferentes tipos de onda generados por el estimulador acústico.

Este tipo de medición acústica se hizo con un software llamado *Adobe Audition CS5.5* y sirvió para la calibración de potencia acústica.

- La **Figura 6.19**, muestra la respuesta en frecuencia de la señal sinusoidal generada por el dispositivo a una frecuencia de 3 [KHz].

Comprobación de las señales en sus diferentes frecuencias.

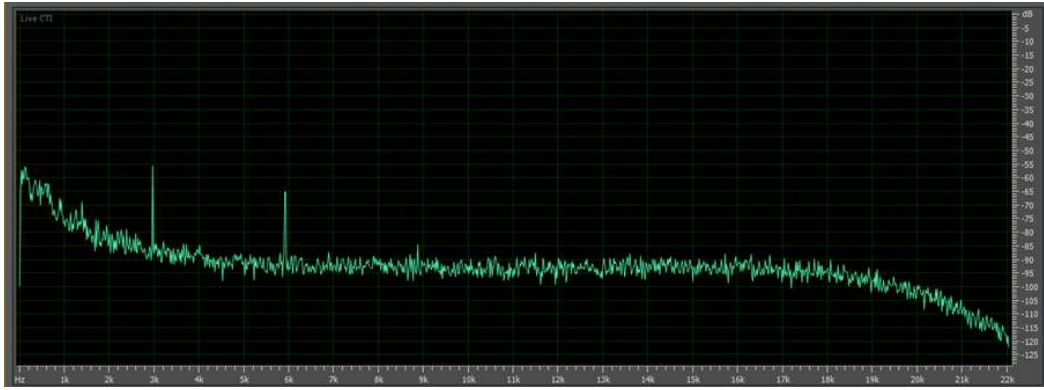


Figura 6.19.: Comprobación acústica de la señal sinusoidal de 3 [KHz].

- La **Figura 6.20**, muestra la respuesta en frecuencia de la señal triangular generada por el dispositivo a una frecuencia de 3 [KHz].

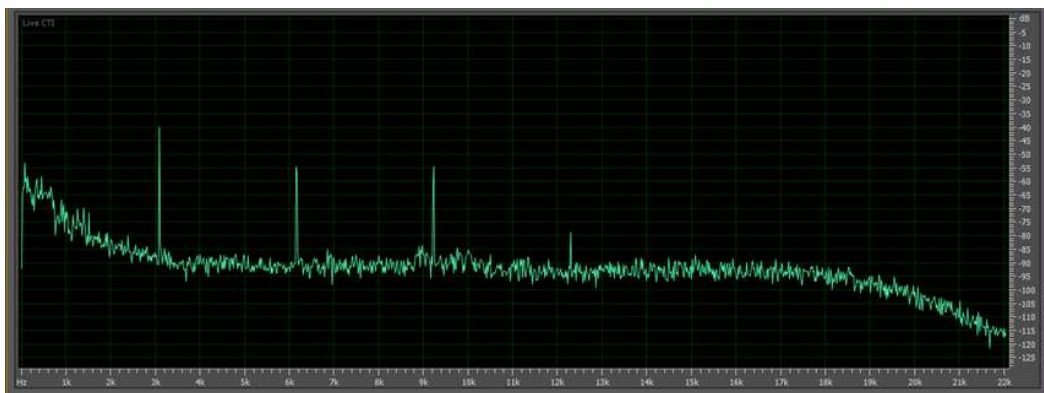


Figura 6.20.: Comprobación acústica de la señal triangular de 3 [KHz].

- La **Figura 6.21**, muestra la respuesta en frecuencia de la señal rectangular generada por el dispositivo a una frecuencia de 3 [KHz], y la **Figura 6.22** es la misma medición a una escala diferente, para demostrar que el contenido armónico en esta forma de onda es mayor.

6.3 Comprobación de señales acústicas producidas.

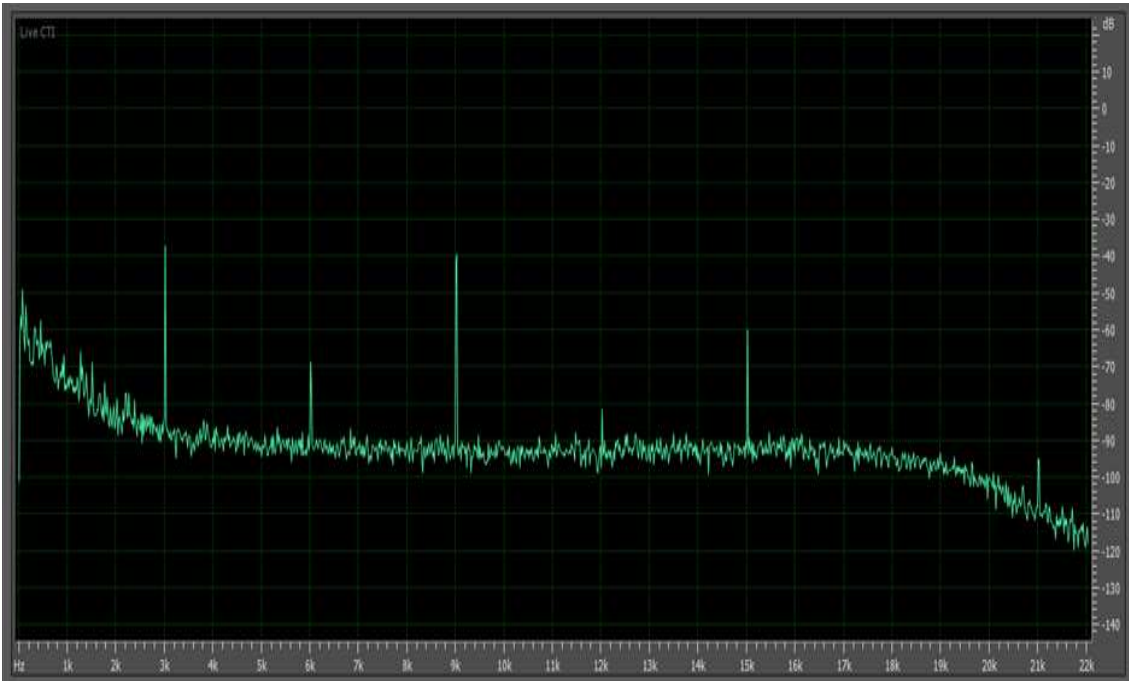


Figura 6.21.: Comprobación acústica de la señal rectangular de 3 [KHz].

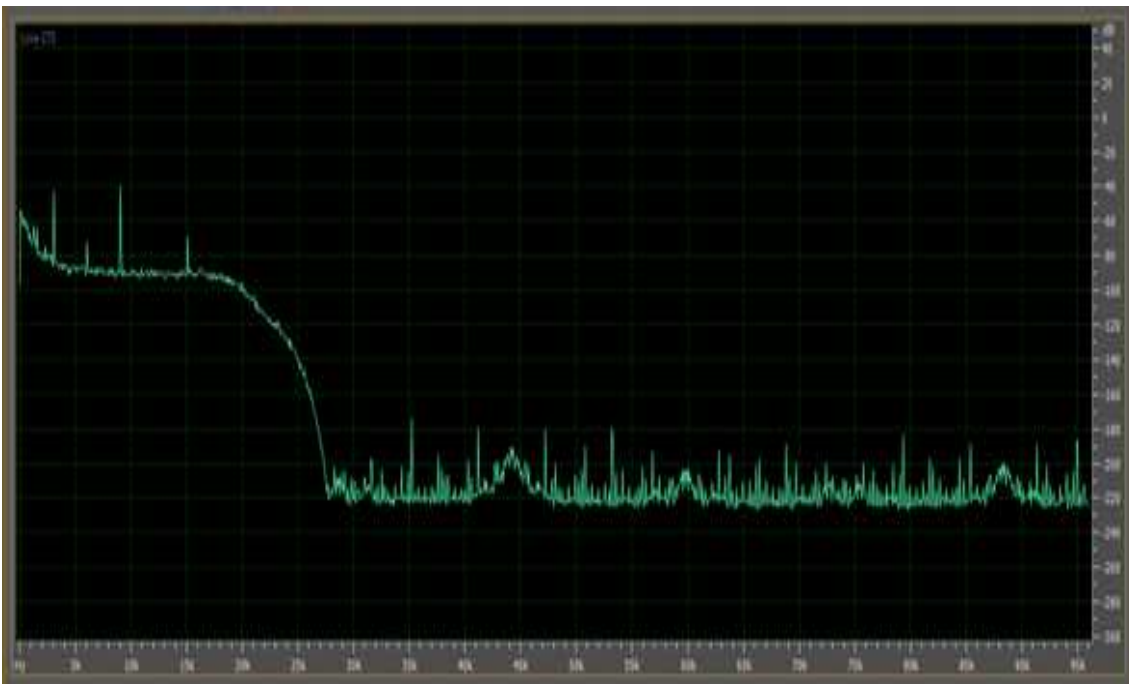
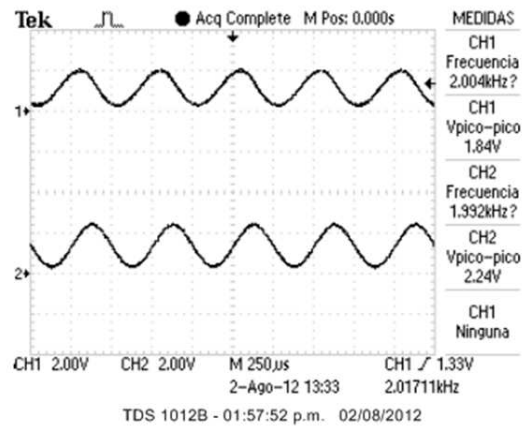


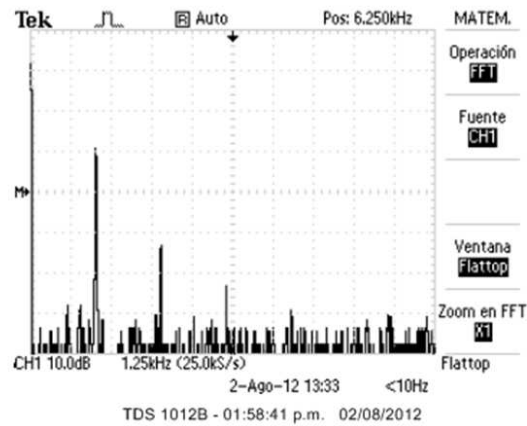
Figura 6.22.: Comprobación acústica de la señal rectangular de 3 [KHz] a diferente escala.

Comprobación de las señales en sus diferentes frecuencias.

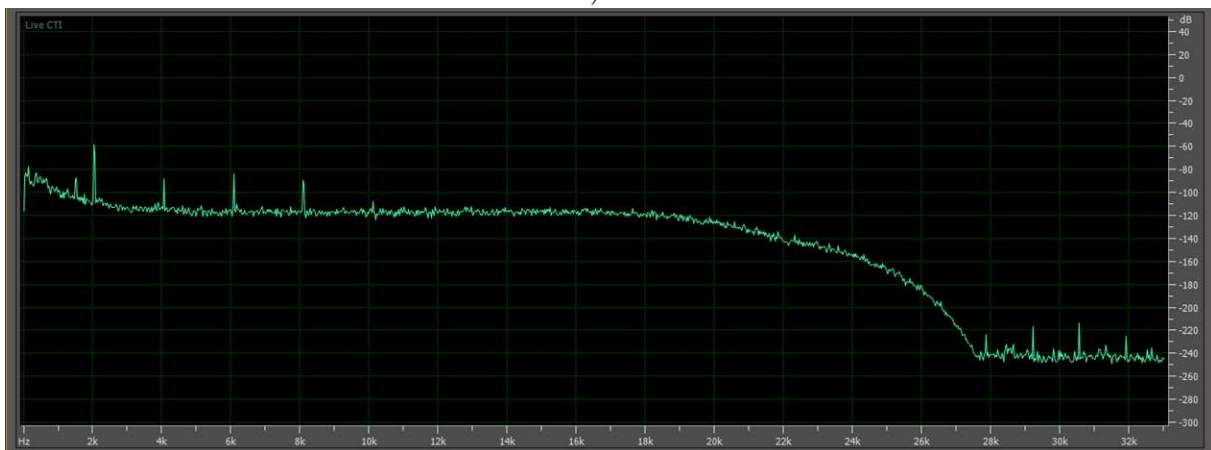
6.4. Comprobación de las señales eléctricas y acústicas generadas.



a)



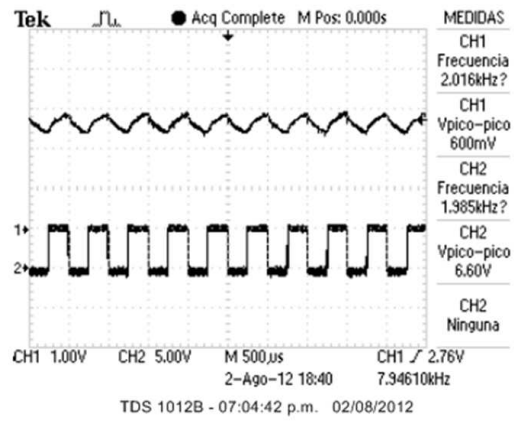
b)



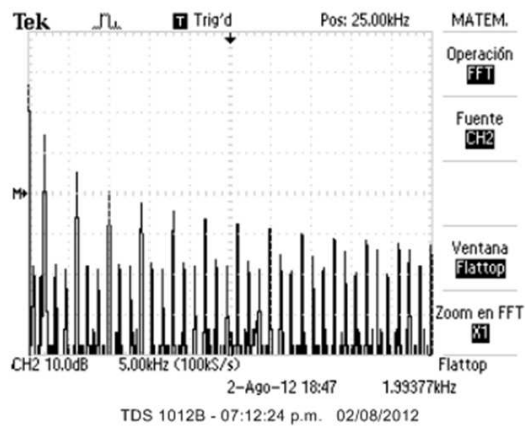
c)

Figura 6.23.: Comprobación eléctrica y acústica de la señal sinusoidal de 2 [KHz].

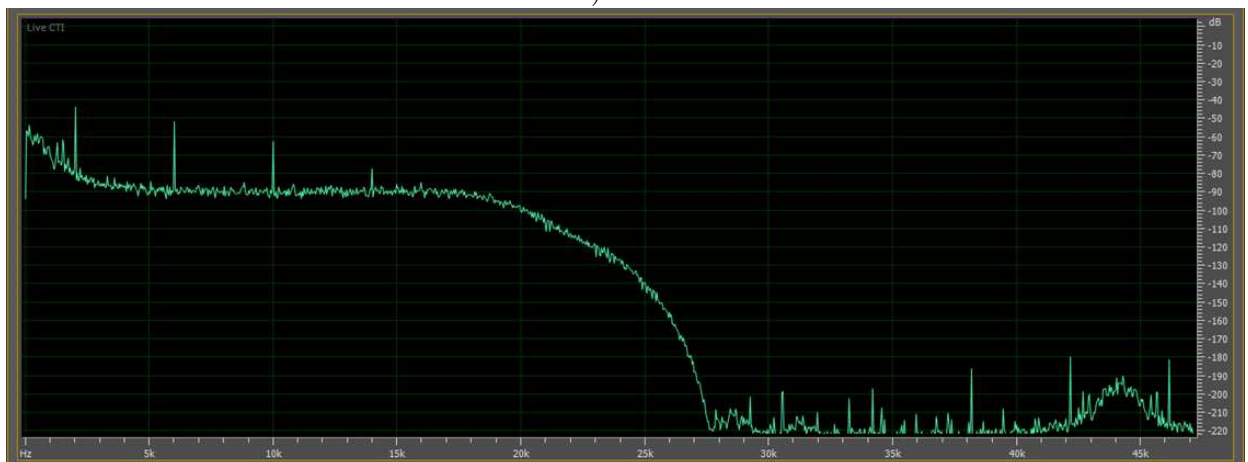
Comprobación de las señales en sus diferentes frecuencias.



a)



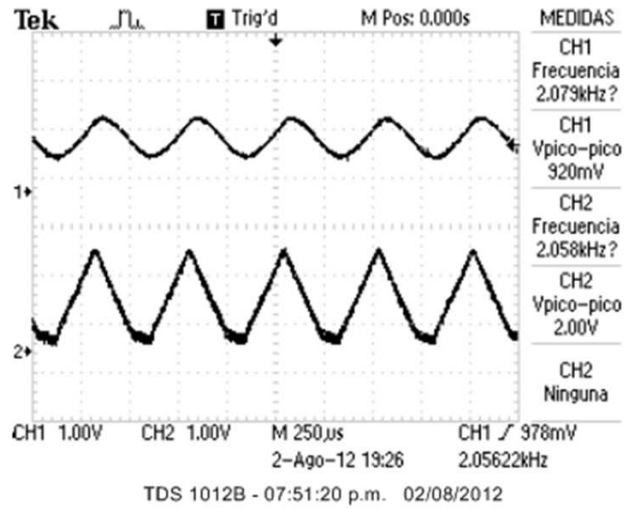
b)



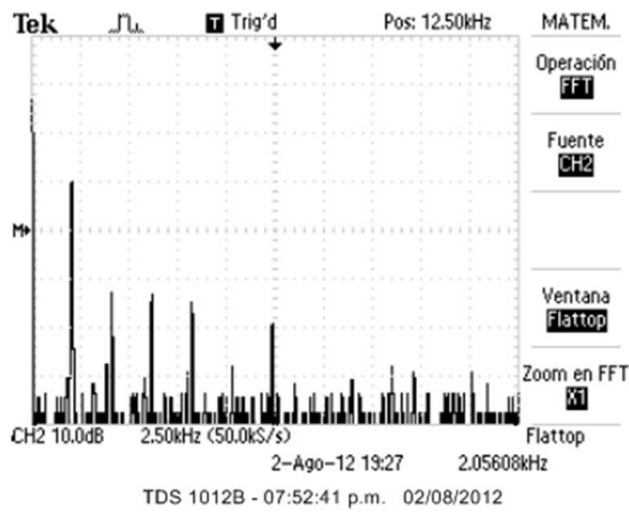
c)

Figura 6.24.: Comprobación eléctrica y acústica de la señal rectangular de 2 [KHz].

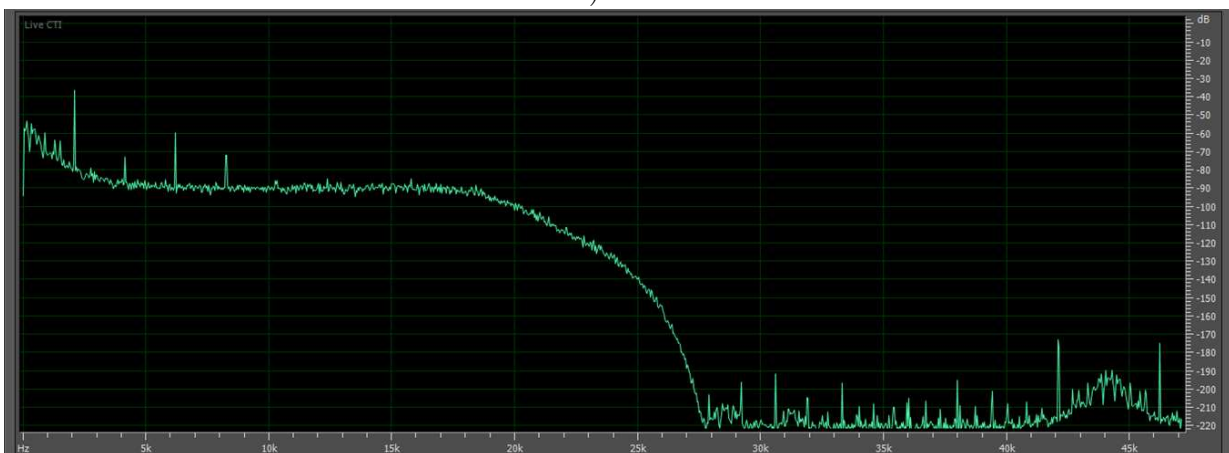
6.4 Comprobación de las señales eléctricas y acústicas generadas.



a)



b)



c)

Figura 6.25.: Comprobación eléctrica y acústica de la señal triangular de 2 [KHz].

En las figuras anteriores (Figura 6.23, Figura 6.24 y Figura 6.25) se muestra en el primer inciso la medición de la señal generada sin carga y con carga¹. En el siguiente inciso se muestra la respuesta en frecuencia del sistema con carga y en el último inciso se muestra la respuesta acústica de dicha señal medida.

6.5. Conclusiones.

A lo largo del capítulo se pudo observar en las gráficas, como al aumentar la frecuencia de la señal generada, disminuye la resolución en la señal de salida.

En el análisis espectral matemático hecho en Matlab, se observa como la espiga fundamental de la señal, se va recorriendo hacia la derecha conforme aumenta la frecuencia.

En cuanto el análisis espectral de la señal acústica de cada una de las formas de ondas generadas por el dispositivo, se observó que el contenido armónico de cada una corresponde adecuadamente a la teoría, esto quiere decir que la señal sinusoidal, debe ser la de menor contenido armónico (una señal sinusoidal perfecta, no debe contener armónicos), la forma de onda triangular tiene mayor contenido armónico a comparación de la anterior, y por último la señal rectangular es la del contenido armónico más elevado.

Debido a la naturaleza piezo-eléctrica del *tweeter*, las frecuencias menores a 2 [KHz] y mayores a 15 [KHz] se ven atenuadas en amplitud. Por otra parte, la capacitancia del *tweeter* al interactuar con el sistema, reacciona como un filtro paso altas, lo cual disminuye el contenido armónico.

De esta forma comprobamos que las ondas sonoras producidas, son congruentes con las señales eléctricas tomadas con el uso del osciloscopio y comparadas, demuestran un buen desempeño del sistema.

Este capítulo valida como las señales que procesará y construirá el microcontrolador, cumplen con las especificaciones al implementar sonidos.

¹*tweeter* piezo-eléctrico

7. Conclusiones y Perspectivas.

Fabricamos un dispositivo de bajo costo para la generación de formas de onda sinusoidal, rectangular y triangular de diferentes frecuencias programables y pre-establecidas por el usuario, que las utilizará para la estimulación acústica en sujetos de prueba.

La reconstrucción de las señales cumplen adecuadamente con la expectativa de las formas de onda propuestas; debido a la naturaleza capacitiva del *tweeter*, la señal rectangular se ve afectada en su forma de onda y las otras señales se ven beneficiadas por la formación de un filtro paso altas en la carga del sistema, obteniendo un filtro paso banda a la salida de éste.

El estimulador acústico que se desarrolló es mejor en cuanto a funciones y compatibilidad respecto a los actualmente utilizados en la facultad de psicología.

En el caso de diseñar una segunda versión de este sistema, se podrían tomar en cuenta las siguientes perspectivas:

- Adquirir un transductor con mayor ancho de banda y mejor respuesta en frecuencia para los propósitos acústicos requeridos.
- Si se perfeccionara el dispositivo "opto-acoplador resistivo controlado por modulación de ancho de pulso", en cuanto a su principio de operación y encapsulado, se podrían implementar filtros más precisos y mejor controlados; nuestras señales se verían beneficiadas en su contenido armónico recorriendo la frecuencia de corte como sea necesario.
- Una carcasa diseñada especialmente para el circuito en el *PCB* sería beneficioso para la disipación de calor.

Apéndices

A. Guía de usuario.

Estimulador acústico para la evaluación de tareas cognitivas en sujetos de prueba.

Guía de usuario.

Universidad Nacional Autónoma de México

Facultad de Ingeniería - Facultad de psicología

Proyecto PAPIME No. PE305511.

México D.F. 2012

Introducción.

El estimulador acústico es un generador de tres diferentes formas de sonido: rectangular, triangular y tipo sinusoidal como muestra la **Figura A.1**, y cada una de éstas se puede fijar a una de las quince frecuencias disponibles.

Las entradas del estimulador acústico se alimentan de los 28 volts de corriente directa del sistema interfaz PCI de *MED associates*¹, para seleccionar una de las quince frecuencias y el estado de espera.

La elección de la forma del sonido es mediante un interruptor, mientras que el volumen se ajustará usando un desarmador pequeño, ver **Figura A.2**.

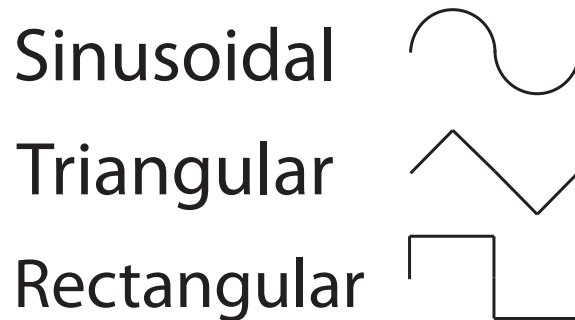


Figura A.1.: Formas de sonido.

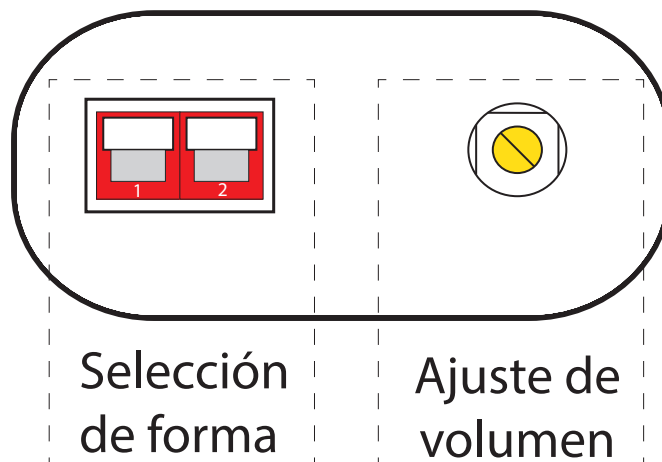


Figura A.2.: Control de funciones.

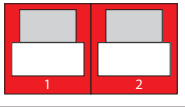
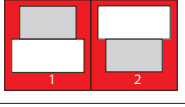
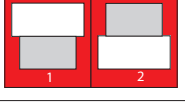
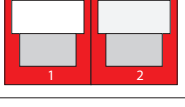
¹<http://www.med-associates.com>

Control de formas de sonido.

La forma de onda está determinada por la configuración de los apagadores, [Tabla A.1](#).

Al situar los apagadores en el estado de “configuración no válida”, el estimulador acústico no generará sonido.

Para realizar el cambio de forma de onda, el estimulador acústico debe de estar en el estado de espera [Tabla A.4](#).

Apagadores	Forma del sonido
	Sinusoidal
	Triangular
	Rectangular
	Configuración no válida

Formas de onda dependientes de los apagadores

[Tabla A.1](#)

Conexiones para 15 frecuencias más estado de espera.

Conexión 1.

El estimulador acústico de terminales “molex” de 3 pines [Figura A.3](#) cuenta con cuatro cables. El uso de cada pin se explica en la [Tabla A.2](#). El diagrama de conexión en la [Figura A.4](#).

Conexión 2.

El estimulador acústico de terminales metálicas tipo “zapata” [Figura A.5](#) cuenta con seis alambres. El uso de cada alambre se explica en la [Tabla A.3](#).

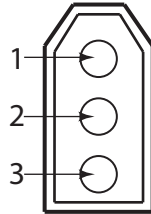


Figura A.3.: Terminal tipo “molex”

Cable	Pin	Color de alambre	Uso
Cable 1	1	Azul	Tierra
	2	Verde	Bit
	3	Rojo	28 [V]
Cable 2	1	—	Bit
	2	Blanco	
	3	—	
Cable 3	1	—	Bit
	2	Amarillo	
	3	—	
Cable 4	1	—	Bit
	2	Morado	
	3	—	

Tabla A.2 Uso de los pines, terminales “molex”

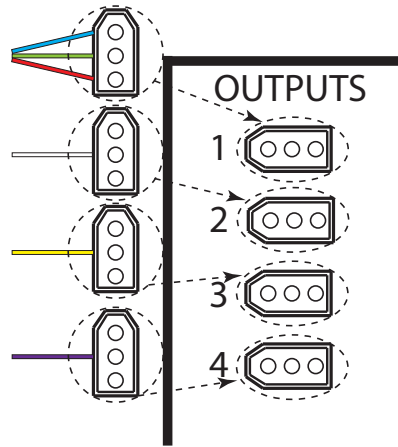


Figura A.4.: Conexión, terminales “molex”.

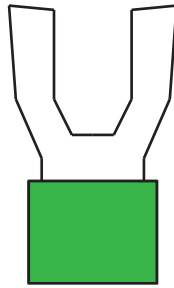


Figura A.5.: Terminal tipo “zapata”.

Color de alambre	Uso
Azul	Tierra
Rojo	28 [V]
Verde	Bit
Blanco	Bit
Amarillo	Bit
Morado	Bit

Tabla A.3 Uso de los alambres, terminales “zapata”

Control de frecuencias.

La [Tabla A.4](#) muestra las combinaciones lógicas de los bits para obtener las quince frecuencias disponibles y el estado de esperando.

Frecuencia Kilo Hertz (KHz)	Color del alambre			
	Morado	Amarillo	Blanco	Verde
Espera	0	0	0	0
1	0	0	0	1
1.5	0	0	1	0
2	0	0	1	1
2.5	0	1	0	0
3	0	1	0	1
3.5	0	1	1	0
4	0	1	1	1
5	1	0	0	0
6	1	0	0	1
8.5	1	0	1	0
10	1	0	1	1
11	1	1	0	0
12	1	1	0	1
14	1	1	1	0
16	1	1	1	1

[Tabla A.4](#) Combinaciones lógicas de los bits para obtener las frecuencias disponibles.

Conexiones para 7 frecuencias más modo de espera.

Para configurar este modo, se debe colocar un “jumper” como se muestra, [Figura A.6](#) y [Figura A.7](#):

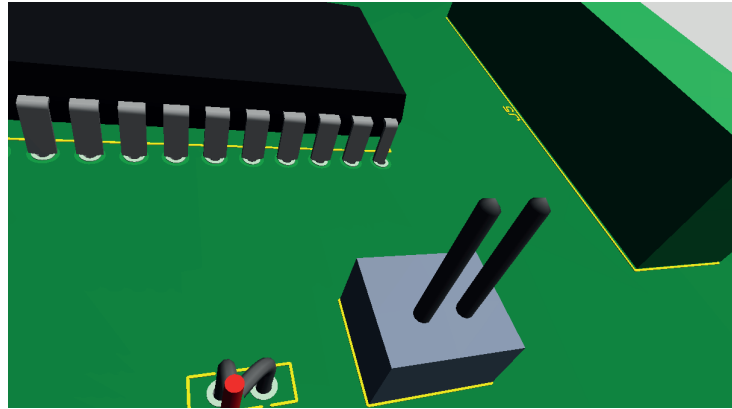


Figura A.6.: Configuración sin jumper.

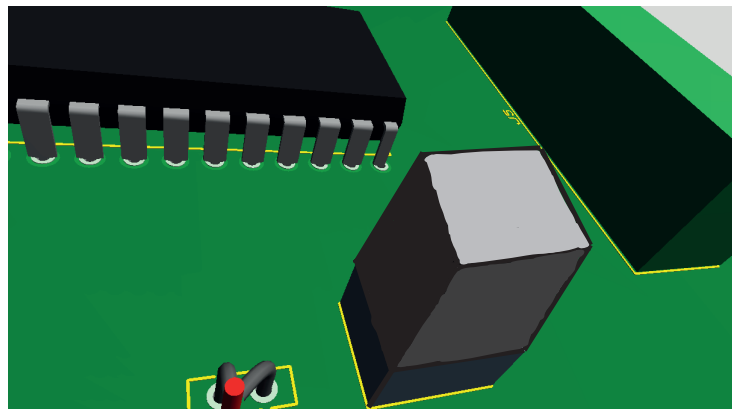


Figura A.7.: Configuración con jumper.

Conexión 1.

Al estimulador acústico de terminales “molex” de 3 pines [Figura A.3](#) cuenta con cuatro cables, pero solo se deben conectar tres de los cables como se muestra en el diagrama de conexión de la [Figura A.8](#).

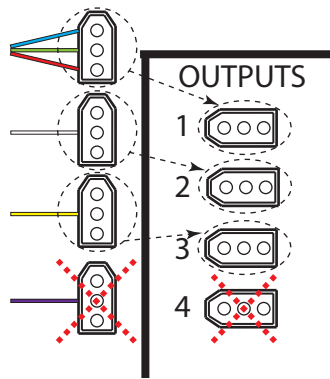


Figura A.8.: Conexión, terminales “molex” en configuración para 7 frecuencias.

Conexión 2.

El estimulador acústico de terminales metálicas tipo “zapata” ([Figura A.5](#)) cuenta con seis cables, pero solo se deben de conectar cinco de los alambres [Tabla A.5](#).

Color de alambre	Uso
Azul	Tierra
Rojo	28 [V]
Verde	Bit
Blanco	Bit
Amarillo	Bit

Tabla A.5 Conexión de alambres

Control de frecuencias.

La **Tabla A.6** muestra las combinaciones lógicas de los bits para obtener las siete frecuencias disponibles y el estado de espera.

Frecuencia	Color del alambre			
Kilo Hertz (KHz)	Morado	Amarillo	Blanco	Verde
Esperando	0	0	0	0
1	0	0	0	1
1.5	0	0	1	0
2	0	0	1	1
2.5	0	1	0	0
3	0	1	0	1
3.5	0	1	1	0
4	0	1	1	1

Tabla A.6 Combinaciones para frecuencias.

B. Curvas características eléctricas.

Diodo zener

De las hojas de especificaciones del diodo zener PTZ5.1B de ROHM CO. obtenemos la gráfica “*electrical characteristic curves*”, la cual nos muestra como se comporta la corriente inversa en función del voltaje sobre el diodo.

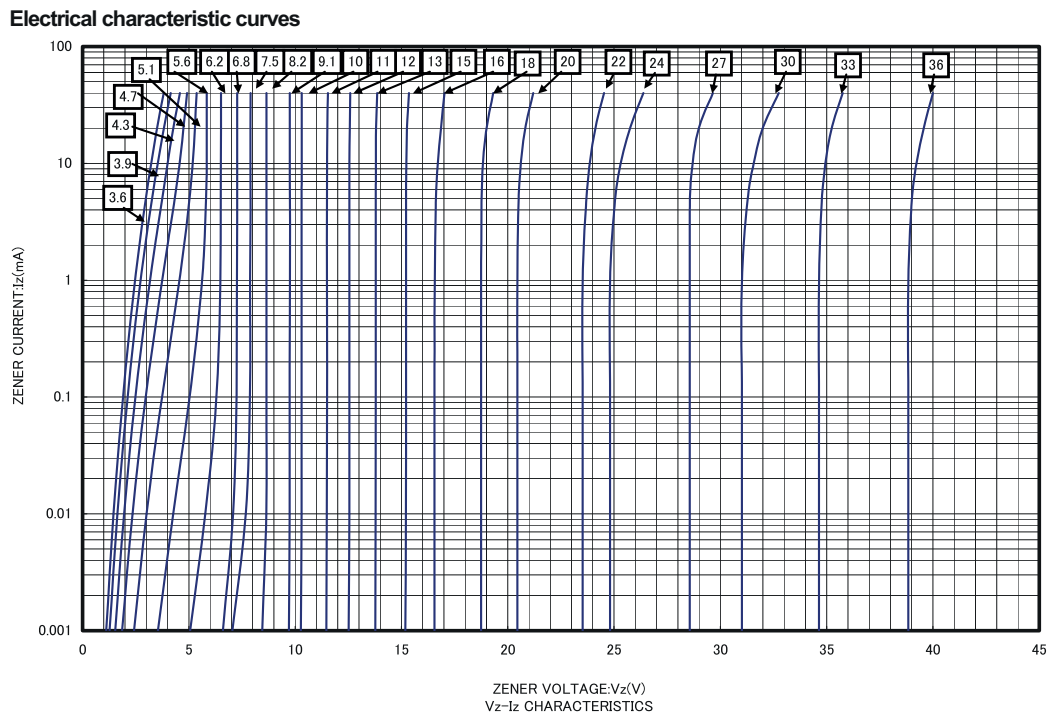


Figura B.1.: Gráfica de curvas a temperatura ambiente $T_a = 25^\circ\text{C}$

Podemos ver que cada tensión de ruptura tiene su propia curva, pero nosotros al querer un estándar de tensión $5[V]$, solo estudiaremos cuatro diferentes tensiones de ruptura: $4.7[V]$, $5.1[V]$, $5.6[V]$ y $6.2[V]$.

Corrientes inversas de diodos zener.

Las siguientes gráficas muestran la tensión zener que se tendrán al aplicar cierta corriente inversa.

Corriente inversa a $1[mA]$.

Electrical characteristic curves ($T_a=25\text{ C}$)

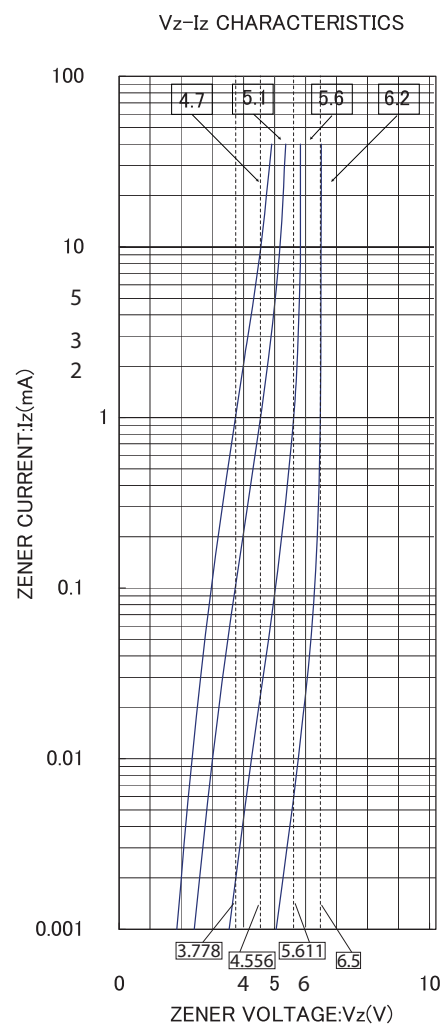


Figura B.2.: Corriente inversa a $1[mA]$.

Corriente inversa a 11[mA].

Electrical characteristic curves (Ta=25 C)

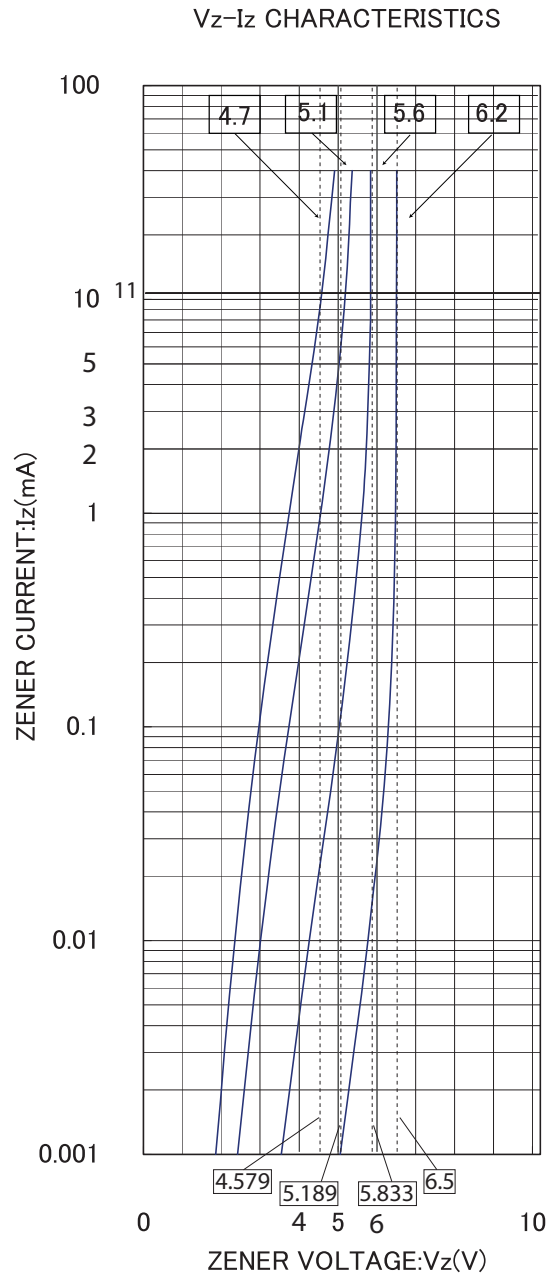


Figura B.3.: Corriente inversa a 11[mA].

Corriente inversa a $4[mA]$.

Electrical characteristic curves ($T_a=25\text{ C}$)

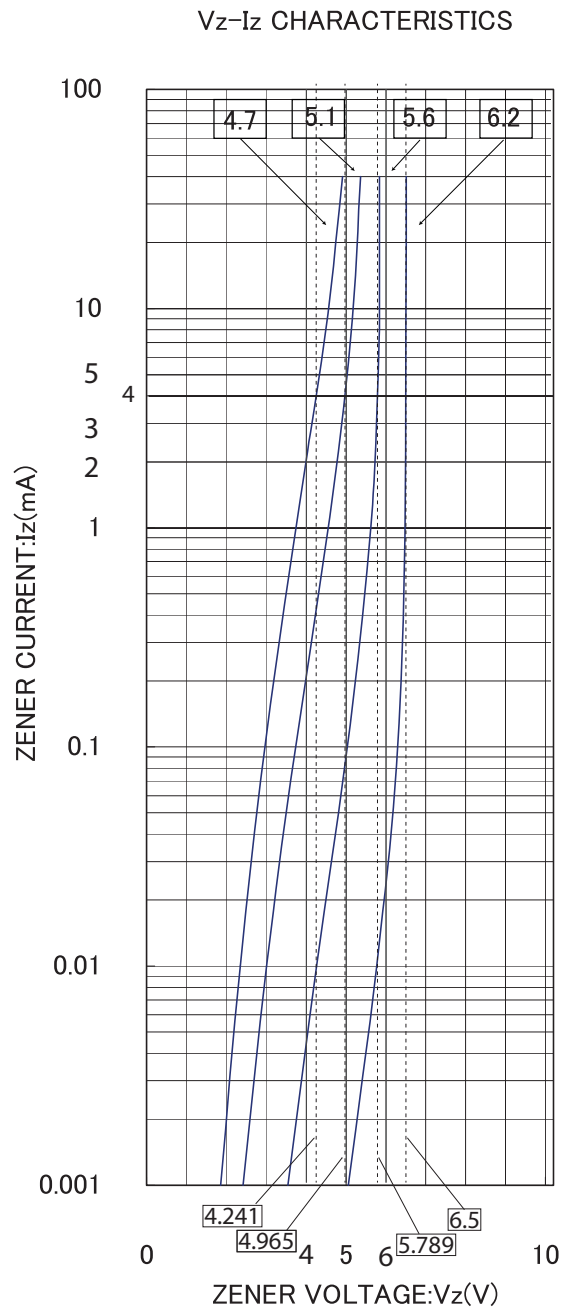


Figura B.4.: Corriente inversa a $4[mA]$.

Corriente inversa a $40[mA]$.

Electrical characteristic curves ($T_a=25\text{ C}$)

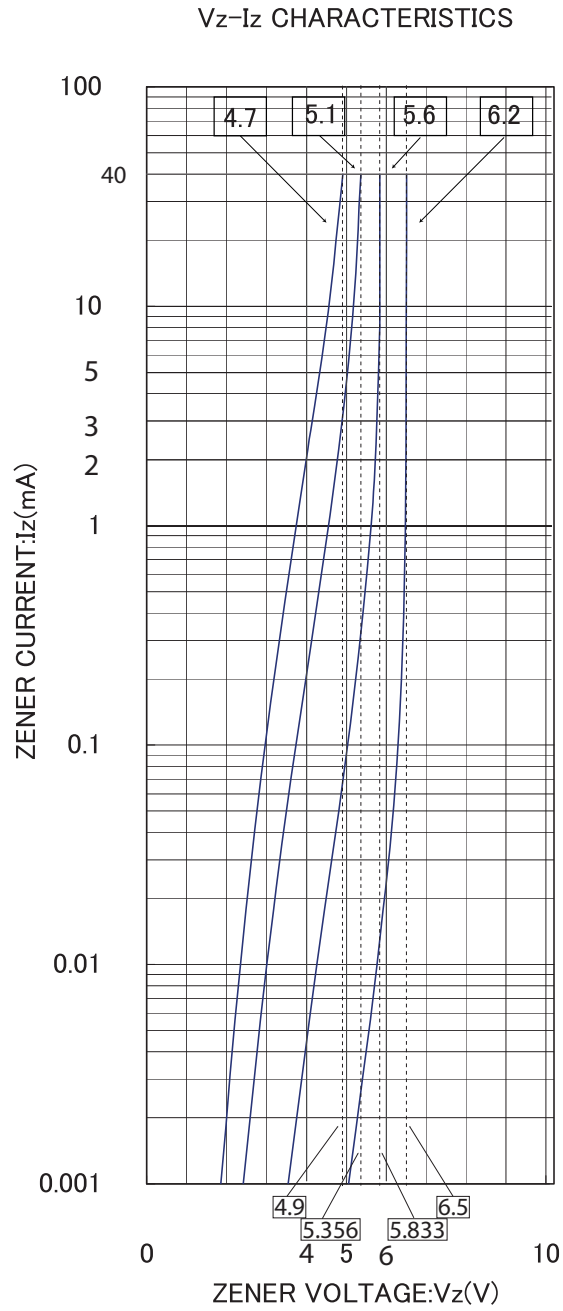


Figura B.5.: Corriente inversa a $40[mA]$.

C. Códigos fuente.

Código para generar gráficas del análisis de espectros.

```
clear all      %Limpia pantalla
grid on       %Activa cuadrícula
grid minor

L=1000;
Fs=200000;
T=1/Fs;
t=(0:L-1)*T;
N0=1024;
f0 =1000;     %frecuencia de salida

i = 100;      %Incrementa longitud de la señal
M = 1020;
N = 1020;
T = 0:1:1020; %Lóngitud Máxima de Registro de Almacenamiento
n = 0:13:1020*i; %Índice
D = (2*pi)/N; %Delta en radianes

%Número en hexadecimal para insertar en el programa del uC

SenoD = 127*sin(2*pi*f0*t)+127; % Valores de Alturas de la Funcion Seno,

Seno1 = 127*sin((2*pi*n)/N)+129; % Valores de Alturas de la Funcion Seno,
Seno2 = 127*sin((2*pi*T)/M)+127;

Y=fft(SenoD,N0);
Pyy=Y.*conj(Y)/N0;
f=(Fs/N0)*(0:N0/2-1);
grid minor

subplot(3,1,1), plot(f,Seno1,'r'); %Imprime señal sinusoidal interpolada
TITLE('Señal reconstruida en el dominio del tiempo')
AXIS([0 200 0 260])
grid on
subplot(3,1,1), plot(f,Seno2,'b');
TITLE('Señal generada con todos los valores de la tabla de consulta lookup')
AXIS([0 1020 0 260])
grid minor
subplot(3,1,2), stem(f,Pyy(1:N0/2),'filled');
TITLE('Análisis espectral de la señal reconstruida')
AXIS([0 21000 0 5000000])
grid minor
```


Código fuente de arduino para comparación.

```
//*****  
//FRECUENCIA DE 1.5 KHZ  
if ( digitalRead(MED1) == HIGH &&  
    digitalRead(MED2) == HIGH &&  
    digitalRead(MED3) == LOW &&  
    digitalRead(MED4) == HIGH )  
{  
    freq=1.5;  
    pantalla5();  
    do {  
        digitalWrite(MCLK, HIGH);  
        delayMicroseconds(319);  
        digitalWrite(MCLK, LOW);  
        delayMicroseconds(319);  
    }  
    while ( digitalRead(MED1) == HIGH &&  
           digitalRead(MED2) == HIGH &&  
           digitalRead(MED3) == LOW &&  
           digitalRead(MED4) == HIGH  
           );  
}  
//*****  
//FRECUENCIA DE 2 KHZ  
if ( digitalRead(MED1) == HIGH &&  
    digitalRead(MED2) == HIGH &&  
    digitalRead(MED3) == LOW &&  
    digitalRead(MED4) == LOW )  
{  
    freq=2;  
    pantalla();  
    do {  
        digitalWrite(MCLK, HIGH);  
        delayMicroseconds(235);  
        digitalWrite(MCLK, LOW);  
        delayMicroseconds(235);  
    }  
    while ( digitalRead(MED1) == HIGH &&  
           digitalRead(MED2) == HIGH &&  
           digitalRead(MED3) == LOW &&  
           digitalRead(MED4) == LOW  
           );  
}  
//*****  
//FRECUENCIA DE 2.5 KHZ  
if ( digitalRead(MED1) == HIGH &&  
    digitalRead(MED2) == LOW &&  
    digitalRead(MED3) == HIGH &&  
    digitalRead(MED4) == HIGH )  
{  
    freq=2;  
    pantalla5();  
    do {  
        digitalWrite(MCLK, HIGH);  
        delayMicroseconds(185);  
        digitalWrite(MCLK, LOW);  
        delayMicroseconds(185);  
    }  
    while ( digitalRead(MED1) == HIGH &&  
           digitalRead(MED2) == LOW &&  
           digitalRead(MED3) == HIGH &&  
           digitalRead(MED4) == HIGH  
           );  
}  
//*****  
//FRECUENCIA DE 3 KHZ  
if ( digitalRead(MED1) == HIGH &&  
    digitalRead(MED2) == LOW &&  
    digitalRead(MED3) == HIGH &&  
    digitalRead(MED4) == LOW )  
{  
    freq=3;  
    pantalla();  
    do {  
        digitalWrite(MCLK, HIGH);  
        delayMicroseconds(150);  
        digitalWrite(MCLK, LOW);  
        delayMicroseconds(150);  
    }  
    while ( digitalRead(MED1) == HIGH &&  
           digitalRead(MED2) == LOW &&  
           digitalRead(MED3) == HIGH &&  
           digitalRead(MED4) == LOW  
           );  
}
```

```
//*****
```

```
//FRECUENCIA DE 1.5 KHZ
if ( digitalRead(MED1) == HIGH &&
    digitalRead(MED2) == HIGH &&
    digitalRead(MED3) == LOW &&
    digitalRead(MED4) == HIGH)
{
    freq=1.5;
    pantalla5();
    do {
        digitalWrite(MCLK, HIGH);
        delayMicroseconds(319);
        digitalWrite(MCLK, LOW);
        delayMicroseconds(319);
    }
    while ( digitalRead(MED1) == HIGH &&
        digitalRead(MED2) == HIGH &&
        digitalRead(MED3) == LOW &&
        digitalRead(MED4) == HIGH
        );
}
//*****
```

```
//*****
```

```
//FRECUENCIA DE 2 KHZ
if ( digitalRead(MED1) == HIGH &&
    digitalRead(MED2) == HIGH &&
    digitalRead(MED3) == LOW &&
    digitalRead(MED4) == LOW )
{
    freq=2;
    pantalla();
    do {
        digitalWrite(MCLK, HIGH);
        delayMicroseconds(235);
        digitalWrite(MCLK, LOW);
        delayMicroseconds(235);
    }
    while ( digitalRead(MED1) == HIGH &&
        digitalRead(MED2) == HIGH &&
        digitalRead(MED3) == LOW &&
        digitalRead(MED4) == LOW
        );
}
//*****
```

```
//*****
```

```
//FRECUENCIA DE 2.5 KHZ
if ( digitalRead(MED1) == HIGH &&
    digitalRead(MED2) == LOW &&
    digitalRead(MED3) == HIGH &&
    digitalRead(MED4) == HIGH )
{
    freq=2;
    pantalla5();
    do {
        digitalWrite(MCLK, HIGH);
        delayMicroseconds(185);
        digitalWrite(MCLK, LOW);
        delayMicroseconds(185);
    }
    while ( digitalRead(MED1) == HIGH &&
        digitalRead(MED2) == LOW &&
        digitalRead(MED3) == HIGH &&
        digitalRead(MED4) == HIGH
        );
}
//*****
```

```
//*****
```

```
//FRECUENCIA DE 3 KHZ
if ( digitalRead(MED1) == HIGH &&
    digitalRead(MED2) == LOW &&
    digitalRead(MED3) == HIGH &&
    digitalRead(MED4) == LOW )
{
    freq=3;
    pantalla();
    do {
        digitalWrite(MCLK, HIGH);
        delayMicroseconds(150);
        digitalWrite(MCLK, LOW);
        delayMicroseconds(150);
    }
    while ( digitalRead(MED1) == HIGH &&
        digitalRead(MED2) == LOW &&
        digitalRead(MED3) == HIGH &&
        digitalRead(MED4) == LOW
        );
}
//*****
```

Códigos fuente.

```
//*****  
//FRECUENCIA DE 3.5 KHZ  
if ( digitalRead(MED1) == HIGH &&  
    digitalRead(MED2) == LOW &&  
    digitalRead(MED3) == LOW &&  
    digitalRead(MED4) == HIGH )  
{  
    freq=3;  
    pantalla5();  
    do {  
        digitalWrite(MCLK, HIGH);  
        delayMicroseconds(128);  
        digitalWrite(MCLK, LOW);  
        delayMicroseconds(128);  
    }  
    while ( digitalRead(MED1) == HIGH &&  
        digitalRead(MED2) == LOW &&  
        digitalRead(MED3) == LOW &&  
        digitalRead(MED4) == HIGH  
        );  
}  
//*****  
//FRECUENCIA DE 4 KHZ  
if ( digitalRead(MED1) == HIGH &&  
    digitalRead(MED2) == LOW &&  
    digitalRead(MED3) == LOW &&  
    digitalRead(MED4) == LOW )  
{  
    freq=4;  
    pantalla();  
    do {  
        digitalWrite(MCLK, HIGH);  
        delayMicroseconds(111);  
        digitalWrite(MCLK, LOW);  
        delayMicroseconds(111);  
    }  
    while ( digitalRead(MED1) == HIGH &&  
        digitalRead(MED2) == LOW &&  
        digitalRead(MED3) == LOW &&  
        digitalRead(MED4) == LOW  
        );  
}  
//*****  
//FRECUENCIA DE 5 KHZ  
if ( digitalRead(MED1) == LOW &&  
    digitalRead(MED2) == HIGH &&  
    digitalRead(MED3) == HIGH &&  
    digitalRead(MED4) == HIGH )  
{  
    freq=5;  
    pantalla();  
    do {  
        digitalWrite(MCLK, HIGH);  
        delayMicroseconds(88);  
        digitalWrite(MCLK, LOW);  
        delayMicroseconds(87);  
    }  
    while ( digitalRead(MED1) == LOW &&  
        digitalRead(MED2) == HIGH &&  
        digitalRead(MED3) == HIGH &&  
        digitalRead(MED4) == HIGH  
        );  
}  
//*****  
//FRECUENCIA DE 6 KHZ  
if ( digitalRead(MED1) == LOW &&  
    digitalRead(MED2) == HIGH &&  
    digitalRead(MED3) == HIGH &&  
    digitalRead(MED4) == LOW )  
{  
    freq=6;  
    pantalla();  
    do {  
        digitalWrite(MCLK, HIGH);  
        delayMicroseconds(70);  
        digitalWrite(MCLK, LOW);  
        delayMicroseconds(70);  
    }  
    while ( digitalRead(MED1) == LOW &&  
        digitalRead(MED2) == HIGH &&  
        digitalRead(MED3) == HIGH &&  
        digitalRead(MED4) == LOW  
        );  
}
```

```

//*****
//FRECUENCIA DE 8 KHZ
if ( digitalRead(MED1) == LOW &&
    digitalRead(MED2) == HIGH &&
    digitalRead(MED3) == LOW &&
    digitalRead(MED4) == HIGH )
{
    freq=8;
    pantalla();
    do {
        digitalWrite(MCLK, HIGH);
        delayMicroseconds(50);
        digitalWrite(MCLK, LOW);
        delayMicroseconds(50);
    }
    while ( digitalRead(MED1) == LOW &&
        digitalRead(MED2) == HIGH &&
        digitalRead(MED3) == LOW &&
        digitalRead(MED4) == HIGH
        );
}
//*****

//FRECUENCIA DE 10 KHZ
if ( digitalRead(MED1) == LOW &&
    digitalRead(MED2) == HIGH &&
    digitalRead(MED3) == LOW &&
    digitalRead(MED4) == LOW )
{
    freq=10;
    pantalla();
    do {
        digitalWrite(MCLK, HIGH);
        delayMicroseconds(44);
        digitalWrite(MCLK, LOW);
        delayMicroseconds(43);
    }
    while ( digitalRead(MED1) == LOW &&
        digitalRead(MED2) == HIGH &&
        digitalRead(MED3) == LOW &&
        digitalRead(MED4) == LOW
        );
}

//*****
//FRECUENCIA DE 12 KHZ
if ( digitalRead(MED1) == LOW &&
    digitalRead(MED2) == LOW &&
    digitalRead(MED3) == HIGH &&
    digitalRead(MED4) == HIGH )
{
    freq=12;
    pantalla();
    do {
        digitalWrite(MCLK, HIGH);
        delayMicroseconds(38);
        digitalWrite(MCLK, LOW);
        delayMicroseconds(38);
    }
    while ( digitalRead(MED1) == LOW &&
        digitalRead(MED2) == LOW &&
        digitalRead(MED3) == HIGH &&
        digitalRead(MED4) == HIGH
        );
}
//*****

//FRECUENCIA DE 15 KHZ
if ( digitalRead(MED1) == LOW &&
    digitalRead(MED2) == LOW &&
    digitalRead(MED3) == HIGH &&
    digitalRead(MED4) == LOW )
{
    freq=15;
    pantalla();
    do {
        digitalWrite(MCLK, HIGH);
        delayMicroseconds(30);
        digitalWrite(MCLK, LOW);
        delayMicroseconds(29);
    }
    while ( digitalRead(MED1) == LOW &&
        digitalRead(MED2) == LOW &&
        digitalRead(MED3) == HIGH &&
        digitalRead(MED4) == LOW
        );
}

```

```

//*****
//FRECUENCIA DE 18 KHZ
if ( digitalRead(MED1) == LOW &&
    digitalRead(MED2) == LOW &&
    digitalRead(MED3) == LOW &&
    digitalRead(MED4) == HIGH )
{
    freq=18;
    pantalla();
    do {
        digitalWrite(MCLK, HIGH);
        delayMicroseconds(10);
        digitalWrite(MCLK, LOW);
        delayMicroseconds(8);
    }
    while ( digitalRead(MED1) == LOW &&
        digitalRead(MED2) == LOW &&
        digitalRead(MED3) == LOW &&
        digitalRead(MED4) == HIGH
        );
}
//*****
//FRECUENCIA DE 20 KHZ
if ( digitalRead(MED1) == LOW &&
    digitalRead(MED2) == LOW &&
    digitalRead(MED3) == LOW &&
    digitalRead(MED4) == LOW )
{
    freq=20;
    pantalla();
    do {
        digitalWrite(MCLK, HIGH);
        digitalWrite(MCLK, LOW);
    }
    while ( digitalRead(MED1) == LOW &&
        digitalRead(MED2) == LOW &&
        digitalRead(MED3) == LOW &&
        digitalRead(MED4) == LOW
        );
}
}

```

Código fuente de Mikro C para las pruebas de comparación.

```

char *freq;
void pantalla0()
{
  Lcd_Out(1,1," LISTO ");
  Lcd_Out(2,1," Esperando2 ");
}
void pantalla()
{ Lcd_Cmd(LCD_CLEAR),
  Lcd_Cmd(Lcd_CURSOR_OFF),
  Lcd_Out(1,4,"Frecuencia");
  Lcd_Out(2,6,freq);
  Lcd_Out(2,10,"KHz"); }
void pantalla2()
{ Lcd_Cmd(LCD_CLEAR),
  Lcd_Cmd(Lcd_CURSOR_OFF),
  Lcd_Out(1,4,"Frecuencia");
  Lcd_Out(2,6,freq);
  Lcd_Out(2,9,"Hz"); }
void main()
{ PCON = 0;
  ANSEL = 0;
    ANSELH = 0;
    TRISA = 0xC0;
  TRISB = 0xFF;
  TRISC = 0x00;
  TRISD = 0x00;
  TRISE = 0;
  Sound_Init(&PORTC,3);
  Lcd_Init(&PORTD);
  Lcd_Cmd(LCD_CLEAR); //Limpia la pantalla
  Lcd_Cmd(LCD_CURSOR_OFF);
  pantalla0();
  PORTC.F0 = 1;
  while(1)
  { switch (PORTB)
    { case 0xF0: while (PORTB == 0xF0)
      {
        pantalla0();
      }
      case 0xE0: freq = "1";
        pantalla(); //Frecuencia de 1 KHz
        while (PORTB == 0xE0){
          PORTC.F3 = 1;
          Delay_us(500);
          PORTC.F3 = 0 ;
          Delay_us(500); }
        break;
      case 0xD0: freq = "1.5",
        pantalla(); //Frecuencia 1.5 KHz
        while (PORTB == 0xD0){
          PORTC.F3 = 1;
          Delay_us(333);
          PORTC.F3 = 0 ;
          Delay_us(333); }
        break;
      case 0xC0: freq = "2",
        pantalla(); //Frecuencia de 2 KHz
        while (PORTB == 0xC0){
          PORTC.F3 = 1;
          Delay_us(250);
          PORTC.F3 = 0 ;
          Delay_us(250); }
        break;
      case 0xB0: freq = "2.5",
        pantalla(); //Frecuencia de 2.5 KHz
        while (PORTB == 0xB0){
          PORTC.F3 = 1;
          Delay_us(200);
          PORTC.F3 = 0 ;
          Delay_us(200); }
        break;
      case 0xA0: freq = "3",
        pantalla(); //Frecuencia de 3 KHz
        while (PORTB == 0xA0){
          PORTC.F3 = 1;
          Delay_us(166);
          PORTC.F3 = 0 ;
          Delay_us(166); }
        break;
      case 0x90: freq = "3.5",
        pantalla(); //Frecuencia de 3.5 KHz
        while (PORTB == 0x90){
          PORTC.F3 = 1;
          Delay_us(142);
          PORTC.F3 = 0 ;
          Delay_us(166); }
        break;
      case 0x80: freq = "4",
        pantalla(); //Frecuencia de 4 KHz
        while (PORTB == 0x80){
          PORTC.F3 = 1;
          Delay_us(125);
          PORTC.F3 = 0 ;
          Delay_us(125); }
        break;
    }
  }
}

```

Códigos fuente.

```
case 0x70: freq = "5",
    pantalla();          //Frecuencia de 5 KHz
    while (PORTB == 0x70){
        PORTC.F3 = 1;
        Delay_us(100);
        PORTC.F3 = 0 ;
        Delay_us(100);   }
    break;
case 0x60: freq = "6",
    pantalla();          //Frecuencia de 6 KHz
    while (PORTB == 0x60){
        PORTC.F3 = 1;
        Delay_us(83);
        PORTC.F3 = 0 ;
        Delay_us(83);   }
    break;
case 0x50: freq = "8",
    pantalla();          //Frecuencia de 8 KHz
    while (PORTB == 0x50){
        PORTC.F3 = 1;
        Delay_us(62);
        PORTC.F3 = 0 ;
        Delay_us(62);   }
    break;
case 0x40: freq = "10",
    pantalla();          //Frecuencia de 10 KHz
    while (PORTB == 0x40){
        PORTC.F3 = 1;
        Delay_us(55);
        PORTC.F3 = 0 ;
        Delay_us(55);   }
    break;
case 0x30: freq = "12",
    pantalla();          //Frecuencia de 12 KHz
    while (PORTB == 0x30){
        PORTC.F3 = 1;
        Delay_us(50);
        PORTC.F3 = 0 ;
        Delay_us(50);   }
    break;
case 0x20: freq = "15",
    pantalla();          //Frecuencia de 15 KHz
    while (PORTB == 0x20){
        PORTC.F3 = 1;
        Delay_us(41);
        PORTC.F3 = 0 ;
        Delay_us(41);   }
    break;
case 0x10: freq = "18",
    pantalla();          //Frecuencia de 18 KHz
    while (PORTB == 0x10){
        PORTC.F3 = 1;
        Delay_us(20);
        PORTC.F3 = 0 ;
        Delay_us(20);   }
    break;
case 0x00: freq = "20",
    pantalla();          //Frecuencia de 20 KHz
    while (PORTB == 0x00){
        PORTC.F3 = 1;
        Delay_us(10);
        PORTC.F3 = 0 ;
        Delay_us(10);   }
    break;
    }
}
```

Código fuente para el microcontrolador.

Códigos fuente.


```
const unsigned char trianX[]=  
{ 1,1,1,2,2,3,3,4,4,5,5,6,6,7,7,8,8,9,9,10,10,11,11,12,12,13,13,14,14,15,15,  
16,16,17,17,18,18,19,19,20,20,21,21,22,22,23,23,24,24,25,25,26,26,27,27,28,28,  
29,29,30,30,31,31,32,32,33,33,34,34,35,35,36,36,37,37,38,38,39,39,40,40,41,41,  
42,42,43,43,44,44,45,45,46,46,47,47,48,48,49,49,50,50,51,51,52,52,53,53,54,54,  
55,55,56,56,57,57,58,58,59,59,60,60,61,61,62,62,63,63,64,64,65,65,66,66,67,67,  
68,68,69,69,70,70,71,71,72,72,73,73,74,74,75,75,76,76,77,77,78,78,79,79,80,80,  
81,81,82,82,83,83,84,84,85,85,86,86,87,87,88,88,89,89,90,90,91,91,92,92,93,93,  
94,94,95,95,96,96,97,97,98,98,99,99,100,100,101,101,102,102,103,103,104,104,  
105,105,106,106,107,107,108,108,109,109,110,110,111,111,112,112,113,113,114,  
114,115,115,116,116,117,117,118,118,119,119,120,120,121,121,122,122,123,123,  
124,124,125,125,126,126,127,127,128,128,128,129,129,130,130,131,131,132,132,  
133,133,134,134,135,135,136,136,137,137,138,138,139,139,140,140,141,141,142,  
142,143,143,144,144,145,145,146,146,147,147,148,148,149,149,150,150,151,151,  
152,152,153,153,154,154,155,155,156,156,157,157,158,158,159,159,160,160,161,  
161,162,162,163,163,164,164,165,165,166,166,167,167,168,168,169,169,170,170,  
171,171,172,172,173,173,174,174,175,175,176,176,177,177,178,178,179,179,180,  
180,181,181,182,182,183,183,184,184,185,185,186,186,187,187,188,188,189,189,  
190,190,191,191,192,192,193,193,194,194,195,195,196,196,197,197,198,198,199,  
199,200,200,201,201,202,202,203,203,204,204,205,205,206,206,207,207,208,208,  
209,209,210,210,211,211,212,212,213,213,214,214,215,215,216,216,217,217,218,  
218,219,219,220,220,221,221,222,222,223,223,224,224,225,225,226,226,227,227,  
228,228,229,229,230,230,231,231,232,232,233,233,234,234,235,235,236,236,237,  
237,238,238,239,239,240,240,241,241,242,242,243,243,244,244,245,245,246,246,  
247,247,248,248,249,249,250,250,251,251,252,252,253,253,254,254,255,254,254,  
253,253,252,252,251,251,250,250,249,249,248,248,247,247,246,246,245,245,244,  
244,243,243,242,242,241,241,240,240,239,239,238,238,237,237,236,236,235,235,  
234,234,233,233,232,232,231,231,230,230,229,229,228,228,227,227,226,226,225,  
225,224,224,223,223,222,222,221,221,220,220,219,219,218,218,217,217,216,216,  
215,214,214,213,213,212,212,211,211,210,210,209,209,208,208,207,207,206,206,  
205,205,204,204,203,203,202,202,201,201,200,200,199,199,198,198,197,197,196,  
196,195,195,194,194,193,193,192,192,191,191,190,190,189,189,188,188,187,187,  
186,186,185,185,184,184,183,183,182,182,181,181,180,180,179,179,178,178,177,  
177,176,176,175,175,174,174,173,173,172,172,171,171,170,170,169,169,168,168,  
167,167,166,166,165,165,164,164,163,163,162,162,161,161,160,160,159,159,158,  
158,157,157,156,156,155,155,154,154,153,153,152,152,151,151,150,150,149,149,  
148,148,147,147,146,146,145,145,144,144,143,143,142,142,141,141,140,140,139,  
138,138,137,137,136,136,135,135,134,134,133,133,132,132,131,131,130,130,129,  
129,128,128,128,127,127,126,126,125,125,124,124,123,123,122,122,121,121,120,  
120,119,119,118,118,117,117,116,116,115,115,114,114,113,113,112,112,111,111,  
110,110,109,109,108,108,107,107,106,106,105,105,104,104,103,103,102,102,101,  
101,100,100,99,99,98,98,97,97,96,96,95,95,94,94,93,93,92,92,91,91,90,90,89,  
89,88,88,87,87,86,86,85,85,84,84,83,83,82,82,81,81,80,80,79,79,78,78,77,77,  
76,76,75,75,74,74,73,73,72,72,71,71,70,70,69,69,68,68,67,67,66,66,65,65,64,  
64,63,63,62,62,61,61,60,60,59,59,58,58,57,57,56,56,55,55,54,54,53,53,52,52,  
51,51,50,50,49,49,48,48,47,47,46,46,45,45,44,44,43,43,42,42,41,41,40,40,39,  
39,38,38,37,37,36,36,35,35,34,34,33,33,32,32,31,31,30,30,29,29,28,28,27,27,  
26,26,25,25,24,24,23,23,22,22,21,21,20,20,19,19,18,18,17,17,16,16,15,15,14,  
14,13,13,12,12,11,11,10,10,9,9,8,8,7,7,6,6,5,5,4,4,3,3,2,2,1,1,1,  };
```

```
//*****CONSTANTES*****  
  
int i=0;  
int A;  
const unsigned char *freq;  
const unsigned char *freqP;  
const unsigned char *freq2;  
  
//*****FUNCIÓN SENO*****  
void senoTE()  
{  
    for (i=0x00;i<0x3FC; i+=A) //i+=A  
        PORTB=senoX[i];  
}  
//*****FUNCION TRIANGULAR*****  
void trian()  
{  
    for (i=0x00;i<0x3FC; i+=A) //i+=A  
        PORTB=trianX[i];  
}  
  
//*****FUNCIONES DE PANTALLA EN ESPERA*****  
void pantalla0C()  
{  
    Lcd_Out(1,1,"Listo, Esperando");  
    Lcd_Out(2,2," Onda Cuadrada ");  
  
}  
void pantalla0S()  
{  
    Lcd_Out(1,1,"Listo, Esperando");  
    Lcd_Out(2,2,"Onda Sinusoidal");  
}  
void pantalla0T()  
{  
    Lcd_Out(1,1,"Listo, Esperando");  
    Lcd_Out(2,2,"Onda Triangular");  
}  
void pantallaERR()  
{  
    Lcd_Out(1,2," Combinacion ");  
    Lcd_Out(2,2," No Valida ");  
}
```

Códigos fuente.

```

//*****FUNCIONES DE PANTALLA*****
void pantallaC()
{
    Lcd_Cmd(LCD_CLEAR),
    Lcd_Cmd(Lcd_CURSOR_OFF),
    Lcd_Out(1,3,"Frec:");
    Lcd_Chr(1,9, freq);
    Lcd_Chr(1,10,freqP);
    Lcd_Chr(1,11,freq2);
    Lcd_Out(1,13, "KHz");
    Lcd_Out(2,2, "Onda Cuadrada");
}
void pantallaS()
{
    Lcd_Cmd(LCD_CLEAR),
    Lcd_Cmd(LCD_CURSOR_OFF),
    Lcd_Out(1,3,"Frec:");
    Lcd_Chr(1,9, freq);
    Lcd_Chr(1,10,freqP);
    Lcd_Chr(1,11,freq2);
    Lcd_Out(1,13, "KHz");
    Lcd_Out(2,2, "Onda Sinusoidal");
}
void pantallaT()
{
    Lcd_Cmd(LCD_CLEAR),
    Lcd_Cmd(LCD_CURSOR_OFF),
    Lcd_Out(1,3,"Frec:");
    Lcd_Chr(1,9, freq);
    Lcd_Chr(1,10,freqP);
    Lcd_Chr(1,11,freq2);
    Lcd_Out(1,12, "KHz");
    Lcd_Out(2,2,"Onda Triangular");
}

```

```

****FRECUENCIAS Y SALTO ****
void frec1()
{
  freq = ' ';
  freqP = '1';
  freq2 = '1';
  A=0x0A;
}
void frec2()
{
  freq = '1';
  freqP = " ";
  freq2 = '5';
  A=0x0F;
}
void frec3()
{
  freq = ' ';
  freqP = '2';
  freq2 = '1';
  A=0x14;
}
void frec4()
{
  freq = '2';
  freqP = " ";
  freq2 = '5';
  A=0x19; // +1
}
void frec5()
{
  freq = ' ';
  freqP = '3';
  freq2 = '1';
  A=0x1E; // +1
}
void frec6()
{
  freq = '3';
  freqP = " ";
  freq2 = '5';
  A=0x24; // +2
}
void frec7()
{
  freq = ' ';
  freqP = '4';
  freq2 = '1';
  A=0x28; } // +2
void frec8()
{
  freq = ' ';
  freqP = '5';
  freq2 = '1';
  A=0x33; //+3
}
void frec9()
{
  freq = ' ';
  freqP = '6';
  freq2 = '1';
  A=0x3D; //+3
}
void frec10()
{
  freq = ' ';
  freqP = '8';
  freq2 = '1';
  A=0x56; //+9
}
void frec11()
{
  freq = ' ';
  freqP = '1';
  freq2 = '0';
  A=0x6F; //+15
}
void frec12()
{
  freq = ' ';
  freqP = '1';
  freq2 = '1';
  A=0x7F; // +15
}
void frec13()
{
  freq = ' ';
  freqP = '1';
  freq2 = '3';
  A=0x8C; // +15
}
void frec14()
{
  freq = ' ';
  freqP = '1';
  freq2 = '4';
  A=0x97; //+11
}
void frec15()
{
  freq = ' ';
  freqP = '1';
  freq2 = '6';
  A=0xBF; //+26
}

```

Códigos fuente.

```
//*****MAIN*****
void main()

{

    PCON = 0;
    ANSEL = 0;
    ANSELH = 0;
    TRISA = 0x00;      //PUERTO PARA DISPLAY
    TRISB = 0x00;      //PUERTO PARA DAC08xx o Arreglo R2R
    TRISC = 0xFF;      //PUERTO PARA LEER ENTRADAS DE MED
    TRISD = 0x00;      //PUERTO PARA ONDA CUADRADA Y PROPÓSITO GENERAL
    TRISE = 0x07;      //PUERTO PARA SELECCIONAR TIPO DE ONDA

    Lcd_Config(&PORTA, 0, 1, 6, 2, 3, 4, 5);
    Lcd_Init(&PORTA);
    Lcd_Cmd(LCD_CLEAR);      //Limpia la pantalla
    Lcd_Cmd(LCD_CURSOR_OFF);

    while(1)
    {

    switch (PORTE)
    {

    case 0x00:

        while (PORTE == 0x00 )
            {
            switch (PORTC)
            {

            case 0x0F:
                while ( PORTC == 0x0F && PORTE == 0x00 )
                {
                pantalla0S();      //ESPERANDO ONDA SINUSOIDAL *****
                }
            }
            }
        }
    }
}
```

```
case 0x0E: frec1();
    pantallaS();          //Frecuencia de 1 KHz  Sinusoidal
    while (PORTE == 0x00, PORTC == 0x0E)
    {
        senoTE();
    }
case 0x0D: frec2();
    pantallaS();          //Frecuencia de 1.5 KHz  Sinusoidal
    while (PORTE == 0x00, PORTC == 0x0D)
    {
        senoTE();
    }

case 0x0C: frec3();
    pantallaS();          //Frecuencia de 2 KHz  Sinusoidal
    while (PORTE == 0x00, PORTC == 0x0C)
    {
        senoTE();
    }

case 0x0B: frec4();
    pantallaS();          //Frecuencia de 2.5 KHz  Sinusoidal
    while (PORTE == 0x00, PORTC == 0x0B )
    {
        senoTE();
    }
case 0x0A: frec5();
    pantallaS();          //Frecuencia de 3 KHz  Sinusoidal
    while (PORTE == 0x00, PORTC == 0x0A )
    {
        senoTE();
    }
case 0x09: frec6();
    pantallaS();          //Frecuencia de 3.5 KHz  Sinusoidal
    while (PORTE == 0x00, PORTC == 0x09)
    {
        senoTE();
    }
case 0x08: frec7();
    pantallaS();          //Frecuencia de 4 KHz  Sinusoidal
    while (PORTE == 0x00, PORTC == 0x08)
    {
        senoTE();
    }
case 0x07: frec8();
    pantallaS();          //Frecuencia de 5 KHz  Sinusoidal
    while (PORTE == 0x00, PORTC == 0x07)
    { senoTE(); }
```



```
case 0x06: frec9();
    pantallaS();          //Frecuencia de 6 KHz Sinusoidal
    while (PORTE == 0x00, PORTC == 0x06)
    {
        senoTE();
    }
case 0x05: frec10();
    pantallaS();          //Frecuencia de 8 KHz Sinusoidal
    while (PORTE == 0x00, PORTC == 0x05)
    {
        senoTE();
    }
case 0x04: frec11();
    pantallaS();          //Frecuencia de 10 KHz Sinusoidal
    while (PORTE == 0x00, PORTC == 0x04)
    {
        senoTE();
    }
case 0x03: frec12();
    pantallaS();          //Frecuencia de 11 KHz Sinusoidal
    while (PORTE == 0x00, PORTC == 0x03)
    {
        senoTE();
    }
case 0x02: frec13();
    pantallaS();          //Frecuencia de 13 KHz Sinusoidal
    while (PORTE == 0x00, PORTC == 0x02)
    {
        senoTE();
    }
case 0x01: frec14();
    pantallaS();          //Frecuencia de 14 KHz Sinusoidal
    while (PORTE == 0x00, PORTC == 0x01)
    {
        senoTE();
    }
case 0x00: frec15();
    pantallaS();          //Frecuencia de 16 KHz Sinusoidal
    while (PORTE == 0x00, PORTC == 0x00)
    {
        senoTE();
    }
}
}
```

```
case 0x01:
  while (PORTE == 0x01 )
  {
    switch (PORTC)
    {

      case 0x0F:
        while ( PORTC == 0x0F && PORTE == 0x01)
        {
          pantalla0C();      //ESPERANDO ONDA CUADRADA
        }
      case 0x0E: frec1();
        pantallaC();      //Frecuencia de 1 KHz Cuadrada
        while (PORTE == 0x01, PORTC == 0x0E)
        {
          PORTD.F5 = 1 ;
          Delay_us(500);
          PORTD.F5 = 0 ;
          Delay_us(500);
        }
      case 0x0D: frec2();
        pantallaC();      //Frecuencia de 1.5 KHz Cuadrada
        while (PORTE == 0x01, PORTC == 0x0D)
        {
          PORTD.F5 = 1;
          Delay_us(333);
          PORTD.F5 = 0 ;
          Delay_us(333);
        }
      case 0x0C: frec3();
        pantallaC();      //Frecuencia de 2 KHz Cuadrada
        while (PORTE == 0x01, PORTC == 0x0C)
        {
          PORTD.F5 = 1;
          Delay_us(250);
          PORTD.F5 = 0 ;
          Delay_us(250);
        }
      case 0x0B: frec4();
        pantallaC();      //Frecuencia de 2.5 KHz Cuadrada
        while (PORTE == 0x01, PORTC == 0x0B)
        {
          PORTD.F5 = 1;
          Delay_us(200);
          PORTD.F5 = 0 ;
          Delay_us(200);
        }
    }
  }
}
```

```
case 0x0A: frec5();
    pantallaC();          //Frecuencia de 3 KHz Cuadrada
    while (PORTE == 0x01, PORTC == 0x0A)
    { PORTD.F5 = 1;
      Delay_us(166);
      PORTD.F5 = 0;
      Delay_us(166); }
case 0x09: frec6();
    pantallaC();          //Frecuencia de 3.5 KHz Cuadrada
    while (PORTE == 0x01, PORTC == 0x09)
    {
      PORTD.F5 = 1;
      Delay_us(142);
      PORTD.F5 = 0;
      Delay_us(142);
    }
case 0x08: frec7();
    pantallaC();          //Frecuencia de 4 KHz Cuadrada
    while (PORTE == 0x01, PORTC == 0x08)
    {
      PORTD.F5 = 1;
      Delay_us(125);
      PORTD.F5 = 0;
      Delay_us(125);
    }
case 0x07: frec8();
    pantallaC();          //Frecuencia de 5 KHz Cuadrada
    while (PORTE == 0x01, PORTC == 0x07)
    { PORTD.F5 = 1;
      Delay_us(100);
      PORTD.F5 = 0;
      Delay_us(100); }
case 0x06: frec9();
    pantallaC();          //Frecuencia de 6 KHz Cuadrada
    while (PORTE == 0x01, PORTC == 0x06)
    { PORTD.F5 = 1;
      Delay_us(83);
      PORTD.F5 = 0;
      Delay_us(83); }
case 0x05: frec10();
    pantallaC();          //Frecuencia de 8 KHz Cuadrada
    while (PORTE == 0x01, PORTC == 0x05)
    { PORTD.F5 = 1;
      Delay_us(62);
      PORTD.F5 = 0;
      Delay_us(62); }
```

```
case 0x04: frec11();
    pantallaC();          //Frecuencia de 10 KHz Cuadrada
    while (PORTE == 0x01, PORTC == 0x04)
    {
        PORTD.F5 = 1;
        Delay_us(50);
        PORTD.F5 = 0;
        Delay_us(50);
    }
case 0x03: frec12();
    pantallaC();          //Frecuencia de 11 KHz Cuadrada
    while (PORTE == 0x01, PORTC == 0x03)
    {
        PORTD.F5 = 1;
        Delay_us(45);
        PORTD.F5 = 0;
        Delay_us(45);
    }
case 0x02: frec13();
    pantallaC();          //Frecuencia de 13 KHz Cuadrada
    while (PORTE == 0x01, PORTC == 0x02)
    {
        PORTD.F5 = 1;
        Delay_us(26);
        PORTD.F5 = 0;
        Delay_us(26);
    }
case 0x01: frec14();
    pantallaC();          //Frecuencia de 14 KHz Cuadrada
    while (PORTE == 0x01, PORTC == 0x01)
    {
        PORTD.F5 = 1;
        Delay_us(36);
        PORTD.F5 = 0;
        Delay_us(36);
    }
case 0x00: frec15();
    pantallaC();          //Frecuencia de 16 KHz Cuadrada
    while (PORTE == 0x01, PORTC == 0x00)
    {
        PORTD.F5 = 1;
        Delay_us(31);
        PORTD.F5 = 0;
        Delay_us(31);
    }
}
}
```

```
/**
 *
 */
case 0x02:
    while (PORTE == 0x02 )
    {
        switch (PORTC)
        {

        case 0x0F:
            while (PORTC == 0x0F && PORTE == 0x02 )
            {
                pantalla0T();          //ESPERANDO ONDA TRIANGULAR
            }
        case 0x0E: frec1();
            pantallaT();          //Frecuencia de 1 KHz Triangular
            while (PORTE == 0x02, PORTC == 0x0E)
            {
                trian();
            }
        case 0x0D: frec2();
            pantallaT();          //Frecuencia de 1.5 KHz Triangular
            while (PORTE == 0x02, PORTC == 0x0D)
            {
                trian();
            }
        case 0x0C: frec3();
            pantallaT();          //Frecuencia de 2 KHz Triangular
            while (PORTE == 0x02, PORTC == 0x0C)
            {
                trian();
            }
        case 0x0B: frec4();
            pantallaT();          //Frecuencia de 2.5 KHz Triangular
            while (PORTE == 0x02, PORTC == 0x0B)
            {
                trian();
            }
        case 0x0A: frec5();
            pantallaT();          //Frecuencia de 3 KHz Triangular
            while (PORTE == 0x02, PORTC == 0x0A)
            {
                trian();
            }
        case 0x09: frec6();
            pantallaT();          //Frecuencia de 3.5 KHz Triangular
            while (PORTE == 0x02, PORTC == 0x09)
            {
                trian();
            }
        }
    }
}
```

```

case 0x08: frec7();
    pantallaT();          //Frecuencia de 4 KHz Triangular
    while (PORTE == 0x02, PORTC == 0x08)
    {   trian();   }
case 0x07: frec8();
    pantallaT();          //Frecuencia de 5 KHz Triangular
    while (PORTE == 0x02, PORTC == 0x07)
    {   trian();   }
case 0x06: frec9();
    pantallaT();          //Frecuencia de 6 KHz Triangular
    while (PORTE == 0x02, PORTC == 0x06)
    {   trian();   }
case 0x05: frec10();
    pantallaT();          //Frecuencia de 8 KHz Triangular
    while (PORTE == 0x02, PORTC == 0x05)
    {   trian();   }
case 0x04: frec11();
    pantallaT();          //Frecuencia de 10 KHz Triangular
    while (PORTE == 0x02, PORTC == 0x04)
    {   trian();   }
case 0x03: frec12();
    pantallaT();          //Frecuencia de 11 KHz Triangular
    while (PORTE == 0x02, PORTC == 0x03)
    {   trian();   }
case 0x02: frec13();
    pantallaT();          //Frecuencia de 13 KHz Triangular
    while (PORTE == 0x02, PORTC == 0x02)
    {   trian();   }
case 0x01: frec14();
    pantallaT();          //Frecuencia de 14 KHz Triangular
    while (PORTE == 0x02, PORTC == 0x01)
    {   trian();   }
case 0x00: frec15();
    pantallaT();          //Frecuencia de 16 KHz Triangular
    while (PORTE == 0x02, PORTC == 0x00)
    {   trian();   } } } }

//*****
case 0x03:
    while (PORTE == 0x03 )
    {
        pantallaERR();
    }
}
}
}

```


D. Diseño del filtro paso-bajas para la interpolación.

Introducción.

Con el filtro pretendemos eliminar frecuencias no deseadas, esto con la finalidad de "suavizar" las señales, en otras palabras, quitar cierta cantidad de contenido armónico para obtener señales más parecidas a las ideales.

Diseño del filtro.

El filtro tendrá una configuración paso bajas como se aprecia en la [Figura D.1](#) para poder controlar de una mejor forma el contenido armónico deseado según sea la señal seleccionada.

Únicamente las señales sinusoidales y triangulares harán uso del filtro paso bajas, debido a que en la señal rectangular debe tener el mayor contenido armónico a comparación de las dos anteriores. Al rango de frecuencias que mantiene su magnitud unitaria constante se le llama banda de paso, mientras que a las frecuencias cuya magnitud es cero se les llama banda suprimida.

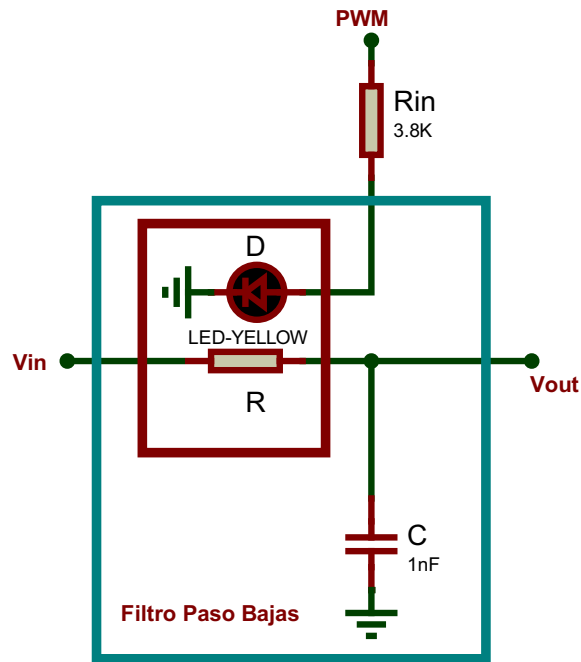


Figura D.1.: Configuración de filtro paso baja.

La banda de paso en esta configuración de filtro, estará dada por la frecuencia de corte superior y la frecuencia de corte inferior, donde sabemos que estas frecuencias comienzan a tener atenuación de $-3[dB]$ o $\frac{1}{\sqrt{2}}$ en adelante y están dadas por:

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

Obteniendo la respuesta en frecuencia como lo muestra la [Figura D.2.](#)

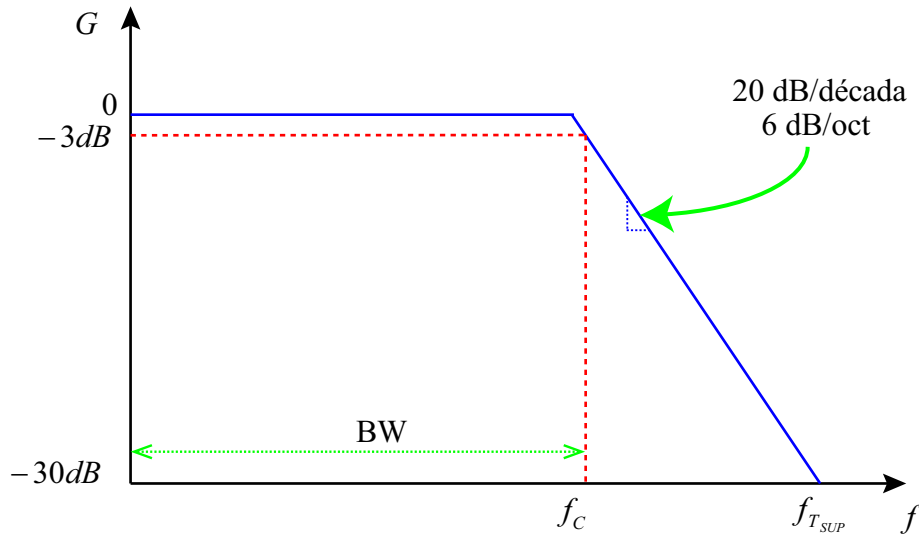


Figura D.2.: Respuesta en frecuencia de un filtro paso bajas.

Implementar un filtro paso bajas con frecuencias de corte fija para ambas señales en nuestro dispositivo propuesto no es una opción adecuada. Se configurara al filtro para que sus frecuencias de corte aumenten o disminuyan según sea el caso necesario.

La manera en que haremos que el filtro recorra su frecuencia de corte, será variando la resistencia del filtro a partir de un opto-acoplador foto resistivo controlado por modulación de ancho de pulso¹.

¹Dispositivo diseñado y fabricado especialmente para este proyecto.

Diseño del filtro paso-bajas para la interpolación.

Ganancia unitaria.

Es la frecuencia a la cual la amplitud es máxima, o su magnitud es de 0 [dB] o 1.

Para que la magnitud de la función de transferencia sea igual a uno:

$$|H(j\omega)| = \frac{1}{\sqrt{1 + (\omega RC)^2}} = \frac{1}{1}$$

se debe de cumplir que:

$$(\omega RC)^2 = 0$$

Por lo tanto, en la frecuencia donde tendremos la máxima amplitud será $\omega = 0 \left[\frac{\text{rad}}{\text{s}} \right]$

Función de transferencia.

Obteniendo así la función de transferencia del filtro:

$$H(s) = \frac{v_{out}}{v_I} = \left(\frac{1}{(sCR + 1)} \right)$$

Sabemos que $s = j\omega$, de tal manera que la magnitud de la función de transferencia es:

$$|H(j\omega)| = \frac{1}{\sqrt{1 + (\omega RC)^2}}$$

Y su fase:

$$\phi(H(j\omega)) = \tan^{-1} \left(\frac{j\omega RC}{1} \right) = -\tan^{-1}(\omega RC)$$

Algunas características importantes que podemos obtener de la magnitud de la función de transferencia son: ganancia unitaria y frecuencia de corte.

Frecuencia de corte.

Es aquella frecuencia donde el circuito tiene una atenuación de -3 [dB] o $\frac{1}{\sqrt{2}}$

Diseño del filtro paso-bajas para la interpolación.

$$|H(j\omega)| = \frac{1}{\sqrt{1 + (\omega RC)^2}} = \frac{1}{\sqrt{2}}$$

Por lo que:

$$(\omega RC)^2 = 1$$

$$\omega = \frac{1}{RC}$$

Caracterización del opto-acoplador resistivo controlado por modulación de ancho de pulso.

La configuración para poder utilizar este dispositivo y la que se ocupó para caracterizarlo se muestra en la [Figura D.4](#).

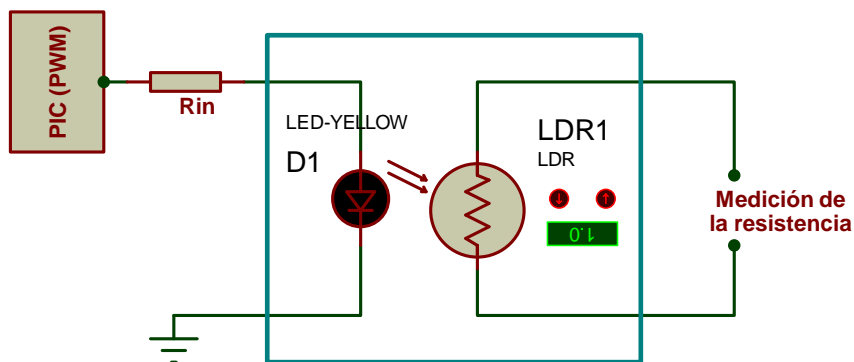


Figura D.4.: Conexión para caracterización del opto-acoplador.

Como ya se ha mencionado el PWM será controlado por el PIC16F887, por lo tanto el PWM puede dar de 0 a 255 ciclos de trabajo diferentes cambiando este valor en el argumento de la función *Pwm_Change_Duty()* que esta en el programa.

Los valores obtenidos se muestran gráficamente en la [Figura D.5](#) y numéricamente en la [Figura D.6](#).

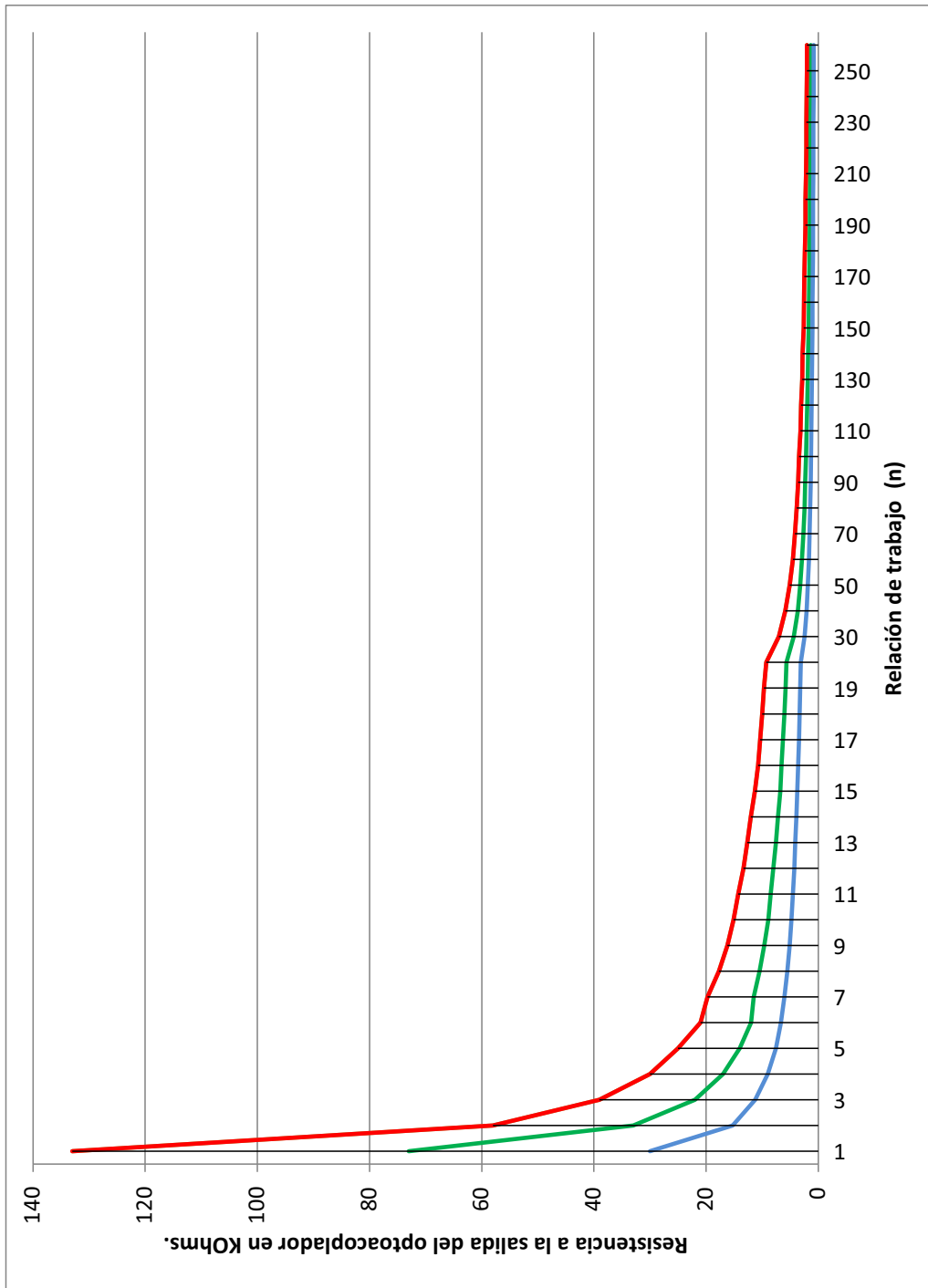


Figura D.5.: Gráfica de valores resistivos en el opto-acoplador.

Diseño del filtro paso-bajas para la interpolación.

n	Ciclo de trabajo en %	Rin = 3.8 K	Rin = 2.2 K	Rin = 1 K
1	0.39	133	73	30
2	0.78	58	33	15.28
3	1.18	39	22	11.25
4	1.57	30	17	9
5	1.96	25	14	7.54
6	2.35	21	12	6.64
7	2.75	19.74	11.52	6.03
8	3.14	17.70	10.48	5.49
9	3.53	16.20	9.62	5.08
10	3.92	15.11	8.9	4.76
11	4.31	14.28	8.46	4.54
12	4.71	13.33	8	4.28
13	5.10	12.67	7.57	4.08
14	5.49	12.03	7.17	3.9
15	5.88	11.30	6.78	3.73
16	6.27	10.70	6.54	3.57
17	6.67	10.35	6.32	3.45
18	7.06	10.00	6.05	3.34
19	7.45	9.67	5.85	3.23
20	7.84	9.28	5.66	3.12
30	11.76	7.00	4.38	2.46
40	15.69	5.86	3.66	2.09
50	19.61	5.12	3.22	1.86
60	23.53	4.54	2.91	1.68
70	27.45	4.15	2.66	1.55
80	31.37	3.83	2.45	1.45
90	35.29	3.59	2.31	1.36
100	39.22	3.42	2.19	1.3
110	43.14	3.18	2.06	1.22
120	47.06	3.05	1.97	1.17
130	50.98	2.87	1.87	1.12
140	54.90	2.78	1.81	1.08
150	58.82	2.61	1.72	1.04
160	62.75	2.55	1.67	1.01
170	66.67	2.47	1.61	0.98
180	70.59	2.40	1.57	0.95
190	74.51	2.29	1.52	0.92
200	78.43	2.26	1.49	0.91
210	82.35	2.19	1.44	0.88
220	86.27	2.15	1.42	0.87
230	90.20	2.11	1.39	0.85
240	94.12	2.05	1.36	0.83
250	98.04	2.03	1.35	0.828
255	100.00	2.01	1.33	0.819

Figura D.6.: Valores de resistencias obtenidas en la prueba.

E. Tarjeta fenólica PCB.

El diseño de la tarjeta fenólica (Figura E.1, Figura E.2) esta hecho con base a varios aspectos que son: dimensiones de la carcasa, elementos electrónicos que interactúan con el usuario.

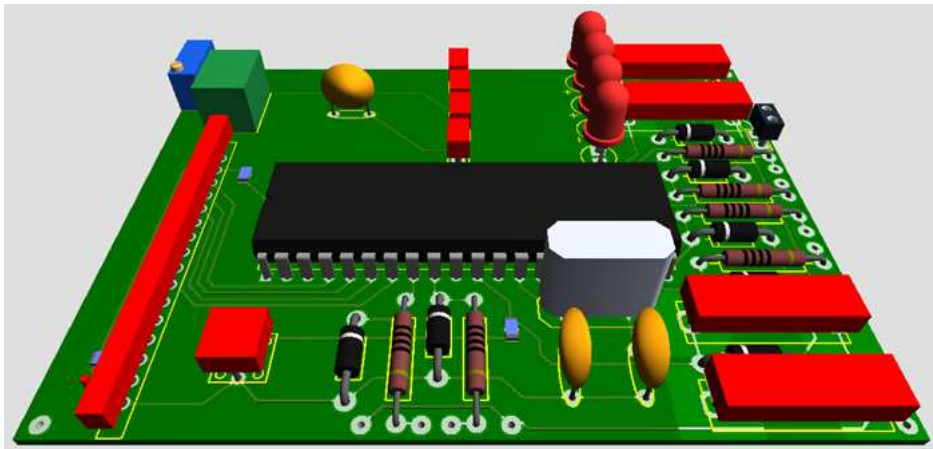


Figura E.1.: Visualización en tercera dimensión de la tarjeta.

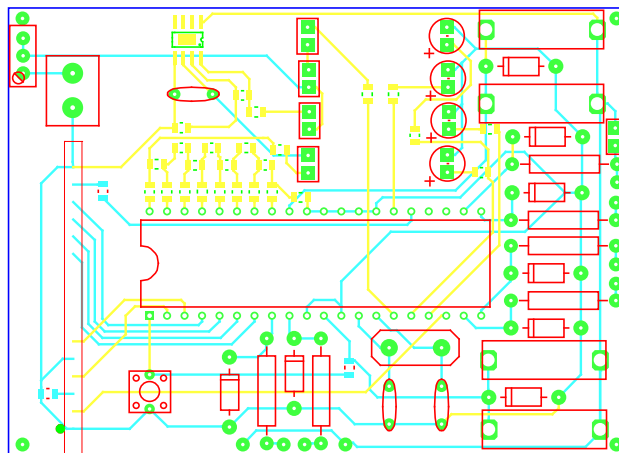


Figura E.2.: Diseño de la tarjeta fenólica.

Carcasa

La carcasa del circuito de la marca "*Really Useful Boxes*" fue adquirida en la tienda de autoservicio "lumen", con el nombre "caja de plástico *really 03*" **Figura E.3**. Las características de esta caja de plástico rígido con tapa y asas laterales son,[38].:

Dimensiones externas:120[mm] de largo, 85[mm] de ancho y 65[mm] de profundidad.

Dimensiones internas:90[mm] de largo, 65[mm] de ancho y 55[mm] de profundidad.

Peso: 54[g]



Figura E.3.: Caja de plástico *really 03*

Al medir las dimensiones internas, nos percatamos que el fabricante nos da medidas aproximadas, ya que las medidas que nosotros adquirimos son de dos a tres milímetros mas grandes.

Material

La siguiente Tabla E.1 muestra el material que se necesita y el código para pedirlo en la tienda "AG electrónica S.A de C.V." [8]. Los precios que se muestran son en moneda nacional por cada pieza y con I.V.A., algunos dispositivos los venden en paquetes con un mínimo de piezas.

Cantidad	Dispositivo	No. Parte AG	Precio	Pzas. mínimas
1	PIC16F887-I/P	PIC16F887-I/P	\$51	1
1	Base 40 pines maquilados	40PF	\$15	1
1	LM358D	LM358D	\$3	1
2	Capacitor cerámico 15pF	cc-15/1000v	\$2	5[**]
1	<i>Mini switch push</i>	PUSH-1/2P	\$2	5[**]
1	<i>Header hembra, 1 línea, 11 barrenos</i>	HOUSING-40	\$10	1
1	<i>Cristal mini20[MHz]</i>	20MHZ-MINI	\$10	1
4	Foto-resistencia 10[MΩ]	GL5549	\$3	5[**]
4	LED amarillo plano	LED-Y5F-AC-F30	\$3	5[**]
1	Trimpot cuadrado 20vueltas, 1[kΩ]	68WR1K	\$17	1
1	Capacitor cerámico 1[ηF]	CC-.001/1000V	\$1	10[**]
1	Pantalla LCD	JHD-162G-B/W	\$145	1
9	Resistor 13.7[kΩ] _{1/8} [W] R-2R	RMS-13K7	\$2	5[*,**]
7	Resistor 6.81[kΩ] _{1/8} [W] R-2R	RMS-6K81	\$2	5[*,**]
1	Resistor 9.53[kΩ] _{1/8} [W] reset	RMS-9K53	\$2	5[**]
1	Resistor 2.2[kΩ] _{1/8} [W] display VEE	RMS-2K2	\$2	5[**]
1	Resistor 330[Ω] _{1/8} [W] display LED	RMS-330E	\$2	5[**]
1	Resistor 1[kΩ] _{1/8} [W] opam2	RMS-1K	\$2	5[**]
1	DIP Switch dos posiciones	DIP-2	\$5	1
1	<i>Header ángulo recto, 1 línea, 11 pines.</i>	HEADER-1AR	\$5	1
1	Puente para header (<i>Jumper</i>)	JUMPER	\$2	5[**]
1	<i>Header, 1 línea, 2 terminales</i>	HEADER-1	\$3	5[**]
9	Diodo zener 5.1[V], 1[W]	1N4733A	\$2	10[**]
4	1[KΩ], 2[W]		\$3.50	1
1	<i>Tweeter</i>		\$30	
6	Zapatillas metálicas		\$2	1
6	Pines metálicos		\$1	1
1.5[metro] por color de alambre		Por metro	\$2.5	
	Azul	Rojo	Amarillo	Termofith
	Verde	Blanco	Morado	

Tabla E.1

[*] Comprar dos paquetes

[**] Sobra material

Bibliografía

- [1] P. Malvino, Principios de electrónica, Editorial McGraw-Hill, 6° edición, 2000.
- [2] C. J. Savant, Martin S. Roden, Gordon L. Carpenter, Diseño Electrónico: Circuitos y Sistemas, Editorial Pearson Educacion, 3° edición ilustrada, 2000.
- [3] Jack R. Smith, Programming the PIC microcontroller with MBASIC. Editorial Newnes, 1° edición, 2005.
- [4] Milan Verle, PIC Microcontrollers - Programming in C, Editorial mickroElektro-nika, 1° edición, 2009.
- [5] Stephen M. Nolan, Jose M. Soltero, Understandig and interpreting Standard-Logic Data Sheets, Application Report SZZA036B, Mayo 2003.
- [6] Roberto Heyer, Tecnologías de armado de módulos electrónicos, Editorial Dunken, 2° edición, 2009.
- [7] Ángel Bueno Martín, Ana I. de Soto Gorroño, Desarrollo y construcción de prototipos electrónicos, Editorial Marcombo, 2005.
- [8] Tienda virtual AG. <http://www.agelectronica.com/inicio.htm> , junio 2012.
- [9] Catálogo en línea, Newark-México. <http://mexico.newark.com/>, junio 2012.
- [10] PIC16F887 Data sheet. <http://ww1.microchip.com/downloads/en/DeviceDoc/41291F.pdf>, junio 2012.
- [11] zener diode PTZ5.1B Data sheet, ROHM Co., LTD. 2007.
- [12] Display 16x2 Dot Grande STN Azul Back Light Blanco, AG electrónica S.A de C.V., 2010.
- [13] zener Voltage Regulator Diodes, 1N4728A Series Technical Data, Motorola Semiconductor , 2007.
- [14] Rodríguez Medrano, Cesar, Rodríguez Medrano, Rubén; Neurootofisiología y Audiología Clínica, Editorial, McGraw-Hill, 2003.
- [15] J.L. Slimobich, P.E. Chacón; Lacan: La marca del leer. Psicología, Editorial Anthropos , 2002.
- [16] A. Jackson-Menaldi, María Cristina; La voz normal; Editorial Médica Panamericana, 1992.

- [17] <http://www.med-associates.com>, junio, 2012.
- [18] Alan V. Oppenheim, Señales y Sistemas, Editorial Prentice Hall, 2° edición, 1997.
- [19] Proakis, D. Manolakis, Digital Signal Processing, Editorial Prentice Hall, 3° edición, 1996.
- [20] ATmega328P Datasheet. <http://www.atmel.com/Images/doc8161.pdf>, junio, 2012.
- [21] <http://www.mikroe.com>, junio, 2012.
- [22] <http://www.arduino.cc/>, junio, 2012.
- [23] Texas Instruments, Precision Digital Sine-Wave Generation with the TMS32010, 1984.
- [24] <https://www.national.com/ds/LM/LM158.pdf>, junio, 2012..
- [25] <http://www.fairchildsemi.com/ds/LM/LM7805.pdf> , junio, 2012.
- [26] <http://sdcpsicologia.blogspot.mx/2011/03/descripcion-de-los-laboratorios.html>, junio, 2012.
- [27] Jack R Smith, Modern Communication Circuits, Editorial McGraw-Hill Science, 2° edición, 1997.
- [28] Kenneth K. Clarke, Communication Circuits: Analysis and Design, Editorial Krieger Pub Co, 2° edición, 1994.
- [29] Adel S. Sedra, Kenneth Carless Smith, Circuitos microelectrónicos, Editorial McGraw-Hill, 4° edición, 2006.
- [30] Milan Verle, PIC Microcontrollers - Programming in C, editorial mikroElektronika, 1° edición 2009.
- [31] Datasheet RAS (General Control Relays)
http://www.sunhold.com/product_2.php?SNO2=42&SNO3=58&Vcode=23.
- [32] Timothy J. Maloney, Electronica Industrial Moderna, Editorial Pearson Educación, 5° edición, 2006.
- [33] Pulsar Developments Ltd., Piezo tweeter application note, 2005.
<http://www.pulsardevelopments.com/products/detail/piezoan.html>.
- [34] Mark N. Horenstein , Microelectronic Circuit and Devices, Editorial Prentice Hall, 2° edición, 1995.
- [35] HCF4051B/52B/53B Datasheet. <http://www.agspecinfo.com/pdfs/H/HCF4052B.PDF>
- [36] LED-Y5F-AC-F30 Datasheet. <http://www.agspecinfo.com/pdfs/L/LEDY5FYT.PDF>
- [37] GL5549 Datasheet. <http://www.agspecinfo.com/pdfs/G/GL5549.PDF>
- [38] http://www.reallyusefulproducts.co.uk/uk/html/onlineshop/rub/b00_3litre.php

[39] <http://www.agspecinfo.com/pdfs/P/PUSH-12P.PDF>

[40] <http://www.agspecinfo.com/pdfs/D/DIP-XX.PDF>