



UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

DISEÑO DE UN AFINADOR ELECTROMECAÁNICO PARA INSTRUMENTOS DE
CUERDA CON SISTEMA SINFÍN-CORONA

MEMORIA PARA OPTAR AL TÍTULO DE INGENIERO CIVIL ELECTRICISTA

CAMILO ANDRÉS SALAZAR RIQUELME

PROFESOR GUÍA:
MARTIN ADAMS

MIEMBROS DE LA COMISIÓN:
OSVALDO MUÑOZ PERALTA
JOSÉ SALINAS HUERTA

SANTIAGO DE CHILE
2015

ABSTRACT

This document corresponds to the final thesis completed in order to obtain the Electrical Engineering degree from the University of Chile. This work consists of the design of an electro-mechanic tuner for string instruments that use a worm-rim gear mechanism in their tuning system. This proposed tuner, in addition to identifying the frequency played by the instrument, must turn the screw of the tuning machine in order to adjust the tension of the string to make it vibrate at the desired frequency, so that it plays the correct tone.

In this report, several conceptual topics involved in the previous stages of the project are explained: First, the physics related to the sound production phenomena in a string instrument, in addition to the characteristics of a worm-rim gear are presented. The way in which a sound is classified as a musical note is also explained. Existing tools used to capture and digitalize sound and the restrictions to correctly reconstruct an electrical signal are also studied in this report. Some mathematical concepts which are useful to identify the frequency of a digital signal are presented. This report also mentions, as the state of the art, several solutions which are available today to tune a musical instrument.

To approach the design of the device itself, this work follows a series of stages: First the reasons why a specific type of sound interface was chosen are explained. Then the construction of two essential circuits, an amplifier and an offset, in the pre-treatment of the captured signal is detailed. In addition, as part of the design, details regarding the programming process of a micro-controller, using an algorithm written in the Arduino IDE, are approached. The micro-controller is capable of accomplishing different tasks in order to identify the frequency of the captured signal. It also controls a SERVO motor, along with an LCD screen. The motor has been modified in order to make continuous 360° turns and its main rotational axis has been adapted to fit the tuning machines of the musical instruments involved in this work. The LCD screen is used as an interface to ease the interaction with the user. The device is studied as a feedback loop using the concepts of control theory.

In the last sections of this report, different results obtained using the designed device in instruments of different frequency spectrum are presented, explaining in which type of instruments the device is more effective, in addition to showing the factors that complicate the correct function of the device. Finally, it is proposed to improve the present design, as part of a future project, in order to construct a final version of the tuner, which could be presented as a product. Another idea to continue the study of this topic is to extend the theoretical concepts of the project into some other possible applications such as developing a faster frequency detection algorithm or designing frequency filters to eliminate the effects of harmonics and vibrational noise.

RESUMEN DE LA MEMORIA PARA OPTAR AL TÍTULO
DE: Ingeniero Civil Electricista.
POR: Camilo Andrés Salazar Riquelme.
FECHA: 05/12/2014.
PROFESOR GUÍA: Martin Adams.

DISEÑO DE UN AFINADOR ELECTROMECAÁNICO PARA INSTRUMENTOS DE CUERDA CON SISTEMA SINFÍN-CORONA

Este documento corresponde al informe de la memoria para optar al diploma de Ingeniero Civil Electricista de la Universidad de Chile. El presente trabajo de título consiste en el diseño de un afinador para instrumentos de cuerda que incorporen un sistema de encordado basado en engranajes sinfín-corona. El diseño propuesto, además de identificar la nota musical que está produciendo el instrumento, hace girar el sistema de clavijas del mismo y ajusta la tensión de la cuerda correspondiente para dejarla en el tono correcto deseado por el usuario.

En el presente informe se abordan distintos temas conceptuales involucrados en las etapas previas al desarrollo del proyecto: Primero se abordan los aspectos físicos que involucran la producción de sonido de un instrumento de cuerda además de las características de los engranajes sinfín-corona. Luego se explica la regla matemática que se usa en música para clasificar un sonido como nota musical. Se revisan las herramientas existentes para capturar y digitalizar sonido, y se estudian las limitantes de un sistema para reconstruir de forma correcta una señal eléctrica al digitalizarla. También se abordan algunos conceptos matemáticos útiles para identificar la frecuencia de una señal digital. Se mencionan, como estado del arte, las soluciones existentes hoy en día para afinar instrumentos musicales.

Posteriormente, abordando el diseño del dispositivo, el trabajo completa una serie de pasos: Primero se argumenta el tipo de interfaz capturadora elegida para implementar la solución. Luego se detalla la construcción de dos bloques de circuito, -un amplificador y un adicionador de componente continua- necesarios para tratar la señal capturada antes de digitalizarla. Además, como parte del diseño, se abordan los detalles informáticos que involucran la programación de un micro-controlador, usando un algoritmo construido en el entorno de programación Arduino. El micro-controlador es capaz de realizar las distintas operaciones matemáticas para determinar la frecuencia de la señal capturada y también controlar un motor SERVO, junto con una pantalla LCD. El motor SERVO ha sido modificado para girar de forma continua en 360° y su eje de rotación fue adaptado para encajar en las clavijas de los instrumentos musicales que este trabajo comprende. La pantalla LCD es utilizada como interfaz para facilitar la utilización del dispositivo por parte del usuario.

En las últimas secciones del informe se presentan distintos resultados obtenidos al usar el dispositivo diseñado en instrumentos de distinto espectro de frecuencias, explicando en qué tipo de instrumento el dispositivo es más eficaz, además de exponer los factores del sistema que entorpecen su funcionamiento. Finalmente se deja propuesto mejorar el diseño expuesto, como parte de algún otro proyecto, para construir una versión de producto final del afinador o extender los conceptos teóricos aquí presentados a otras aplicaciones posibles.

Dedicado a los estudiosos que puedan continuar o sacar algo de aquí.

A todos los músicos que luchan con sus afinadores.

Y a nadie más, porque sería absurdo dedicarlo a quién no entienda el documento.

AGRADECIMIENTOS

Representa para mi persona un grato y abismal sentir de satisfacción poder comenzar este documento agradeciendo a todos los colegas músicos que han participado conmigo en cada una de las distintas propuestas e instancias que motivaron originalmente la idea de realizar este trabajo. A mi querida Tuna Universitaria y especialmente a los adláteres pardillos, que nunca tienen afinados sus instrumentos; a la Corporación Cultural de Pulso y Púa de Chile, por haber generado los recursos para restaurar el laudón; a mis camaradas folcloristas; a mis compañeros del grupo celta; a los muchachos de Tré; y naturalmente a mis amigos jazzistas de Ya te dije Jazz, de La Martinique, y todos los demás que tanto desafío se plantean constantemente.

Agradezco al profesor guía de este trabajo de título, Dr. Martin Adams, por haber acogido mi propuesta y aceptar la realización del proyecto sin que este forme parte de su propia línea de investigación, además de haberme apoyado con sus conocimientos en control de sistemas dinámicos.

Agradezco también a mi profesor co-guía de trabajo de titulación, Osvaldo Muñoz Peralta, por demostrar gran interés en el tema propuesto, por dejar a disposición su experticia en diseño para intervenir la interfaz mecánica del afinador aquí propuesto, por involucrarse en el desarrollo del proyecto, y por darme todos los valiosos consejos recibidos en la estructuración del mismo, aspectos que fueron gran motivación para llevar a cabo esta empresa.

No puedo dejar de agradecer a todas las personas que se interesaron en este proyecto y aportaron con ideas para su desarrollo. Al grupo 2 de la sección 2 del curso EL3003 primavera 2014, que realizaron su experiencia electiva en concordancia con este trabajo; a mis alumnos de la academia de música, que tantas sugerencias y preguntas presentan incansablemente; y obviamente a los estudiantes de ingeniería, que siempre tienen la solución para todo.

Extiendo además un especial y profundo agradecimiento a los más cercanos y todas aquellas personas que no creían que este trabajo sería posible o que postulaban que mis otras actividades podrían interferir con el buen desarrollo del quehacer académico, porque simbolizan un acicate fundamental en el cumplimiento de los plazos y metas propuestas.

TABLA DE CONTENIDO

| | | |
|--------|--|----|
| 1 | INTRODUCCIÓN..... | 1 |
| 1.1 | Motivación..... | 1 |
| 1.2 | Objetivo general..... | 1 |
| 1.3 | Objetivos específicos..... | 2 |
| 1.4 | Alcances..... | 2 |
| 1.5 | Estructura del documento..... | 3 |
| 2 | REVISIÓN BIBLIOGRÁFICA..... | 4 |
| 2.1 | El sonido..... | 4 |
| 2.1.1 | Altura..... | 4 |
| 2.1.2 | Intensidad..... | 5 |
| 2.1.3 | Timbre..... | 5 |
| 2.2 | Instrumentos de cuerda..... | 5 |
| 2.3 | Notas musicales..... | 7 |
| 2.3.1 | El cent..... | 9 |
| 2.4 | El sistema de encordado sinfín-corona..... | 10 |
| 2.5 | Interfaces captadoras de sonido..... | 11 |
| 2.5.1 | Micrófono..... | 11 |
| 2.5.2 | Cápsula magnética..... | 15 |
| 2.5.3 | Transductor de cobre..... | 16 |
| 2.6 | Transformada de Fourier..... | 17 |
| 2.6.1 | Rellenar con ceros..... | 20 |
| 2.7 | Transformada de Hartley..... | 22 |
| 2.8 | Teorema del muestreo..... | 24 |
| 2.9 | Control automático..... | 26 |
| 2.10 | Estado del arte..... | 27 |
| 2.10.1 | Diapasón..... | 28 |
| 2.10.2 | Afinadores convencionales..... | 29 |
| 2.10.3 | Afinadores dedicados..... | 31 |
| 2.10.4 | Afinadores de pinza..... | 31 |
| 2.10.5 | Afinador estroboscópico..... | 32 |

| | | |
|-------|--|----|
| 3 | DISEÑO DEL DISPOSITIVO | 34 |
| 3.1 | Selección de la interfaz captadora | 34 |
| 3.1.1 | Micrófono de contacto | 36 |
| 3.2 | Amplificación de la señal capturada | 37 |
| 3.2.1 | Construcción del amplificador de señal | 39 |
| 3.3 | Agregar componente continua | 41 |
| 3.3.1 | Construcción del circuito offset | 42 |
| 3.4 | Interfaz de digitalización | 44 |
| 3.4.1 | Frecuencia de muestreo | 45 |
| 3.4.2 | Entorno de programación | 46 |
| 3.4.3 | Relojes internos del ATmega328 | 48 |
| 3.4.4 | Indicador de saturación | 51 |
| 3.5 | Identificación de la frecuencia | 54 |
| 3.5.1 | Uso de la biblioteca FHT | 55 |
| 3.6 | Motor SERVO | 58 |
| 3.6.1 | Uso de la biblioteca SERVO | 59 |
| 3.6.2 | Modificación del motor SERVO para giro continuo | 61 |
| 3.6.3 | Adaptación del motor para clavijas | 62 |
| 3.6.4 | Inclusión del motor en el montaje del dispositivo afinador | 63 |
| 3.7 | Sistema controlado | 65 |
| 3.8 | Interfaz de utilización | 66 |
| 3.8.1 | Pantalla LCD | 66 |
| 3.8.2 | Botones | 68 |
| 3.8.3 | Inclusión de interfaz de entrada y salida para uso del dispositivo. | 70 |
| 4 | RESULTADOS | 73 |
| 4.1 | Pruebas realizadas | 73 |
| 4.1.1 | Afinación de una guitarra | 73 |
| 4.1.2 | Afinación de un laúd español | 74 |
| 4.1.3 | Afinación de una bandurria | 75 |
| 4.1.4 | Afinación de un laudón | 76 |
| 4.1.5 | Afinación de un bajo eléctrico | 77 |

| | | |
|-------|--|-----|
| 4.2 | Discusión de resultados..... | 77 |
| 4.2.1 | Precisión..... | 77 |
| 4.2.2 | Número de muestras..... | 79 |
| 4.2.3 | Intensidad de giro..... | 80 |
| 4.2.4 | Tiempo de afinación..... | 81 |
| 4.2.5 | Torque insuficiente..... | 83 |
| 4.2.6 | Modos normales y armónicos..... | 83 |
| 4.2.7 | Caídas de tensión..... | 85 |
| 5 | CONCLUSION..... | 87 |
| 5.1 | Trabajo futuro propuesto..... | 88 |
| 6 | BIBLIOGRAFÍA..... | 89 |
| 7 | ANEXOS..... | 91 |
| 7.1 | Dimensiones de una clavija de guitarra GROVER..... | 91 |
| 7.2 | Datasheet SHURE PG48..... | 92 |
| 7.3 | Datasheet RODE NT-A..... | 93 |
| 7.4 | KORG CM-200..... | 94 |
| 7.5 | Datasheet RIGOL MSO4012..... | 95 |
| 7.6 | Datasheet TL082..... | 100 |
| 7.7 | Sparkfun REDBOARD..... | 103 |
| 7.8 | Datasheet ATmega328..... | 104 |
| 7.9 | Datasheet LED luminoso..... | 110 |
| 7.10 | Motor SERVO MG995..... | 111 |
| 7.11 | Datasheet LCD ADM1602K..... | 112 |
| 7.12 | Datasheet SMS/PSM..... | 115 |
| 7.13 | Encordado para guitarra D'ADDARIO EJ47 80/20..... | 118 |
| 7.14 | Encordado para laúd español LA BELLA ML450..... | 118 |
| 7.15 | Encordado para bandurria LA BELLA MB550..... | 119 |
| 7.16 | Encordado para bajo eléctrico GHS L9000..... | 119 |
| 7.17 | Código final completo..... | 120 |

INDICE DE TABLAS

| | |
|---|----|
| Tabla 2-1: Frecuencias de las notas musicales..... | 8 |
| Tabla 3-1: Posibles configuraciones para <i>Clock Select Bits</i> de un reloj interno. | 49 |
| Tabla 3-2: Funciones de la biblioteca FHT..... | 55 |
| Tabla 3-3: Funciones de la biblioteca SERVO. | 59 |
| Tabla 3-4: Funciones de la biblioteca LiquidCrystal. | 67 |
| Tabla 3-5: Funciones implementadas. | 71 |
| Tabla 4-1: Resultados al afinar una guitarra..... | 73 |
| Tabla 4-2: Resultados al afinar un laúd español. | 74 |
| Tabla 4-3: Resultados al afinar una bandurria..... | 75 |
| Tabla 4-4: Resultados al afinar un laudón..... | 76 |
| Tabla 4-5: Resultados al afinar un bajo eléctrico..... | 77 |

INDICE DE FIGURAS

| | |
|---|----|
| Figura 2-1: Cuerdas vibrantes en una guitarra..... | 6 |
| Figura 2-2: Distancia lineal entre las notas musicales..... | 8 |
| Figura 2-3: Distancia logarítmica entre las notas musicales. | 8 |
| Figura 2-4: Clavija de guitarra. | 10 |
| Figura 2-5: Micrófono dinámico. | 12 |
| Figura 2-6: Micrófono de condensador..... | 13 |
| Figura 2-7: Micrófono de piezoeléctrico. | 14 |
| Figura 2-8: Cápsula magnética. | 15 |
| Figura 2-9: Guitarra eléctrica con sus cápsulas. | 15 |
| Figura 2-10: Cápsula doble. | 15 |
| Figura 2-11: Transductor de cobre. | 16 |
| Figura 2-12: Transductor instalado en contrabajo. | 16 |
| Figura 2-13: Captura de un <i>Mi3</i> producido por una guitarra..... | 17 |
| Figura 2-14: Transformada de Fourier de la seña de un <i>Mi3</i> producido por una guitarra. | 18 |
| Figura 2-15: Intervalos equiespaciados en el dominio de la frecuencia. | 19 |
| Figura 2-16: Intervalos equiespaciados en el dominio de frecuencias altas..... | 19 |
| Figura 2-17: Captura de un acorde de Mi mayor producido por una guitarra..... | 20 |
| Figura 2-18: Transformada de Fourier de un acorde de Mi mayor. | 20 |
| Figura 2-19: Señal capturada luego de aplicar Zero Padding. | 21 |
| Figura 2-20: Transformada de Fourier luego de aplicar Zero Padding..... | 21 |
| Figura 2-21: Ejemplo de señal discreta. | 22 |
| Figura 2-22: Densidad espectral usando Fourier. | 23 |
| Figura 2-23: Densidad espectral usando Hartley. | 23 |
| Figura 2-24: Señal reconstruida con $f_s = 150\text{Hz}$ | 24 |
| Figura 2-25: Densidad espectral $f_s = 150\text{Hz}$ | 24 |
| Figura 2-26: Señal reconstruida con $f_s = 400\text{Hz}$ | 25 |
| Figura 2-27: Densidad espectral $f_s = 400\text{Hz}$ | 25 |
| Figura 2-28: Señal reconstruida con $f_s = 200\text{Hz}$ | 25 |
| Figura 2-29: Lazo de control..... | 26 |
| Figura 2-30: Ejemplo de panel de efectos que incorpora un afinador. | 27 |
| Figura 2-31: Diapasón..... | 28 |
| Figura 2-32: Afinador de aguja. | 29 |
| Figura 2-33: Afinador LED..... | 30 |
| Figura 2-34: Afinador LCD..... | 30 |
| Figura 2-35: Afinador electrónico con sistema de pinza..... | 31 |
| Figura 2-36: Afinador estroboscópico..... | 32 |
| Figura 2-37: Disco de un afinador de estroboscopio. | 33 |
| Figura 3-1: Diagrama de bloques del funcionamiento del dispositivo..... | 34 |

| | |
|--|----|
| Figura 3-2: Respuesta eléctrica de un micrófono dinámico..... | 35 |
| Figura 3-3: Respuesta eléctrica de un micrófono de condensador bidireccional..... | 35 |
| Figura 3-4: Micrófono de contacto KORG CM-200..... | 37 |
| Figura 3-5: Respuesta eléctrica del micrófono del contacto..... | 38 |
| Figura 3-6: Digitalización de la respuesta del micrófono sin previo tratamiento..... | 38 |
| Figura 3-7: Micrófono de contacto montado en clavijero de guitarra..... | 38 |
| Figura 3-8: Esquema del amplificador no inversor..... | 39 |
| Figura 3-9: Respuesta eléctrica del micrófono luego de ser amplificada..... | 40 |
| Figura 3-10: Digitalización de la respuesta del micrófono luego de ser amplificada..... | 40 |
| Figura 3-11: Señales sin amplificar (amarillo) y sin amplificar (cyan) medidas con osciloscopio..... | 41 |
| Figura 3-12: Esquema del circuito <i>offset</i> | 42 |
| Figura 3-13: Respuesta eléctrica del micrófono luego de ser amplificada y trasladada..... | 43 |
| Figura 3-14: Digitalización de la respuesta del micrófono luego de ser amplificada y trasladada..... | 43 |
| Figura 3-15: Señales de salida del amplificador (amarillo) y de salida del offset (cyan) medidas con osciloscopio..... | 43 |
| Figura 3-16: Placa de desarrollo Sparkfun REDBOARD..... | 44 |
| Figura 3-17: Ejemplo simple de código en el entorno de programación..... | 47 |
| Figura 3-18: Circuito que ejecuta las acciones del código programado..... | 47 |
| Figura 3-19: Código para medir una muestra de una señal analógica con $f_s \sim 300[Hz]$ | 50 |
| Figura 3-20: Esquema del circuito formado por el micrófono, el amplificador, el offset y el REDBOARD..... | 51 |
| Figura 3-21: Señal analógica amplificada en exceso..... | 52 |
| Figura 3-22: Digitalización distorsionada..... | 52 |
| Figura 3-23: Esquema del circuito incluyendo indicador de saturación..... | 52 |
| Figura 3-24: Código que incorpora el indicador de saturación..... | 53 |
| Figura 3-25: Umbral de detección de pulsación de la cuerda..... | 56 |
| Figura 3-26: Implementación del algoritmo para llenar el vector <i>fht_input[]</i> | 56 |
| Figura 3-27: Cálculo de la frecuencia de la señal medida..... | 57 |
| Figura 3-28: Esquema de un ejemplo de montaje entre el REDBOARD y el motor SERVO. | 59 |
| Figura 3-29: Ejemplo de uso de la biblioteca SERVO..... | 60 |
| Figura 3-30: Bobinadora para encordar guitarras..... | 62 |
| Figura 3-31: Uso de bobinadora en ukelele..... | 62 |
| Figura 3-32: Motor SERVO adaptado para mover clavijas de instrumentos musicales..... | 62 |
| Figura 3-33: Motor SERVO siendo utilizado en las clavijas de un Laudón..... | 62 |
| Figura 3-34: Función <i>freq()</i> | 63 |
| Figura 3-35: Esquema del montaje con REDBOARD, amplificador, offset y motor SERVO. | 63 |
| Figura 3-36: Control del motor usando la frecuencia de la señal medida..... | 64 |

| | |
|--|----|
| Figura 3-37: Lazo de control del dispositivo afinador. | 65 |
| Figura 3-38: Montaje del módulo LCD al REDBOARD..... | 66 |
| Figura 3-39: Código que usa la biblioteca LiquidCrystal. | 68 |
| Figura 3-40: Módulo LCD funcionando..... | 68 |
| Figura 3-41: Ejemplo de montaje usando botones. | 69 |
| Figura 3-42: Ejemplo de código que utiliza botones..... | 69 |
| Figura 3-43: Esquema del circuito final del dispositivo afinador diseñado. | 70 |
| Figura 3-44: Comportamiento de los botones. | 72 |
| Figura 4-1: Afinador electro-mecánico afinando cuarta cuerda de guitarra. | 74 |
| Figura 4-2: Dispositivo afinando segundo par de cuerdas de laúd..... | 74 |
| Figura 4-3: Dispositivo ajustando tensión de las cuerdas del laúd..... | 74 |
| Figura 4-4: Afinador electro-mecánico detectando la nota más alta que es capaz de discriminar..... | 75 |
| Figura 4-5: Laudón restaurado para usar 14 cuerdas dobles..... | 76 |
| Figura 4-6: Errores absolutos en frecuencia..... | 78 |
| Figura 4-7: Errores absolutos en escala de igual temperamento para todas las notas musicales. | 78 |
| Figura 4-8: Densidad espectral usando 256 muestras. | 79 |
| Figura 4-9: Densidad espectral usando 4096 muestras..... | 79 |
| Figura 4-10: Tiempos de afinación de cada cuerda. | 80 |
| Figura 4-11: Tiempo de afinación bandurria..... | 81 |
| Figura 4-12: Tiempo de afinación bandurria..... | 81 |
| Figura 4-13: Tiempo de afinación guitarra..... | 82 |
| Figura 4-14: Tiempo de afinación guitarra..... | 82 |
| Figura 4-15: Tiempo de afinación bajo eléctrico..... | 82 |
| Figura 4-16: Tiempo de afinación bajo eléctrico..... | 82 |
| Figura 4-17: Tamaño de clavija de bajo eléctrico e interfaz mecánica del motor. | 83 |
| Figura 4-18: Motor ejerciendo torque sobre clavija de bajo eléctrico. | 83 |
| Figura 4-19: Captura de un Fa#3 producido por un laúd..... | 84 |
| Figura 4-20: Densidad espectral de un Fa# producido por un laúd. | 84 |
| Figura 4-21: Caída de tensión del motor SERVO..... | 85 |

1 INTRODUCCIÓN

En el contexto del uso de un instrumento musical de cuerda es que se presenta el problema de lograr una afinación correcta. Cada cuerda del instrumento, al momento de vibrar, produce una nota musical específica. Por diversas razones (ya sea el desgaste natural de las cuerdas, el uso prolongado del instrumento, los cambios de temperatura u otros), la tensión de las cuerdas varía, lo que resulta en una vibración distinta a la deseada para producir la nota musical correcta. En otras palabras, las cuerdas se desafinan y el usuario debe verificar que estén correctamente afinadas cada vez que se disponga a utilizar el instrumento. Esta acción requiere destreza por parte del usuario, además de tiempo y capacidad de verificación, es decir, quién afina un instrumento necesita tener un muy buen oído y ser capaz de igualar y comparar notas e intervalos musicales con el oído.

1.1 Motivación

Hoy existen soluciones electrónicas para ayudar a un músico a afinar un instrumento determinado. Es posible encontrar afinadores electrónicos como productos en el mercado de las casas de música y tiendas relacionadas. Sin embargo, estos dispositivos sólo son capaces de identificar la nota que produce el instrumento musical en cuestión y entregar una referencia al usuario que le permite saber qué tan desafinado está el instrumento para posteriormente corregir su afinación manualmente.

En este trabajo se postula que un afinador no sólo debería cumplir con la función de identificar la nota musical según la frecuencia que está emitiendo el instrumento, sino que también debería hacer el trabajo de ajustar la tensión de la cuerda para que esta produzca la nota musical deseada por el usuario, entonces realmente se le podría llamar “afinador” a este tipo de dispositivos.

1.2 Objetivo general

El trabajo consiste en diseñar y construir un artefacto que solucione el problema de afinar rápida y eficazmente un cordófono con sistema sinfín-corona. El artefacto debe ser capaz de ajustar la tensión de cada cuerda del instrumento y dejarlo en condiciones adecuadas para su uso. Se propone como objetivo general entonces, construir un prototipo que se pueda usar; del que se puedan obtener resultados medibles; y que funcione, es decir que pueda afinar correctamente un instrumento de cuerda.

1.3 Objetivos específicos

Los objetivos específicos de este trabajo son:

- Recoger el estado del arte de las herramientas de asistencia para afinación de instrumentos musicales junto con los distintos métodos y tecnologías que usan tales herramientas, en orden de establecer elementos de juicio para tomar decisiones proyectuales.
- Establecer un vínculo entre disciplinas distintas como lo son la música; la mecánica; las matemáticas; el diseño de circuitos eléctricos; el diseño industrial; la electrónica; y la informática.
- Adaptar herramientas existentes para darle un uso contextual sobre la base de observación del estado del arte.
- Implementar una interfaz de utilización simple y amigable para el usuario, que permita interactuar con el dispositivo; ingresar variables; y obtener datos.
- Considerar autonomía en los componentes involucrados en el funcionamiento del afinador, permitiendo que este se pueda utilizar sin dificultado, con una sola mano.
- Permitir versatilidad: el afinador debe ser compatible con el mayor número de instrumentos musicales posible. Esto significa que el dispositivo debe ser capaz de detectar notas musicales de alta y de baja frecuencia.

1.4 Alcances

Este proyecto apunta a los cordófonos o instrumentos de cuerda, es decir, aquellos instrumentos que producen sonido por medio de la vibración de una o más cuerdas tensadas, obedeciendo al fenómeno físico de la cuerda vibrante. Particularmente se estudiarán aquellos cordófonos que utilicen el sistema de engranajes sinfín-corona para amarrar y tensar sus cuerdas. Ejemplos de este tipo de instrumentos son: la guitarra; el charango; el laúd español; el bajo eléctrico; el ukelele; la vihuela; el cuatro venezolano; la balalaika; entre varios otros.

También se incluye el uso de herramientas matemáticas y de control de sistemas dinámicas involucradas en el tratamiento de señales.

A pesar de que se abordan detalles que implican instrucciones secuenciales y construcción de algoritmos, no se cubren los aspectos más básicos de la programación computacional.

Este trabajo considera el diseño y construcción de un prototipo de afinador electro-mecánico, por lo que no se incluirán aspectos de producción, comercialización y modelo de negocio asociados.

1.5 Estructura del documento

En concordancia con los objetivos expuestos anteriormente, este informe se ordena de la siguiente manera:

En el segundo capítulo se presenta una revisión de las principales disciplinas involucradas en el desarrollo teórico y conceptual de este trabajo. Esto incluye aspectos del estudio de fenómenos físicos asociados a la producción y transmisión de sonido; dispositivos existentes para transformar ondas acústicas en señales eléctricas; herramientas matemáticas para el estudio de señales; condiciones necesarias para la reconstrucción de una señal; modelamiento de control de un sistema dinámico; y el estado del arte en afinadores de instrumentos musicales, junto con el aporte de este trabajo.

Una vez planteado el estado del arte, en el tercer capítulo se describen las distintas etapas del diseño del dispositivo propuesto, junto con el desarrollo de la implementación de los distintos componentes que constituyen el artefacto. Se presenta el transductor utilizado; el tratamiento de la señal previa; el proceso de digitalización de la señal y el cálculo de la frecuencia; el control mecánico que ajusta la tensión de las cuerdas; y la interfaz de utilización con la que interactúa el usuario.

En el cuarto capítulo se exponen las pruebas realizadas que permiten evaluar el diseño del dispositivo construido. Se estudia su comportamiento frente a distintos instrumentos de cuerda, su eficacia según el rango de frecuencias en el que se trabaja y se discuten los posibles factores que alteran el buen funcionamiento del afinador electro-mecánico.

Finalmente, y a partir de lo anterior, en el quinto capítulo se presentan las conclusiones y se plantean los posibles trabajos futuros relacionados a esta memoria.

2 REVISIÓN BIBLIOGRÁFICA

El presente capítulo tiene por objetivo ubicar al lector en el entorno que se desarrolla este trabajo de título, entregando los antecedentes previos y necesarios para su contextualización.

- En la sección 2.1, se indican las características básicas del sonido.
- En el apartado 2.2 se señala la forma en que un instrumento de cuerda produce sonido.
- En el apartado 2.3 se describe cómo en música se definen ciertos sonidos como notas musicales.
- La sección 2.4 explica el sistema de encordado que utilizan los instrumentos de cuerda de interés para este trabajo.
- El apartado 2.5 hace una revisión de las distintas herramientas captadoras de sonido.
- En el apartado 2.6 explica en qué consiste la herramienta matemática de la Transformada de Fourier, para el análisis de espectro de frecuencias de una señal.
- La sección 2.7 muestra una alternativa a la Transformada de Fourier.
- El apartado 2.8 contextualiza las limitantes para reconstruir una señal.
- La sección 2.9 muestra cómo se modela el control de sistemas dinámicos.
- La sección 2.10 hace referencia al estado del arte en dispositivos afinadores de instrumentos musicales.

2.1 El sonido

Desde un punto de vista físico, el sonido audible producido por un instrumento musical es una serie de variaciones de presión en el aire, que a su vez es el medio por el que se propaga, transmitidas en forma de ondas. Estas ondas son de naturaleza similar a las que podría producir una piedra al perturbar la superficie de un charco de agua. El sonido producido por un instrumento musical tiene tres características que le identifican y/o distinguen [1].

2.1.1 Altura

También llamado tono. La altura de un sonido permite identificarlo como grave o agudo, más detalladamente, tiene que ver con qué nota musical percibe el oyente. Esta característica está relacionada con la cantidad de veces por segundo que se comprime y descomprime el aire, es decir, la frecuencia con que vibra el medio de propagación. Este es el elemento que se estudiará en el desarrollo del presente trabajo, pues para asociar un sonido con una nota musical específica se le debe asociar también una correspondiente frecuencia. En otras palabras, el proceso de afinar un instrumento musical corresponde a ajustarlo para que produzca un sonido con una frecuencia específica.

2.1.2 Intensidad

Esta cualidad es la que se percibe más fácilmente y dice cuán fuerte o débil es el sonido. Físicamente, la intensidad corresponde a la potencia emitida o recibida, o a la cantidad de energía que atraviesa una superficie por unidad de tiempo. La analogía más evidente es la de asociar la intensidad del sonido con el volumen que está percibiendo el oyente.

2.1.3 Timbre

Característica que distingue la fuente emisora del sonido. Dos sonidos con la misma intensidad y la misma altura, por ejemplo emitidos por un saxofón y un violín reproduciendo la misma nota musical al mismo volumen, pueden ser discriminados por su timbre. Cada instrumento musical tiene un timbre característico. En términos físicos, dos sonidos de iguales intensidad y frecuencia fundamental, se diferencian entre sí por los llamados armónicos que acompañan a la fundamental u onda principal. En otras palabras, es la combinación de frecuencias armónicas que son parte de un mismo sonido las que definen el timbre del mismo.

2.2 Instrumentos de cuerda

En esta sección se asume que el lector está familiarizado con el fenómeno físico de la cuerda vibrante [2]. Los instrumentos musicales de cuerda o cordófonos utilizan cuerdas vibrantes para producir sonido. Al hacer vibrar la cuerda, los extremos de esta permanecen fijos y se comportan como nodos de vibración, por lo que se puede estudiar como una onda. Lo que nos interesa es la nota musical que produce la cuerda al ser pulsada, es decir, la frecuencia de vibración de la misma. El largo de la cuerda, esto es, la distancia entre los nodos de vibración, resulta ser media longitud de onda $L = \frac{\lambda}{2}$, por lo tanto, considerando que la velocidad v , está dada por la longitud de onda λ , multiplicada por la frecuencia f .

$$v = \lambda \times f \quad (2-1)$$

Obtenemos entonces que la frecuencia que emite una cuerda de un instrumento al vibrar está dada por la fórmula:

$$f = \frac{v}{2L} \quad (2-2)$$

Donde v es la velocidad de propagación de la onda correspondiente a la frecuencia de vibración y L es el largo de la cuerda. La velocidad de propagación v de una onda en una cuerda es proporcional a la raíz cuadrada de la tensión de la cuerda τ e inversamente proporcional a la raíz cuadrada de la densidad lineal de masa δ de la cuerda [2].

$$v = \sqrt{\frac{\tau}{\delta}} \quad (2-3)$$

Reemplazando (2-3) en (2-2) se obtiene que la frecuencia fundamental del sonido producido por una cuerda vibrante de un instrumento musical está dada por:

$$f = \frac{1}{2L} \times \sqrt{\frac{\tau}{\delta}} \quad (2-4)$$

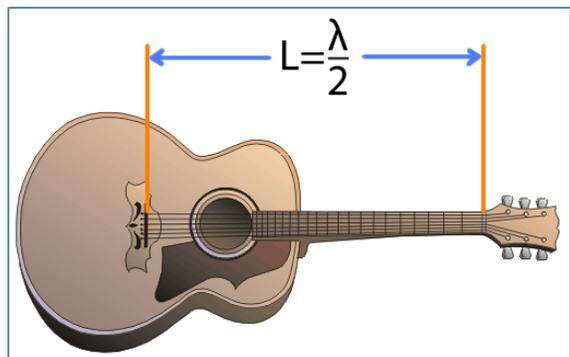


Figura 2-1: Cuerdas vibrantes en una guitarra.

Al hacer girar la clavija (ver apartado 2.4) de un instrumento de cuerda, lo que se está ajustando son las variables τ y δ . Cuando se enrolla la cuerda, se obtiene menos masa en la misma distancia, o sea, disminuye δ . Además, se aumenta la tensión τ . Se obtendrá entonces un sonido de mayor frecuencia (más agudo). El proceso de afinación de este tipo de instrumentos consiste en aplicar la tensión correcta a la cuerda para obtener la frecuencia deseada (nota musical).

¹ Imagen extraída de una búsqueda en Internet, desde la dirección web <http://www.upscale.utoronto.ca/GeneralInterest/Harrison/Vibrations/Images/PythagGuitar.gif>

2.3 Notas musicales

Ya se ha mencionado que una nota musical es un sonido con una frecuencia determinada. El oído humano no es capaz de percibir un rango infinito de frecuencias, sólo aquellas vibraciones que producen más de 16 ciclos de compresión y descompresión del aire por segundo -16[Hz]- o aquellas que se encuentren por debajo de los 20 000 ciclos por segundo -20 000[Hz]- son interpretadas como sonido por un ser humano. Todas las notas musicales existentes audibles se encuentran entonces, dentro de este rango de frecuencias [1].

En música existen 12 notas musicales fundamentales que se vuelven a repetir de forma cíclica al duplicar la frecuencia del sonido que emiten. Por ejemplo, la nota musical *Do* que se usa de referencia -*Do* central, también llamado *Do₄*- tiene una frecuencia de 261.63 [Hz], la nota que le sigue, *Re*, suena a 293.66 [Hz], luego *Mi* tiene 329.63 [Hz] y si se continúa aumentando la frecuencia del sonido se encuentran notas musicales más agudas, hasta topar con *Si*, que vibra a 493.88 [Hz]. La nota musical que viene a continuación de este *Si*, también es un *Do* -esta vez, *Do₅*- y tiene una frecuencia de 523.25 [Hz], que es justamente el doble de la frecuencia del *Do* central. Se dice entonces, que este nuevo *Do* pertenece a una octava superior. Dentro del rango de audición del ser humano, podemos encontrar varias octavas de notas musicales, cada una con las mismas 12 notas.

Existe una convención que asigna una frecuencia determinada a cada nota musical, desde el rango más grave hasta el más agudo dentro del espectro de audición [3] [4]. En esta convención, se numeran desde cero hasta 10 todas las octavas que poseen notas musicales con frecuencias dentro del espectro de audición humano. Cada frecuencia de las 12 notas pertenecientes a las 10 octavas, responde a la siguiente fórmula, en que *o* corresponde a la octava en que se encuentra la nota y *n* es la nota musical:

$$f(n, o) = 440 \times \sqrt[12]{2}^{(o-4) \times 12 + (n-10)} \quad (2-5)$$

La Tabla 2-1 muestra el resultado de aplicar esta fórmula a todas las notas musicales que se encuentran dentro del espectro de audición humano. Estas frecuencias corresponden a la convención del índice acústico científico, propuesto por Robert Young [3].

| n Nota | Nombre Nota | Octava | | | | | | | | | | |
|-----------|----------------|--------|------|-------|-------|--------------|-------|--------|--------|--------|---------|---------|
| | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 | Do | 16,4 | 32,7 | 65,4 | 130,8 | 261,6 | 523,3 | 1046,5 | 2093 | 4186 | 8372 | 16744 |
| 2 | Do# | 17,3 | 34,6 | 69,3 | 138,6 | 277,2 | 554,4 | 1108,7 | 2217,5 | 4434,9 | 8869,8 | 17739,6 |
| 3 | Re | 18,4 | 36,7 | 73,4 | 146,8 | 293,7 | 587,3 | 1174,7 | 2349,3 | 4698,6 | 9397,3 | 18794,5 |
| 4 | Re# | 19,4 | 38,9 | 77,8 | 155,6 | 311,1 | 622,3 | 1244,5 | 2489 | 4978 | 9956,1 | 19912,1 |
| 5 | Mi | 20,6 | 41,2 | 82,4 | 164,8 | 329,6 | 659,3 | 1318,5 | 2637 | 5274 | 10548,1 | |
| 6 | Fa | 21,8 | 43,7 | 87,3 | 174,6 | 349,2 | 698,5 | 1396,9 | 2793,8 | 5587,7 | 11175,3 | |
| 7 | Fa# | 23,1 | 46,2 | 92,5 | 185 | 370 | 740 | 1480 | 2960 | 5919,9 | 11839,8 | |
| 8 | Sol | 24,5 | 49 | 98 | 196 | 392 | 784 | 1568 | 3136 | 6271,9 | 12543,9 | |
| 9 | Sol# | 26 | 51,9 | 103,8 | 207,7 | 415,3 | 830,6 | 1661,2 | 3322,4 | 6644,9 | 13289,7 | |
| 10 | La | 27,5 | 55 | 110 | 220 | 440 | 880 | 1760 | 3520 | 7040 | 14080 | |
| 11 | La# | 29,1 | 58,3 | 116,5 | 233,1 | 466,2 | 932,3 | 1864,7 | 3729,3 | 7458,6 | 14917,2 | |
| 12 | Si | 30,9 | 61,7 | 123,5 | 246,9 | 493,9 | 987,8 | 1975,5 | 3951,1 | 7902,1 | 15804,2 | |

Tabla 2-1: Frecuencias de las notas musicales.

Se observa entonces que a medida que aumenta el espectro de frecuencias, la distancia entre una nota y otra es mayor. Sin embargo, el oído humano percibe estas diferencias de forma logarítmica. Por ejemplo, la diferencia de frecuencia entre las notas *La* y *La#* de la cuarta octava es de 26.2 [Hz], mientras que la diferencia entre las notas *La* y *La#* de la quinta octava es de 52.3 [Hz] (el doble). La diferencia entre *La* y *La#* de la sexta octava será de 104.7 [Hz] (cuatro veces la diferencia en la cuarta octava). Sin embargo, el oído humano percibe que la diferencia entre un *La* y un *La#* (y también la diferencia entre cualquier otro par de notas continuas) es la misma en cualquiera de las octavas.

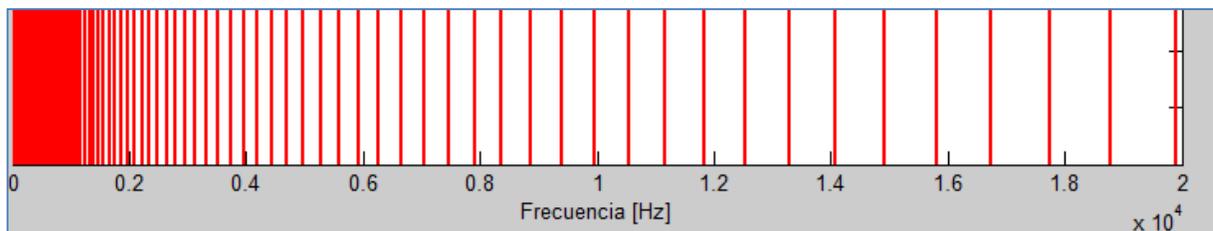


Figura 2-2: Distancia lineal entre las notas musicales.

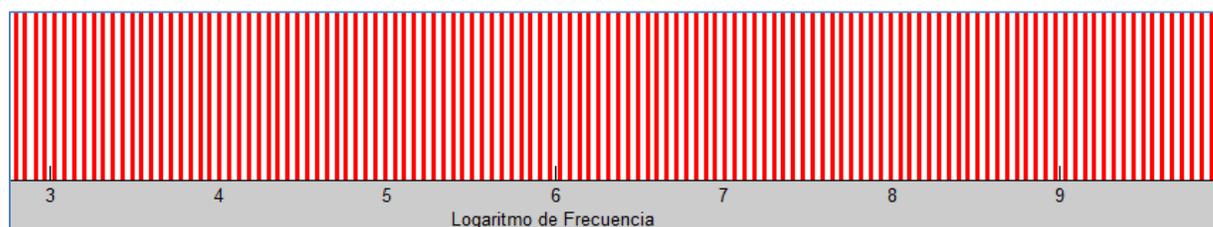


Figura 2-3: Distancia logarítmica entre las notas musicales.

La Figura 2-2 muestra un gráfico en una dimensión. Dentro del espectro de frecuencias audibles, se ha trazado una línea vertical en el valor de la frecuencia de cada una de las notas musicales obtenidas en la Tabla 2-1. Se evidencia entonces que las notas musicales de mayor altura, están mucho más distantes en frecuencia unas de otras.

La Figura 2-3 muestra la misma información, pero usando el logaritmo decimal de la frecuencia de cada nota musical, en esta escala se ve que cada nota es equidistante una de otra, esta es la sensación o percepción del oído humano.

Finalmente, a modo de ejemplo y combinando los valores de la Tabla 2-1 con la ecuación (2-4), la sexta cuerda de una guitarra afinada debería producir la nota Mi de la segunda octava -también llamado Mi_2 -, es decir, debería vibrar a $82.4 [Hz]$. Una típica sexta cuerda de guitarra tiene regularmente una densidad lineal de masa de $\delta = 8 \left[\frac{g}{m} \right]$. Además, el tiro (distancia entre los nodos de vibración de la cuerda) de una guitarra común y corriente es de unos $660 [mm]$. Utilizando la fórmula de (2-4) obtenemos que la tensión necesaria para que la cuerda esté afinada es de $94.64 [N]$ o puesto en valores industriales, $21.29 [lb]$.

2.3.1 El cent

Existe una unidad de medida para dimensionar diferencias en la tonalidad de dos sonidos de forma lineal. El cent es la unidad mínima de distancia tonal que se usa en música para medir intervalos y se expresa con el símbolo $[\text{¢}]$. Un semitono es la distancia lineal, usando la medida logarítmica, que existe entre dos notas musicales consecutivas. Por ejemplo, la diferencia de frecuencia que hay entre un Mi_1 y un Fa_1 es de $43.7 - 41.2 = 2.45 [Hz]$, pero en escala logarítmica es de $\log_2 \frac{43.7}{41.2} = 0.0833$; por otro lado, la diferencia en frecuencia entre un Mi_5 y un La_5 es de $698.5 - 659.3 = 39.2 [Hz]$, pero en la escala de igual temperamento la diferencia es de $\log_2 \frac{698.5}{659.3} = 0.833$, igual que en la primera octava. Entonces, se dice que en la escala de igual temperamento para todas las notas musicales, la distancia entre dos notas consecutivas (un semitono) es siempre la misma. La medida de un cent es igual a $\frac{1}{100}$ de semitono, es decir $1 [tono] = 200 [\text{¢}]$. La relación matemática para obtener la distancia en cents entre dos frecuencias distintas f_1 y f_2 está dada por [5]:

$$\text{¢} = 1200 \times \log_2 \frac{f_2}{f_1} \quad (2-6)$$

2.4 El sistema de encordado sinfín-corona

El sistema sinfín-corona es un montaje de engranajes que permite transmitir movimientos de giro en ejes distintos. Este sistema también permite amplificar fuerzas para hacer girar elementos que involucren cargas elevadas. Se compone de un tornillo sinfín que comunica el movimiento de giro a la rueda dentada que engrana con él, llamada corona [6].

Una vuelta completa del tornillo sinfín provoca el avance de un diente de la corona. Esto permite realizar una transformación de “velocidad en fuerza” es decir, varios giros del tornillo sinfín, provocarán un movimiento mucho más leve en la corona, pero sin involucrar grandes esfuerzos para realizarlo. Mientras más grande es la corona, más giros se necesitan del tornillo sinfín para hacerla moverse, pero también se necesitará menos fuerza para realizar el movimiento.



Figura 2-4: Clavija de guitarra.

En un instrumento musical, la clavija es una pieza que se hace girar para tensar o afinar la cuerda enrollada a ella. Este trabajo se enmarca en el contexto de los instrumentos musicales cuyas clavijas hacen uso del sistema sinfín-corona. En el caso de los instrumentos musicales que usan este tipo de engranajes en su clavijero, se une la corona al cilindro en el que se enrolla la cuerda. De esta forma al hacer girar el tornillo sinfín, el movimiento es transmitido a la corona que a su vez hace girar el cilindro, lo que provocará un cambio de tensión en la cuerda. La Figura 2-4 muestra una clavija de guitarra que usa el montaje del sistema sinfín-corona.

El sentido de giro que se necesita aplicar para ajustar la tensión depende de la técnica de encordado empleada, pero la convención internacional plantea al aplicar un torque positivo sobre la clavija (movimiento para “atornillar”) la cuerda disminuye su tensión, y al aplicar un torque negativo (movimiento para “desatornillar”) la cuerda aumenta su tensión.

¹ Imagen extraída desde una búsqueda en Internet, desde la dirección web <http://www.jaimevera.tecnoies.com/mecanismos/imagenes/sinfin8.jpg>

2.5 Interfaces captadoras de sonido

Anteriormente se mencionó que este trabajo propone un sistema que permite ajustar el sonido producido por una cuerda vibrante de un instrumento musical a una frecuencia específica. Para poder estudiar la frecuencia que tiene un sonido emitido, es necesario registrarlo de alguna forma. Existen distintas técnicas para poder transformar un sonido o vibración acústica en una señal eléctrica. Esto es, en efecto, el trabajo de una interfaz captadora de sonido: transformar las vibraciones del sonido que produce el instrumento musical, en una respuesta eléctrica medible y tratable. A continuación se mencionan algunos tipos de interfaces que capturan sonido.

2.5.1 Micrófono

Un micrófono es un sensor que convierte el sonido propagado por el aire, en una señal eléctrica. Los micrófonos son actualmente usados en distintos tipos de propósitos, esta es una lista no exhaustiva de distintos tipos de aplicaciones en que el micrófono es utilizado:

- El teléfono.
- Las grabadoras de sonido.
- Los sistemas de servicio médicos para déficit de audición.
- La producción cinematográfica y audiovisual.
- El megáfono.
- Los sistemas de reconocimiento de voz.
- El escáner ultrasónico.
- La transmisión de radio y de televisión.
- Afinadores de instrumentos musicales.

Según su sistema de funcionamiento, existen varias clasificaciones y tipos de micrófonos.

2.5.1.1 Micrófono dinámico

Los micrófonos dinámicos funcionan con el fenómeno de inducción electromagnética. Este es el tipo más común de micrófonos, debido a su bajo costo y características como la resistencia a la humedad y ganancia elevada, es decir, la señal generada no necesita ser pre-amplificada. Sin embargo, producen una señal de menor calidad que un micrófono capacitivo (ver sección 2.5.1.2), ya que capturan un nivel de ruido bastante alto. Esta relación calidad/precio, hace que los micrófonos dinámicos sean ideales para usar en eventos en vivo o amplificar sonido para un gran número de personas.



1

Figura 2-5: Micrófono dinámico.

El principio de funcionamiento de un micrófono dinámico es el mismo que el de una bocina o un parlante, pero revertido. Una bobina inductiva, ubicada en un campo magnético producido por un imán es conectada al diafragma del micrófono. El diafragma es la pieza que vibra debido a las oscilaciones de compresión y descompresión del aire cuando se transmite un sonido. Cuando el diafragma vibra, la bobina se mueve en el campo magnético, produciendo una variación de corriente, por inducción magnética.

Una sola membrana o diafragma no tiene un comportamiento lineal frente a distintos tipos de frecuencias del sonido. Por esta razón, se implementan varias membranas, para cubrir distintos intervalos de frecuencia del espectro audible. Obtener una combinación ordenada y óptima de todas las respuestas de cada membrana es complicado y costoso, es por eso que lo más común es que existan micrófonos dinámicos dedicados a capturar sonidos graves, medios o agudos.

¹ Imagen extraída del sitio web de la empresa SHURE, fabricante de micrófonos y soluciones audiovisuales.

2.5.1.2 *Micrófono de condensador*

El micrófono capacitivo fue inventado en los laboratorios BELL en 1916 por E.C. Wente [7]. También es llamado micrófono de condensador o micrófono electrostático. En estos micrófonos, el diafragma actúa como una de las placas de una capacitancia, y las vibraciones producen un cambio en la distancia entre las placas de la misma. Las placas tienen una carga fija. Obedeciendo la ecuación de la capacitancia $C = \frac{Q}{V}$, se tiene que dado que la carga Q es constante y la distancia entre las placas variable, se hace también variar la capacidad C , entonces el voltaje contenido en la capacitancia cambia con la vibración del aire.

Este tipo de micrófonos necesita ser alimentado (pre-amplificado) por una fuente de poder externa y la calidad de la señal generada es muy superior a la de un micrófono dinámico, es por eso que los micrófonos de condensador son usados para aplicaciones profesionales como por ejemplo, en estudios de grabación y post producción audiovisual. Su costo también es mucho más elevado que el de un micrófono dinámico.



Figura 2-6: Micrófono de condensador.

¹ Imagen extraída del sitio web de la empresa RODE, fabricante de micrófonos y soluciones audiovisuales.

2.5.1.3 Piezoeléctrico

Un micrófono de cristal o piezoeléctrico usa la característica de ciertos materiales para producir un voltaje cuando son sometidos a cambios de presión. Un cristal piezoeléctrico puede reaccionar a los cambios de presión producidos por un sonido propagándose en el aire y producir una diferencia de potencial. Este tipo de micrófonos tiene una impedancia muy alta, por lo que eran difíciles de hacer funcionar con equipos de audio a base de transistores, esto provocó que fueran rápidamente reemplazados por micrófonos dinámicos. La alta impedancia provoca además que este tipo de micrófonos sean muy susceptibles a generar ruido, al manipular el micrófono mismo o los cables que le conectan a los demás artefactos electrónicos.

Los piezoeléctricos son comúnmente utilizados como micrófonos de contacto para amplificar el sonido producido por instrumentos musicales acústicos, para percibir golpes de tambores y para grabar sonidos en ambientes complicados, como por ejemplo bajo el agua o a en lugares sometidos a altos niveles de presión.



Figura 2-7: Micrófono de piezoeléctrico.

¹ Imagen extraída de una búsqueda en Internet, desde la dirección web http://www.rtc magazine.com/files/images/4065/Meggitt-102816_medium.jpg

2.5.2 Cápsula magnética

La cápsula magnética es la interfaz más usada en instrumentos eléctricos con cuerdas metálicas. Una cápsula magnética se construye a partir de un imán que produce un campo magnético permanente, embobinado con varios miles de vueltas de cable de cobre. Una cápsula común y corriente de una guitarra eléctrica está generalmente embobinada con unas 5 000 vueltas del cable de cobre.

El funcionamiento es bastante simple: una porción de las cuerdas del instrumento está posicionada dentro del campo magnético producido por el imán y como las cuerdas son metálicas, estas tienen una respuesta magnética al campo permanente. Cuando las cuerdas vibran ocurre una perturbación en el campo magnético y por lo tanto, un cambio en el flujo magnético del imán. Luego, este cambio en el flujo magnético induce una diferencia de potencial en la bobina. Esta señal es transportada hacia algún sistema de amplificación o de grabación por medio de un cable.



Figura 2-8: Cápsula magnética.



Figura 2-9: Guitarra eléctrica con sus cápsulas.



Figura 2-10: Cápsula doble.

Una técnica moderna de construcción de cápsulas magnéticas utiliza dos bobinas orientadas de forma opuesta entre sí. Son llamadas cápsulas dobles. El objetivo es que cada bobina cancele el campo magnético residual en los bordes de la cápsula, y entonces minimizar el ruido transmitido en la señal producida.

¹ Imagen extraída del sitio web de OC DUFF PICKUPS, fabricante de cápsulas magnéticas.

² Imagen extraída del artículo disponible en Internet, en la dirección web <http://music.tutsplus.com/tutorials/replacing-electric-guitar-pickups--audio-10571>

³ Imagen extraída de una búsqueda en Internet, desde la dirección web <http://chasesraceplace.com/h15wn.jpg>

2.5.3 Transductor de cobre

Un transductor es un dispositivo dedicado a transformar una vibración acústica en una respuesta de señal eléctrica. En estricto rigor, todos los dispositivos mencionados hasta ahora son transductores. Sin embargo, el aparato descrito en el presente apartado no califica para ninguna de las categorías anteriores y por la misma razón, se identifica simplemente con el nombre genérico de “transductor”.

Se trata de una placa de cobre que va insertada a presión en alguna juntura entre las piezas de madera del instrumento. La placa de cobre forma un circuito cerrado que va conectado a un cable. Cuando el cable es energizado, la placa de cobre es sometida a una tensión eléctrica. Cuando el instrumento es ejecutado, la vibración de la madera del mismo es transmitida a la placa de cobre, lo que hace cambiar su resistencia interna, ya que varía el espesor de la misma. Luego, la tensión eléctrica de la placa varía junto con la vibración del instrumento, esta es la señal que entrega el transductor al sistema de amplificación o de grabación.



Figura 2-11: Transductor de cobre.

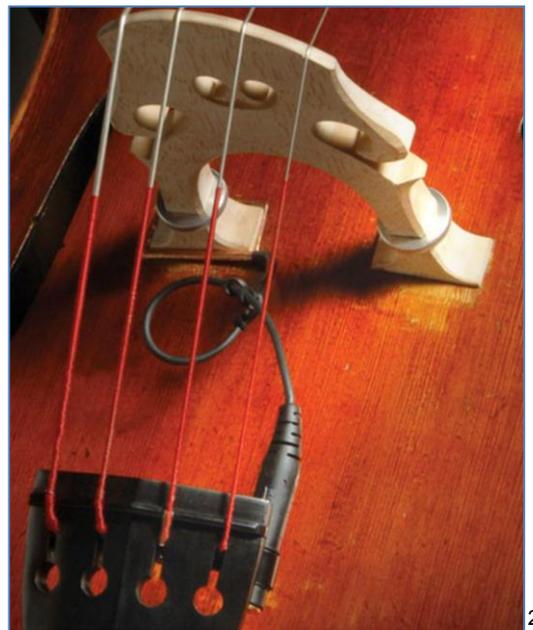


Figura 2-12: Transductor instalado en contrabajo.

La Figura 2-11 muestra la imagen de un transductor, compuesto por la placa de cobre y el cable que se conecta al sistema de amplificación. La Figura 2-12 muestra el mismo transductor, instalado entre el cuerpo y el puente de un contrabajo.

¹ Imagen extraída desde la dirección http://static.music123.com/derivates/19/001/249/470/DV020_Jpg_Jumbo_473983.933_bass.jpg

² Imagen extraída del sitio web de THE REALIST, fabricante de transductores de cobre.

2.6 Transformada de Fourier

Para detectar frecuencias de una determinada señal que ya ha sido capturada, existen técnicas que realizan cálculos según los valores de la señal en el dominio del tiempo. Sin embargo, en forma general, es mucho más eficiente, de bajo costo computacional, preciso y simple, usar el dominio de la frecuencia [8].

Una señal capturada posee un número determinado de muestras que corresponden a los valores de la señal en distintos instantes de tiempo. Para estudiar la señal en el dominio de la frecuencia se usa la Transformada de Fourier Discreta (DFT por sus siglas en inglés). La Transformada rápida de Fourier (FFT por sus siglas en inglés) corresponde al algoritmo computacional que permite calcular la DFT a partir de una señal temporal discreta.

A partir de una señal temporal discreta, se definen:

- f_s , la frecuencia de muestreo, esto es el número de muestras por unidad de tiempo que se usó para construir la señal discreta.
- T , el período de muestreo, esto es la duración de la señal temporal.
- N , el número total de muestras que contiene la señal discreta, o sea $N = T \times f_s$

La DFT se calcula a partir de la siguiente relación:

$$\hat{X}_k = \sum_{n=0}^{N-1} X_n e^{-\frac{i2\pi n k}{N}}, \quad 0 \leq k < N \quad (2-7)$$

En que X_n corresponde al n-elemento de la señal discreta y \hat{X}_k es el k-elemento de la DFT de la señal.

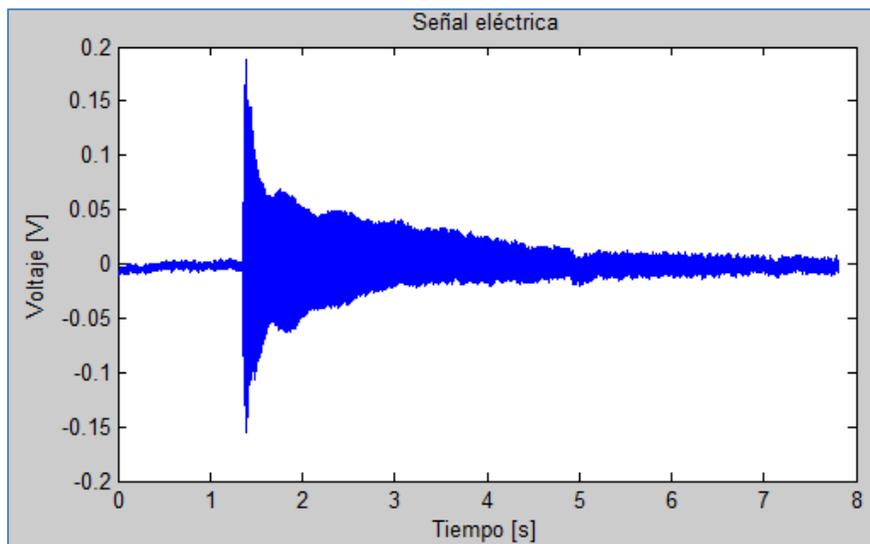


Figura 2-13: Captura de un Mi_3 producido por una guitarra.

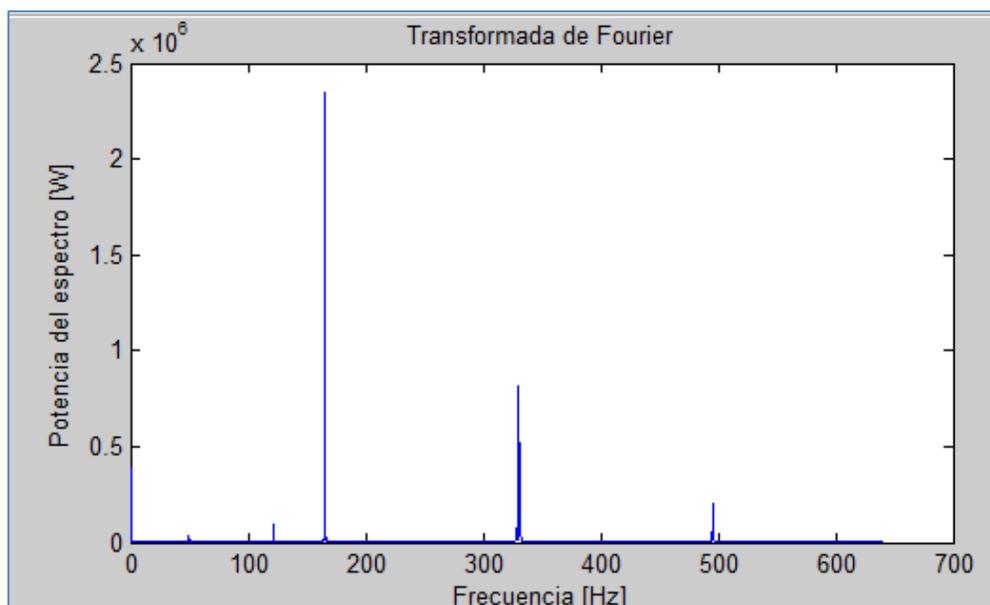


Figura 2-14: Transformada de Fourier de la señal de un Mi_3 producido por una guitarra.

La Figura 2-13 muestra la gráfica de la captura de audio de una guitarra haciendo sonar la nota musical Mi_3 (ver Tabla 2-1) y la Figura 2-14 muestra la gráfica del cuadrado del módulo de Transformada de Fourier de la misma señal. Para identificar la frecuencia fundamental de la nota musical capturada, se observa el pico más grande del espectro de frecuencias, en este caso se aprecia que la frecuencia más significativa es de 164.8 [Hz] lo cual es consistente, pues esta es la frecuencia que el índice acústico científico asigna a la nota musical Mi_3 (ver apartado 2.3). Además, se ven otros picos más pequeños en frecuencias armónicas, estas son las demás frecuencias que acompañan a la frecuencia fundamental y le dan el timbre característico al sonido capturado (ver sección 2.1.3). En el caso particular de esta señal, los armónicos que acompañan a la frecuencia fundamental son ondas con las frecuencias de las notas musicales Mi_4 y Si_4 .

Entonces, la Transformada de Fourier toma una serie de datos enmarcados en un período de tiempo determinado y devuelve el espectro de frecuencias de los datos. La FFT es una implementación computacional de la Transformada de Fourier y por lo tanto, debido a la naturaleza discreta de una serie de datos almacenados computacionalmente, no devuelve un espectro continuo. Luego, la señal transformada también es discreta y contiene la información del espectro de frecuencias en un número finito de intervalos (también llamados bins) equiespaciados en el dominio de la frecuencia. Estos bins de frecuencias tienen una resolución de frecuencia, expresada según el valor de la variable Δf , que se calcula con siguiente relación:

$$\Delta f = \frac{f_s}{N} = \frac{1}{T} \quad (2-8)$$

En la sección 2.3 se mencionó que las notas musicales no están equiespaciadas en el dominio de la frecuencia. Esto representa un problema en el uso de la Transformada de Fourier. Ya se explicó en párrafos anteriores que una forma de identificar la frecuencia fundamental de un sonido capturado, es observando el pico más significativo en el espectro de frecuencias de la Transformada de Fourier. Sin embargo, como las notas musicales de tonos más bajos están más cercanas unas de otras (ver Figura 2-2), podría ocurrir que en el momento de detectar frecuencias, más de una nota música de frecuencia baja caiga dentro del mismo bin. La Figura 2-15 evidencia tal problema en un gráfico de una dimensión del espectro de frecuencias entre 0 y 250 [Hz], esto es el ancho de banda que comprende las notas musicales de las octavas 0, 1, 2 y 3 (ver Tabla 2-1). En azul se han trazado líneas verticales marcando varios intervalos equiespaciados en el dominio de frecuencias. En verde aparecen círculos que muestran la ubicación de las notas musicales de las primeras 4 octavas en frecuencia. Se puede apreciar entonces, que en el sector de más bajas frecuencias (notas musicales más graves) hay intervalos que contienen más de una nota musical.

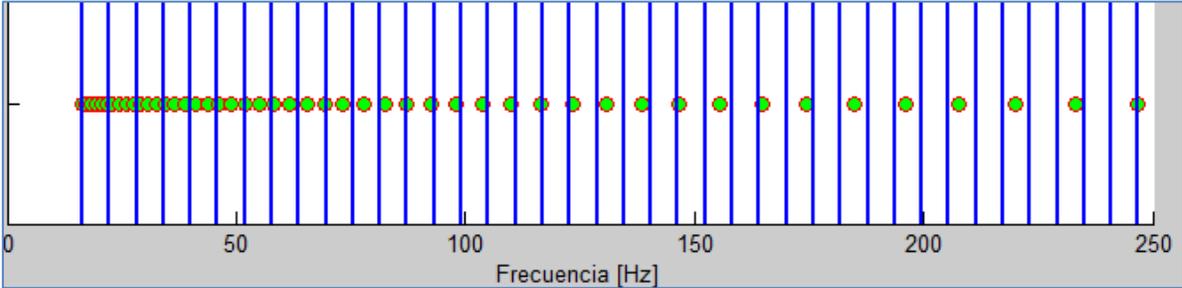


Figura 2-15: Intervalos equiespaciados en el dominio de la frecuencia.

La solución más evidente para este problema sería disminuir la variable Δf , o sea, hacer que la FFT utilice intervalos de mayor resolución de frecuencia, pero esto no siempre resulta conveniente, ya que para aumentar la resolución de frecuencia, el número de muestras adquiridas N al capturar el sonido debería ser mayor, lo cual aumenta el costo computacional de los cálculos que se deben realizar en el algoritmo de la FFT. El número de operaciones que realiza la FFT es del orden de $N \log N$, lo que representa una limitante más importante al aumentar N [8].

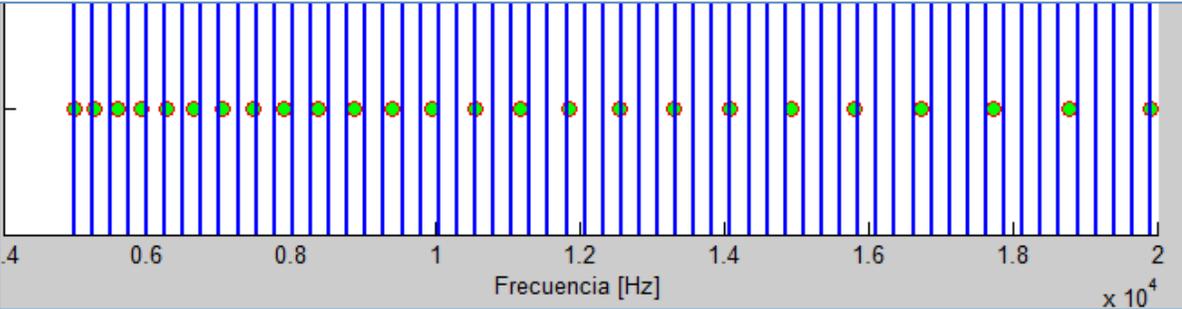


Figura 2-16: Intervalos equiespaciados en el dominio de frecuencias altas.

Si se aumenta la resolución de frecuencia para corregir el problema en el dominio de frecuencias bajas, aparece el problema contrario en el extremo opuesto del ancho de banda de frecuencias audibles. La Figura 2-16 muestra un gráfico de una dimensión del espectro de frecuencias entre 4 000 y 20 000 [Hz]. En azul se han trazado líneas verticales marcando varios intervalos equiespaciados de mayor resolución. En verde aparecen círculos que muestran la ubicación de las notas musicales de las últimas 2 octavas en frecuencia. Se puede apreciar entonces, que en el sector de más altas frecuencias (notas musicales más agudas) hay intervalos que no contienen (no discriminan) ninguna nota musical, lo que es en otras palabras, cálculos computacionales innecesarios.

2.6.1 Rellenar con ceros

Ya se mencionó que para aumentar la resolución de frecuencia, es necesario aumentar el número de muestras N . Asumiendo que no se puede variar la tasa de muestreo f_s , la única forma de aumentar N sería capturar una señal de mayor duración en el tiempo. Para el diseño de un dispositivo afinador, esta solución no es muy atractiva pues el usuario no va a desear ejecutar una nota musical por un período más largo de tiempo, además que un tiempo mayor de captura implicaría un mayor tiempo total necesario para la identificación de la nota musical, lo que podría entorpecer el uso del dispositivo. Para evitar tener que capturar la señal durante un período más grande de tiempo, se usa una técnica llamada (en inglés) Zero Padding [9].

Esta técnica consiste en agregar más muestras con el valor cero a la señal capturada, para entonces tener un N más grande y por lo tanto un Δf más pequeño.

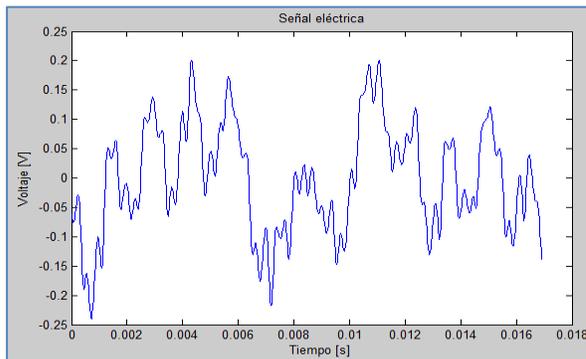


Figura 2-17: Captura de un acorde de Mi mayor producido por una guitarra.

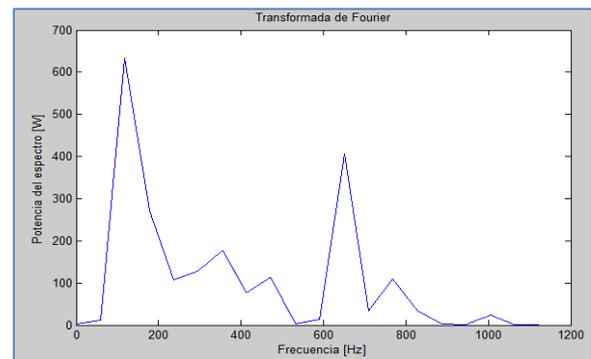


Figura 2-18: Transformada de Fourier de un acorde de Mi mayor.

La Figura 2-17 muestra un gráfico de la captura del sonido de un acorde de Mi mayor¹ producido por una guitarra, esta señal contiene $N = 746$ muestras. La Figura 2-18 muestra la transformada de Fourier de la misma señal.

¹ Un acorde de Mi mayor es un sonido compuesto por las notas musicales Mi Sol# y Si sonando simultáneamente.

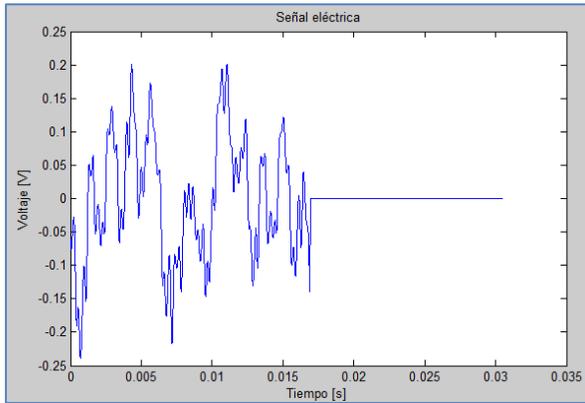


Figura 2-19: Señal capturada luego de aplicar Zero Padding.

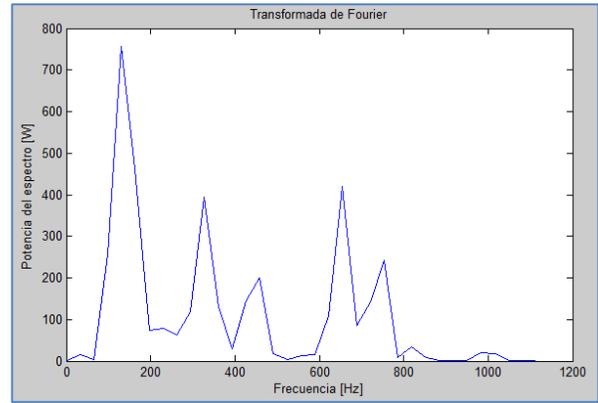


Figura 2-20: Transformada de Fourier luego de aplicar Zero Padding.

La Figura 2-19 muestra la misma señal, luego de agregar 600 muestras con el valor cero. La Figura 2-20 muestra la Transformada de Fourier de la señal luego de aplicar las 600 muestras con el valor cero. Se observa entonces, que la nueva Transformada es más suave y discrimina mejor los picos de las distintas frecuencias presentes en la señal capturada, sin necesidad de tener una captura de mayor duración en el tiempo.

Finalmente, a pesar de que se puede usar la técnica de Zero Padding para obtener una resolución mayor sin tener que hacer una captura más larga en el tiempo, nada de esto soluciona el problema de tener muchos intervalos equiespaciados (y por lo tanto, cálculos computacionales innecesarios) vacíos en el sector de frecuencias más altas.

Sería ideal entonces, para representar el espectro de frecuencias audibles y entonces poder discriminar las notas musicales existentes, poder usar bins o intervalos de mayor resolución en el espectro de frecuencias bajas y usar bins de mayor ancho de banda en el espectro de frecuencias altas.

2.7 Transformada de Hartley

Una de las desventajas de la Transformada de Fourier es que usa números complejos, que requieren vectores adicionales para ser representados en un sistema digital como una computadora o un micro-controlador. En el caso particular de tener una señal discreta de N términos representada totalmente con números reales, la Transformada de Fourier contiene información redundante, ya que los últimos $\frac{N}{2}$ términos resultan ser los conjugados complejos de los primeros $\frac{N}{2}$ términos.

Existe una herramienta matemática que permite obtener la misma información sobre el espectro de frecuencias de una señal que la Transformada de Fourier, pero que tiene la utilidad de sólo utilizar términos reales para su cálculo. La Transformada de Hartley se define por:

$$\hat{X}_k = \sum_{n=0}^{N-1} X_n \left[\cos\left(\frac{2\pi nk}{N}\right) + \sin\left(\frac{2\pi nk}{N}\right) \right], \quad 0 \leq k < N \quad (2-9)$$

De esta forma, la Transformada de Hartley pasa desde valores reales a valores reales, reduciendo el costo de memoria computacional para realizar el cálculo, a la mitad [10].

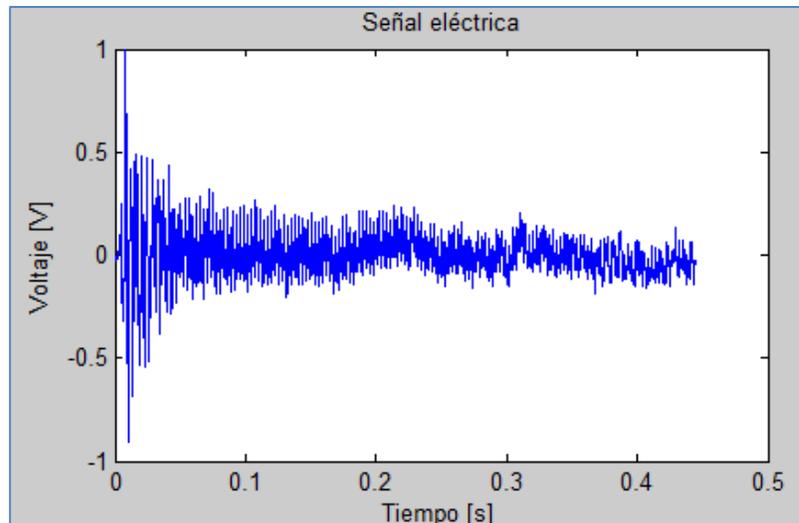


Figura 2-21: Ejemplo de señal discreta.

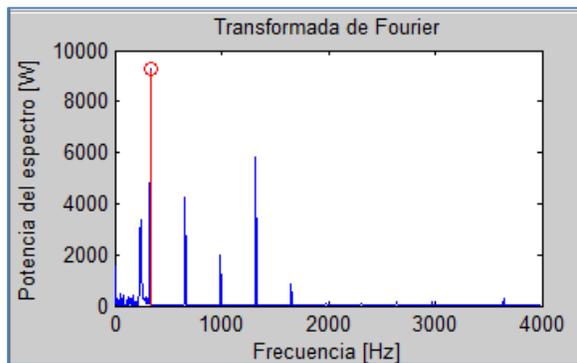


Figura 2-22: Densidad espectral usando Fourier.

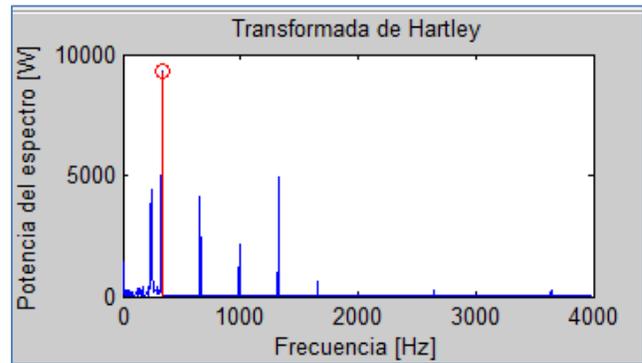


Figura 2-23: Densidad espectral usando Hartley.

La Figura 2-21 muestra un ejemplo de señal discreta correspondiente a la representación de una captura de sonido en el tiempo. La Figura 2-22 muestra la densidad espectral de la Transformada de Fourier. La Figura 2-23 muestra la densidad espectral de la Transformada de Hartley.

Se observa que con ambas herramientas matemáticas se obtiene exactamente la misma información. Esto es válido para cualquier señal discreta a números reales.

En el presente trabajo de título se estudiarán capturas de sonido representadas por medio de señales eléctricas de voltajes oscilantes, es decir, no es necesario trabajar con números complejos para crear sus representaciones digitales. En ese contexto, es más conveniente usar la Transformada de Hartley, que ahorra el costo computacional de tener que usar vectores de memoria adicionales para almacenar las partes imaginarias de una Transformada.

2.8 Teorema del muestreo

En el contexto de la teoría de la información, existe un teorema fundamental que establece una condición para que una señal analógica sea reproducida de forma correcta. Es el llamado Teorema de muestreo de Nyquist-Shannon y plantea que la reconstrucción exacta de una señal periódica continua a partir de muestras en instantes de tiempo, es matemáticamente posible si la señal está limitada en banda de frecuencia y la tasa de muestreo es superior al doble de su ancho de banda [11].

Llevado al contexto de este trabajo de título, este Teorema expone que si se quiere reconstruir una señal eléctrica analógica cuya frecuencia fundamental es F , entonces la tasa de muestreo en el proceso de digitalización de la señal f_s debe ser a lo menos dos veces la frecuencia fundamental de la señal estudiada, escrito en forma de inecuación:

$$f_s \geq 2F \quad (2-10)$$

Por ejemplo, si se desea identificar una nota musical La_4 (ver Tabla 2-1) en una señal capturada, entonces la tasa de muestreo en el proceso de digitalización debería ser mayor que $880[Hz]$.

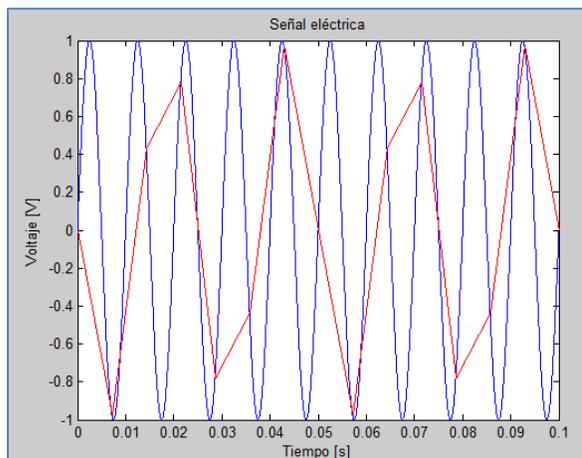


Figura 2-24: Señal reconstruida con $f_s = 150[Hz]$

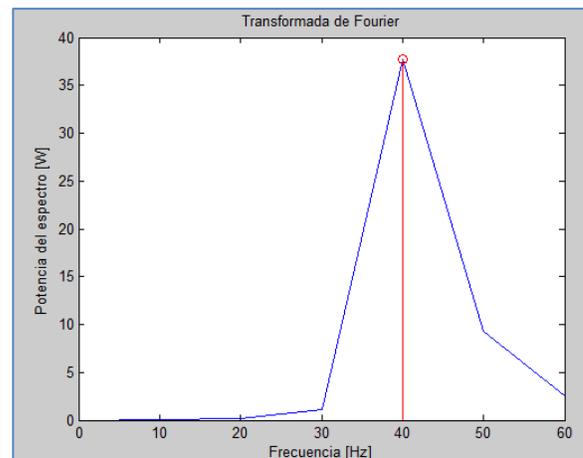


Figura 2-25: Densidad espectral $f_s = 150[Hz]$.

La Figura 2-24 muestra en azul una señal sinusoidal de amplitud $1[V]$ y de frecuencia $F = 100[Hz]$; y en rojo su reconstrucción digital usando una tasa de muestreo de $f_s = 150[Hz]$. Evidentemente la señal reconstruida no corresponde a la señal original, se ha perdido información y se ha obtenido una señal errada. La Figura 2-25 muestra que la frecuencia fundamental de la señal reconstruida es de $40[Hz]$, lo que reafirma que debido a la baja tasa de muestreo empleada en la reconstrucción, se adquieren datos errados, el valor de frecuencia fundamental que se desea obtener en este caso es de $100[Hz]$, tal cual es la característica de la senoide original.

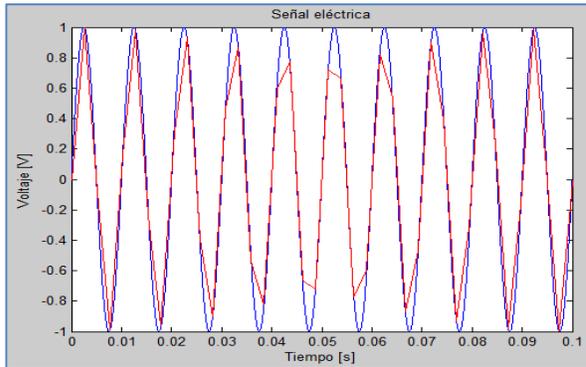


Figura 2-26: Señal reconstruida con $f_s = 400[Hz]$

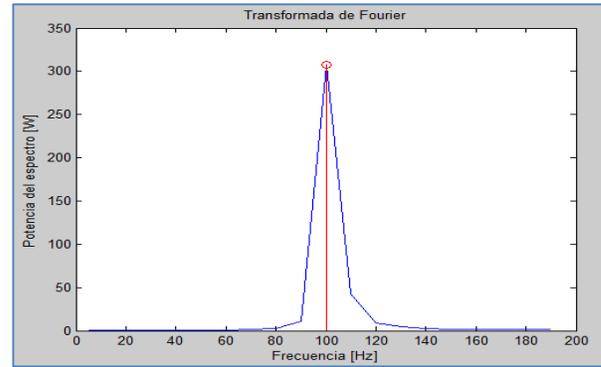


Figura 2-27: Densidad espectral $f_s = 400[Hz]$

La Figura 2-26 muestra en azul la misma señal anterior de amplitud $1[V]$ y de frecuencia $F = 100[Hz]$; y en rojo su reconstrucción digital usando una tasa de muestreo, esta vez de $f_s = 400[Hz]$. En este caso la señal reconstruida es mucho más cercana a la original, no se ha perdido información relevante. La Figura 2-27 muestra que la frecuencia fundamental de la señal reconstruida es de $100[Hz]$, tal cual es la característica de la sinusoide original.

Notar que a pesar de que se está cumpliendo la condición que plantea el Teorema del muestreo, la señal reconstruida en este caso no es exactamente igual a la original. El Teorema plantea que su reconstrucción exacta es matemáticamente posible, lo que no significa que tal reconstrucción sea directa al usar una tasa de muestreo superior a dos veces la frecuencia fundamental. En efecto, para lograr la reconstrucción exacta es necesario usar técnicas de interpolación que escapan los alcances del presente trabajo de título [12].

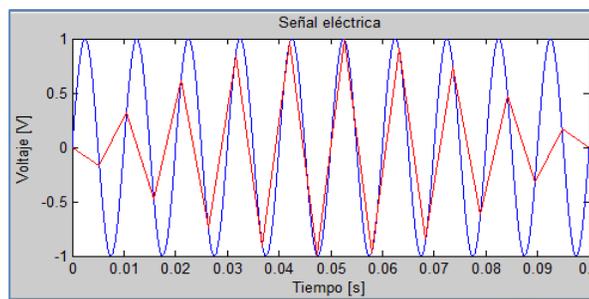


Figura 2-28: Señal reconstruida con $f_s = 200[Hz]$.

La Figura 2-28 muestra en rojo la reconstrucción de la señal sinusoidal usando una tasa de muestreo igual al límite impuesto por el Teorema del muestreo, esto es $f_s = 200[Hz]$. Se observa que los extremos de la onda están distorsionados, pero la frecuencia de la señal es conservada. En el contexto de la identificación de una nota musical, este estudio indica que es necesario usar como frecuencia de muestreo, a lo menos el doble de la frecuencia de la nota que se desea discriminar.

2.9 Control automático

A cualquier sistema físico cuyo estado cambia a través del tiempo se le denomina sistema dinámico. En el contexto de este trabajo, un instrumento musical es efectivamente, sin perder generalidad, un sistema dinámico y la variable del sistema que es de interés en este caso sería la frecuencia con que vibra una determinada cuerda del instrumento.

Existe una disciplina de las matemáticas y la ingeniería llamada Teoría de Control, que estudia el comportamiento de los sistemas dinámicos y plantea que cuando se desea que una variable determinada de un sistema siga un valor de referencia determinado, es posible construir un controlador que supervise algún elemento del sistema para lograr que la variable estudiada se acerque al valor deseado [13]. En el contexto del presente trabajo de titulación, el sistema controlado es el instrumento musical, el elemento que se controla es la tensión de la cuerda y la variable estudiada es la frecuencia de vibración de la cuerda.

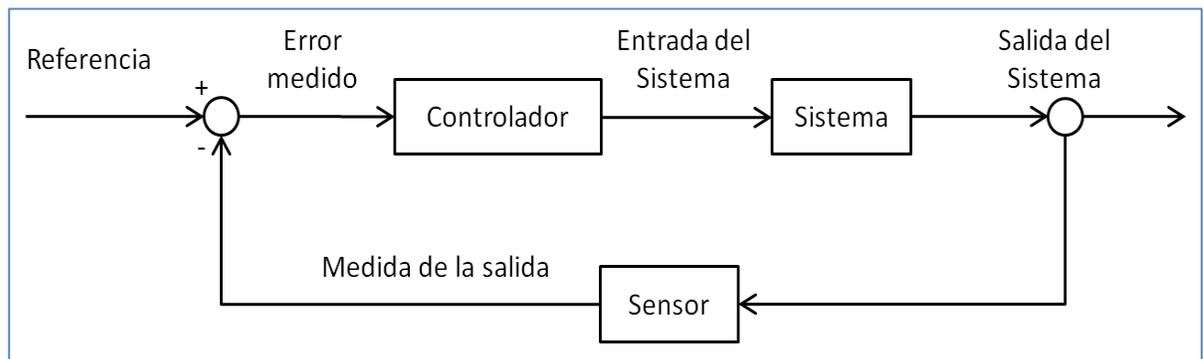


Figura 2-29: Lazo de control.

La Figura 2-29 muestra el principal postulado de la Teoría de Control, evidenciando gráficamente un esquema al que se le llama lazo de control [13]. Se tiene una referencia, que generalmente corresponde a un valor deseado para la variable de estudio. Cuando la variable de salida del sistema controlado se aleja de la referencia en el tiempo, un controlador manipula la entrada del sistema para hacer que la salida del sistema vuelva a acercarse a la referencia (retroalimentación). La retroalimentación puede ser negativa (regulación auto-compensatoria) o positiva (efecto "bola de nieve" o "círculo vicioso").

¹ Imagen extraída del sitio web de Wikipedia.

2.10 Estado del arte

Un afinador es un dispositivo de asistencia capaz de identificar la nota que produce el instrumento musical sobre el que se usa y entregar una referencia al usuario que le permite saber qué tan desafinado está el instrumento para posteriormente ajustarlo manualmente.

Los afinadores más económicos son también los más simples, generalmente usan luces LED para indicar si el sonido que está emitiendo el instrumento es más grave, más agudo o si está en la frecuencia deseada. Existen afinadores de mayor costo y de funcionamiento más complejo, que pueden indicar con mayor precisión la diferencia entre la altura emitida y la altura deseada, algunos dispositivos electrónicos de afinación son incluso más precisos que el oído humano al momento de identificar la frecuencia de un sonido.

Varios modelos de afinadores incorporan un circuito que hace el trabajo de detectar qué tono se está emitiendo y además lo compara con la altura deseada. Muchos músicos intérpretes en guitarra eléctrica o bajo eléctrico incorporan un panel de distintos efectos electrónicos para filtrar la señal de audio producida por el instrumento y producir nuevos sonidos. Es muy común que dentro de estos filtros se incorpore también un afinador electrónico, fijo dentro del panel de efectos del músico.



Figura 2-30: Ejemplo de panel de efectos que incorpora un afinador.

¹ Imagen extraída del sitio web de la empresa BOSS, fabricante de efectos y filtros para instrumentos eléctricos.

Algunos afinadores también incluyen un dispositivo que permite al usuario escuchar un sonido con la frecuencia deseada y poder afinar su instrumento “a oído” o para tener una referencia que permita encontrar las demás notas musicales deseadas. Este tipo de herramienta es muy utilizada con fines académicos, para entrenar el oído musical de los estudiantes.

Todas las soluciones existentes en el mercado hoy, están limitadas a sólo detectar la frecuencia del sonido emitido por el instrumento musical y compararla con la frecuencia de una nota musical deseada, es decir, es el usuario el que debe finalmente ajustar el instrumento, en otras palabras, es él quién afina el instrumento y no el dispositivo afinador.

Este trabajo de titulación pretende hacer una contribución en el campo de soluciones de asistencia para afinación de instrumentos y proponer un dispositivo que además de cubrir la funcionalidad de los afinadores actuales, también realice el trabajo mecánico de afinar, efectivamente, el instrumento. A continuación se mencionan los distintos tipos de afinadores existentes hoy en día.

2.10.1 Diapasón

El diapasón fue inventado en 1711 por el músico de origen británico John Shore [14]. Un diapasón es un resonador acústico, generalmente tiene forma de horquilla y está construido por un material metálico elástico, normalmente acero.

Al ser golpeado con algún otro objeto, un diapasón comienza a vibrar y entra en resonancia a la frecuencia de un tono específico y entonces emite una nota musical “pura”, es decir, produce un sonido con una sola frecuencia fundamental, libre de armónicos. El tono de la nota musical que produzca el diapasón depende del largo y la masa de los dos dientes. El mayor uso que se le da, es para generar el tono correspondiente a la nota musical La_4 , es decir, vibrar a $440[Hz]$.



Figura 2-31: Diapasón.

¹ Imagen extraída del artículo disponible en Internet en la dirección <http://blog.gfi.es/alineando-con-diapason/>

2.10.2 Afinadores convencionales

Un afinador utiliza un microprocesador (alternativa al circuito eléctrico) para medir la frecuencia media de una señal de entrada, generada por el sonido de un instrumento y la usa para mover la aguja que indica la frecuencia; para controlar la pantalla LCD que tenga incorporada; o para activar los LED según el sistema de señalización que tenga incorporado. Cuando el usuario hace que el instrumento reproduzca un sonido, el afinador lo detecta por medio de un micrófono o una línea de entrada (para un instrumento eléctrico). Luego el afinador muestra la diferencia en frecuencia que existe entre el sonido detectado y la nota musical deseada, indicando si el sonido es más grave, más agudo o de misma altura a la nota deseada.

2.10.2.1 Afinadores de aguja

Los afinadores de aguja indican una nota afinada (sonido con la correspondiente frecuencia) cuando la aguja se encuentra en posición vertical, cualquier desviación hacia la derecha o hacia la izquierda indicará que el sonido medido está fuera del tono correcto. Generalmente un afinador de aguja incorpora también una luz en el visor de la misma, para facilitar su uso en ambientes oscuros.

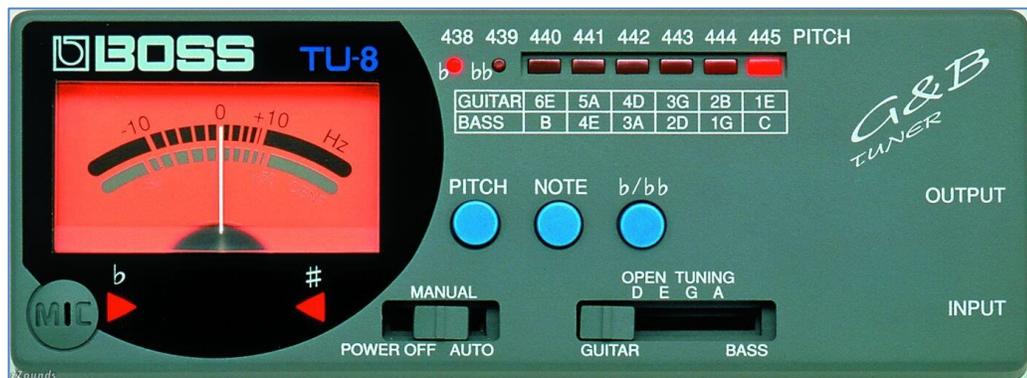


Figura 2-32: Afinador de aguja.

¹ Imagen extraída del sitio web de la empresa BOSS, fabricante de efectos y filtros para instrumentos eléctricos.

2.10.2.2 Afinadores LED

Los afinadores con LED usan el mismo concepto pero sólo usan luces LED para indicar si la señal de entrada posee la frecuencia deseada o si está fuera de tono. Si el sonido emitido por el instrumento corresponde a la nota musical deseada, los LED lo indicarán con una luz en el centro.



Figura 2-33: Afinador LED.

2.10.2.3 Afinadores LCD

Este tipo de afinador cuenta con una pantalla LCD, la mayoría imita el comportamiento de los afinadores de aguja mostrando una línea que se comporta de la misma forma. Existen también algunos que imitan el funcionamiento de un afinador de estroboscopio (ver 2.10.5), haciendo girar una referencia en la pantalla de forma cíclica, sin embargo, estos son sólo efectos visuales para facilitar la lectura al usuario y la forma en que detectan la frecuencia del sonido de entrada es siempre la misma, sin incorporar cambios en la precisión de la detección.

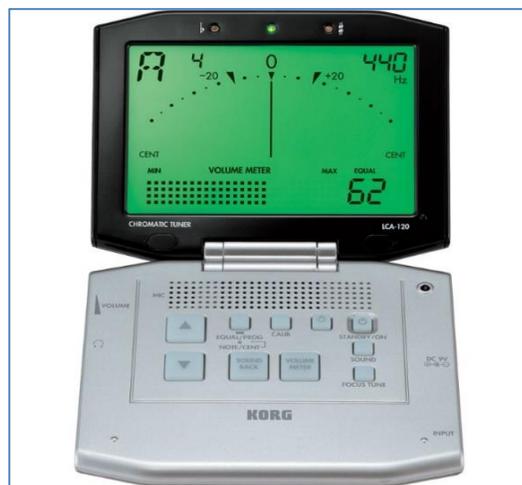


Figura 2-34: Afinador LCD.

¹ Imagen extraída de una búsqueda en Internet, desde la dirección http://en.academic.ru/pictures/enwiki/83/Sabine_STX-1100_Auto_Tuner.jpg

² Imagen extraída del sitio web de la empresa KORG, fabricante de afinadores electrónicos.

2.10.3 Afinadores dedicados

Para minimizar costo, se han diseñado afinadores con funciones limitadas, esto es, que sólo son capaces de detectar un específico número de notas musicales. Se usan para afinar un instrumento musical determinado. Por ejemplo, el caso más común es el de un afinador dedicado para afinar una guitarra, este tipo de afinador será capaz sólo de afinar 6 notas musicales, una por cada cuerda de la guitarra: Mi_2 ; La_2 ; Re_3 ; Sol_3 ; Si_3 ; Mi_4 ¹.

Este tipo de afinador es útil para orquestas pequeñas, bandas o grupos musicales que sólo usen un determinado tipo de instrumento o para músicos especialistas dedicados a la ejecución de un instrumento. Las orquestas convencionales, que incorporan una gran variedad de instrumentos que se afinan de formas distintas necesitarán un afinador capaz de detectar las 12 notas musicales existentes, también llamados afinadores cromáticos.

2.10.4 Afinadores de pinza

Esta categoría es independiente de las demás que se han mencionado hasta ahora, esto es, pueden existir afinadores de pinza que incorporen aguja, luces LED o pantallas LCD. La pinza es un sistema desarrollado en 1995 por Mark Wilson de la corporación *OnBoard Research*. Se trata de la interfaz que usa el afinador para captar la señal de audio. Este tipo de afinadores no usa un micrófono ni una línea eléctrica de audio, sino que se ajustan por presión a la madera del instrumento, de manera que todo el dispositivo vibra junto con el resto del instrumento cuando la cuerda del mismo es pulsada. Es decir, la onda mecánica viaja desde la cuerda, transmitida a la madera del instrumento y posteriormente transmitida al dispositivo mismo. Como no usan micrófonos, estos afinadores ignoran todo el ruido ambiental que puede haber en la habitación en que se encuentra el instrumento, permitiendo que sigan siendo efectivos en espacios abiertos o en salones de alto nivel de ruido.



Figura 2-35: Afinador electrónico con sistema de pinza.

¹ Se usa la notación con subíndices para indicar la escala a la que pertenece la nota musical referenciada, ver sección 2.3.

² Imagen extraída del sitio web de la empresa KORG, fabricante de afinadores electrónicos.

2.10.5 Afinador estroboscópico

Uno de los dispositivos más precisos (y de más alto costo) es el afinador estroboscópico o de estroboscopio y es utilizado por técnicos profesionales en sonido, músicos y lutieres¹.

Se trata de un artefacto mecánico que cuenta con una serie de LED alimentados por una señal de audio amplificada, esta señal de audio es la representación del sonido que está emitiendo el instrumento que se desea afinar. Los LED parpadean entonces, a la misma frecuencia que tiene la señal de audio (frecuencia del sonido que emite el instrumento). Por ejemplo, una nota La_2 emitida por la quinta cuerda de una guitarra tiene una frecuencia de $110[Hz]$ si se encuentra afinada. Un La_4 producido por el primer par de cuerdas de un Laúd español² suena a $440 [Hz]$. Con cualquiera de estos dos sonidos, los LED del afinador de estroboscopio van a parpadear 110 o 440 veces por segundo correspondientemente. Frente a estas luces intermitentes se sitúa un disco translúcido con sectores transparentes y opacos alternados.

El disco se hace rotar a una velocidad específica, asignada por el usuario. Existen 12 velocidades posibles, cada una corresponde a una frecuencia particular de una nota musical. Si el sonido que hace parpadear los LED se encuentra a la misma frecuencia que el giro del disco, ocurre un fenómeno visual que hace que el disco parezca estar detenido (efecto estroboscópico). Si el sonido que alimenta los LED está desafinado (no corresponde a la exacta frecuencia de la nota musical deseada), el patrón del disco parece estar en movimiento, ya que el parpadeo de los LED con la rotación del disco no se encontrarían sincronizados. Mientras más lejos se encuentre la frecuencia del sonido emitido (más desafinado el instrumento), más rápido parece moverse el patrón del disco, a pesar de que este esté siempre girando a la misma velocidad para una nota dada.



Figura 2-36: Afinador estroboscópico.

¹ Un lutier es un artesano que construye o repara instrumentos de cuerda.

² El Laúd español es un instrumento de cuerda pulsada de 12 cuerdas afinadas de a pares.

³ Imagen extraída del sitio web de la empresa PETERSON, fabricante de afinadores estroboscópicos.

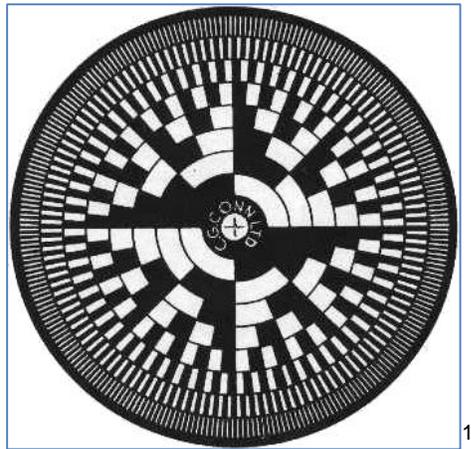


Figura 2-37: Disco de un afinador de estroboscopia.

La Figura 2-37 muestra que el disco del afinador de estroboscopia tiene varios patrones, distribuidos en forma de anillos cada uno con diferentes distancias entre las zonas opacas y transparentes. Cada uno de los anillos de patrones en el disco puede ser interpretado como una escala distinta para una sola nota siendo afinada. Esto permite que este tipo de afinador tenga una muy alta precisión, imposible de alcanzar en afinadores tradicionales de LED o de aguja. El sistema de estroboscopia de afinación es alrededor de 30 veces más sensible que los demás dispositivos de afinación [15]. Teóricamente, este tipo de afinador es capaz de identificar variaciones de $\frac{1}{10} [\text{c}]^2$. Para instrumentos de cuerda como la guitarra, se requiere una precisión de unos 20 [c] para afinar [5].

Existen afinadores de estroboscopia que cuentan con 12 discos independientes, uno para cada nota, capaces de afinar varias notas al mismo tiempo o el sonido de un acorde³. Sin embargo, este tipo de herramientas es muy pesada y además frágil. Cada uno de los 12 discos requiere calibración y mantenimiento dependiendo qué tanto se use. El uso general que se le da a estos dispositivos también abarca fines académicos, en que los estudiantes pueden observar las subestructuras de una nota musical, en cada uno de los discos.

¹ Imagen extraída del sitio web de la empresa PETERSON, fabricante de afinadores estroboscópicos.

² El cent es la menor unidad usual que se emplea para medir intervalos musicales, la distancia entre dos notas consecutivas, esto es un semitono, es de 100 cent.

³ Un acorde consiste en un conjunto de varias notas musicales distanciadas por un específico patrón armónico entre ellas.

3 DISEÑO DEL DISPOSITIVO

El funcionamiento del dispositivo que propone este trabajo de título va de la mano junto con su diseño y ambos se pueden expresar en bloques o módulos con distintas funciones:



Figura 3-1: Diagrama de bloques del funcionamiento del dispositivo.

La Figura 3-1 muestra un diagrama de bloques representando cada uno de los módulos que forman parte del afinador electromecánico. Cada uno de estos módulos será diseñado de forma independiente según la tarea que cumplen en el proceso completo de afinación.

3.1 Selección de la interfaz capturadora

El primer bloque de la Figura 3-1 señala que la primera etapa en el proceso de funcionamiento del afinador es adquirir la señal acústica que emite el instrumento que se desea afinar. Dentro del diseño del artefacto, la primera etapa será también elegir una interfaz adecuada para convertir la señal acústica en una respuesta eléctrica que se pueda estudiar, tratar y a partir de la cual el dispositivo pueda tomar decisiones. Se establecen entonces, tres criterios importantes para elegir una buena interfaz capturadora:

- **Bajo nivel de ruido:** Es necesario poder contar con una señal que no traiga información no deseada, es decir, la interfaz debe ser capaz de transformar en respuesta eléctrica sólo el sonido emitido por el instrumento y no el de otras fuentes, que para efectos del funcionamiento del afinador, son consideradas ruido.
- **Bajo costo:** Existen muchos tipos distintos de interfaces. Aquellas que realizan una réplica más fiel (de mayor calidad) de la señal transformada son considerablemente más caras. Para efectos de la afinación de un instrumento sólo se necesita poder recuperar correctamente la frecuencia de la señal capturada y no tener una captura de calidad de estudio profesional. Según este criterio, no es necesario contar con un micrófono electrostático (ver apartado 2.5.1) de elevado costo que se usa para aplicaciones profesionales de grabación y producción audiovisual.
- **Tamaño reducido:** Este criterio es válido para todos los elementos del afinador y particularmente para la interfaz: Es necesario que sea poco invasiva y no supere las dimensiones naturales de las piezas del instrumento que se desea afinar. Por ejemplo, el elemento del instrumento musical que permite ajustar la afinación es

la clavija (ver sección 2.4). Una típica clavija de guitarra mide unos 3 [cm] de ancho (ver anexo 7.1). Según este criterio, no se podría elegir, por ejemplo, un micrófono dinámico de 20 [cm] de alto.

Para cumplir con el primer criterio es necesario hacer pruebas para medir la respuesta que tiene un micrófono o transductor según su orientación espacial y distancia con respecto a la fuente de ruido. En un contexto de uso del afinador, además de tener el sonido propio del instrumento que se desea afinar, se pueden tener otras fuentes de sonido a distintas orientaciones y distancias que también pueden ser capturadas por la interfaz.

En la sección 2.5 se realizó una revisión de distintos tipos de interfaces captadoras de sonido, todas cumplen la misma función y se diferencian en el fenómeno físico que utiliza para realizar la conversión. Dentro del marco académico del curso EL3003 - Laboratorio de Ingeniería Eléctrica, el grupo de trabajo número 2 de la sección 2 realizó una experiencia de laboratorio electiva para estudiar transductores eléctricos [16]. En particular, se estudió la respuesta que entregaban micrófonos dinámicos y electrostáticos, según la disposición espacial que tenían con respecto a la fuente que produce el sonido. El informe del mencionado trabajo se encuentra disponible en la herramienta pedagógica de u-cursos.

La Figura 3-2 muestra la respuesta en voltaje que se observó luego de orientar un micrófono dinámico SHURE PG-48 (ver anexo 7.2) en distintas posiciones. La posición en 0° es tal que el lóbulo del patrón de captura del micrófono (cardioides) se orienta directamente de frente a la fuente de sonido. La posición en 90° es tal que el eje principal del lóbulo del patrón de captura del micrófono forma 90° con el eje principal de la dirección en que es emitido el sonido desde la fuente. La posición en 180° es tal que el lóbulo del patrón de captura del micrófono se orienta en sentido contrario a la fuente de sonido. Todas estas pruebas fueron realizadas frente a un parlante amplificador conectado a un generador de funciones, desde donde se emitía un tono de 1000 [Hz].

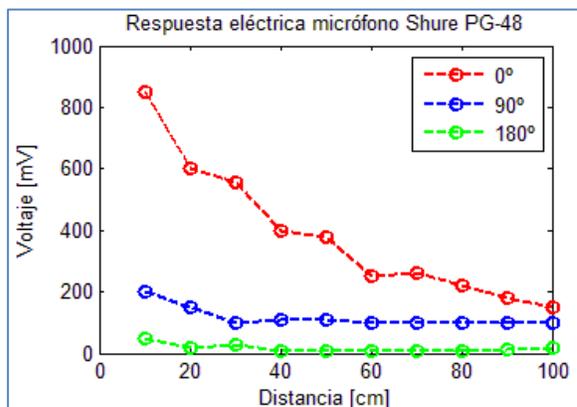


Figura 3-2: Respuesta eléctrica de un micrófono dinámico.

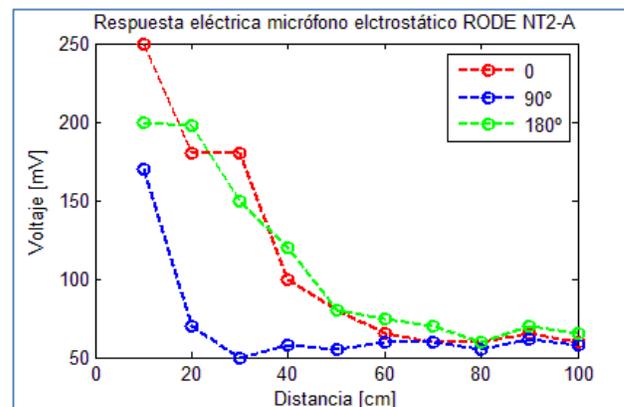


Figura 3-3: Respuesta eléctrica de un micrófono de condensador bidireccional.

La Figura 3-3 muestra la respuesta en voltaje que se observó luego de orientar un micrófono electrostático RODE NT2-A (ver anexo 7.3) en distintas posiciones y distancias usando su patrón bidireccional. La posición en 0° es tal que el lóbulo del patrón de captura del micrófono (cardioides) se orienta directamente de frente a la fuente de sonido. La posición en 90° es tal que el eje principal del lóbulo del patrón de captura del micrófono forma 90° con el eje principal de la dirección en que es emitido el sonido desde la fuente. La posición en 180° es tal que el lóbulo del patrón de captura del micrófono se orienta en sentido contrario a la fuente de sonido. Todas estas pruebas fueron realizadas frente a un parlante amplificador conectado a un generador de funciones, desde donde se emitía un tono de 1000 [Hz].

3.1.1 Micrófono de contacto

La Figura 3-2 y Figura 3-3 muestran que a pesar de que una fuente de sonido se encuentre distanciada del micrófono, este de todas formas genera una respuesta eléctrica. Esto se debe a que este tipo de transductores funciona con un diafragma o membrana (ver sección 2.5.1) que vibra junto con las compresiones y descompresiones del aire producidas por la propagación del sonido. Esta característica representa una desventaja pues el micrófono es susceptible a producir respuesta eléctrica incluso frente a los sonidos ambientales o ruidos propagados por el aire que no corresponden al sonido producido por el instrumento musical que se desea afinar.

En la sección 2.5.1.3 se mencionó el funcionamiento de un micrófono piezoeléctrico. Es posible crear un micrófono usando material piezoeléctrico carente de diafragma que vibre con las compresiones y descompresiones del aire. Un micrófono de contacto tiene esta característica: no genera ninguna respuesta frente al sonido que viaja en el aire. La respuesta eléctrica del micrófono de contacto se obtiene al juntar la superficie del mismo al material (generalmente madera) del instrumento musical que produce el sonido. De esta forma, el material piezoeléctrico vibra junto con el sólido al que se encuentra adherido y no junto con el aire.

Este tipo de micrófono es ideal para la aplicación que se está proponiendo en el presente trabajo de título, porque a pesar de que no tenga una gran calidad de audio en la señal obtenida, la información importante (frecuencia) sí es recuperada sin error. La onda mecánica que produce la cuerda es propagada por la madera del instrumento musical hacia el material piezoeléctrico del micrófono de contacto. De esta manera, cualquier otro sonido acústico o ruido presente en la sala en que se esté utilizando el instrumento que se desea afinar, simplemente no es detectado (debido a la ausencia de diafragma) por el micrófono de contacto.

Para el diseño del dispositivo afinador, se utiliza un micrófono de contacto Korg CM-200 (ver anexo 7.4). Este micrófono captura directamente las vibraciones del instrumento musical y genera la señal eléctrica analógica que será usada en el proceso de afinación. La Figura 3-4 muestra el transductor elegido para capturar la onda mecánica

correspondiente a la nota musical que produce el instrumento y transformarla en señal eléctrica.



Figura 3-4: Micrófono de contacto KORG CM-200.

Este micrófono cumple con todos los criterios expuestos en la sección 3.1:

- No es sensible al ruido. Debido a que no tiene diafragma, no produce respuesta eléctrica frente a los sonidos que se propagan por el aire.
- Es económico y fácil de obtener. Se encuentra disponible en la mayoría de las tiendas de música por un valor que no supera los \$6.000.- pesos.
- Su tamaño es limitado a la pinza que afirma el dispositivo al instrumento y su dimensión más importante no supera los 30 [mm].

3.2 Amplificación de la señal capturada

El primer problema que se presenta al usar directamente un micrófono pasivo, es que la señal eléctrica que este entrega es demasiado débil o de muy baja amplitud. Para poder ser tratada, la señal analógica producida por el micrófono será digitalizada.

Los micro-controladores y sistemas de procesamiento digital de datos generalmente tiene un rango de tolerancia que varía desde los 0 [V] hasta los 5 [V]. Esto significa que a un valor analógico entre 0 y 5 se le asigna un número entero entre 0 y 1023 (para el caso de un byte de buffer) [17] distribuido linealmente. De esta forma, por ejemplo, si un micro-controlador detecta un voltaje de 0 [V], se asignará el número 0. Si se detecta un voltaje de 5 [V], se asignará el número 1023. Si se detecta un voltaje de 3 [V], entonces se asignará el número 613. A cualquier valor que esté por debajo de los 0 [V] se le asigna el número 0 y a cualquier valor que esté por sobre los 5 [V] se le asigna el número 1023.

El problema de tener una señal analógica muy débil o de baja amplitud es que se pierde resolución en la información obtenida. Por ejemplo, una señal típica obtenida de

un micrófono tiene una amplitud de unos 200 [mV], es decir, el máximo número que asignaría el proceso de digitalización sería el entero 40, quedando 984 números disponibles para representar la señal sin uso.

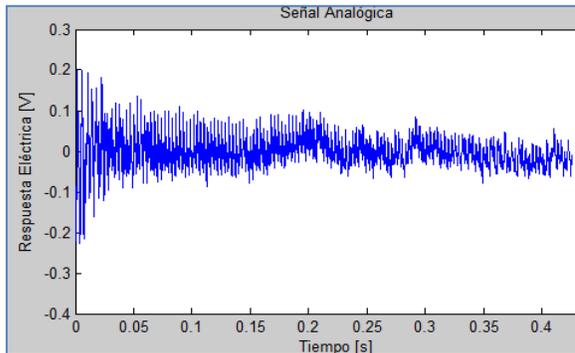


Figura 3-5: Respuesta eléctrica del micrófono del contacto.

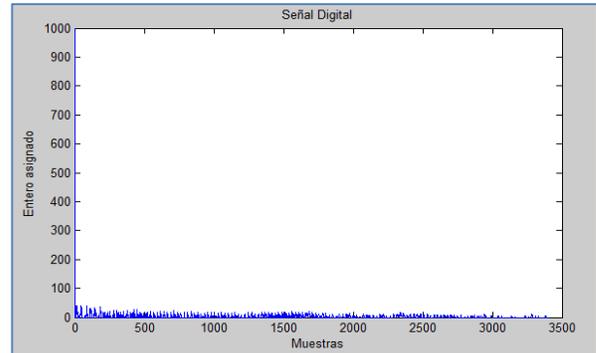


Figura 3-6: Digitalización de la respuesta del micrófono sin previo tratamiento.

La Figura 3-5 muestra la gráfica de la respuesta eléctrica del micrófono de contacto elegido al detectar las vibraciones de una guitarra acústica común y corriente (ver Figura 3-7). Se observa que la amplitud de la señal es de unos 200 [mV] (400 [mV] V_{pp}). La Figura 3-6 muestra una simulación del proceso de digitalización sin previo tratamiento de la señal analógica. Se observa entonces el problema descrito en el párrafo anterior: Toda la información de la señal es concentrada entre los enteros 0 y 40, quedando todo el resto de la capacidad de procesamiento del micro-controlador desaprovechada, esto implica que se está perdiendo información de la señal original. Además, la Figura 3-6 evidencia un segundo problema: La señal original oscila en torno al valor 0 [V] y naturalmente tiene valores negativos, todos los valores negativos de la señal están siendo ignorados pues el proceso de digitalización les asigna el valor cero. En otras palabras, la Figura 3-6 está mostrando una representación de “media onda” de la señal original. Este problema es estudiado en la sección 3.3.

Para poder usar todos los números enteros disponibles por el micro-controlador, es necesario amplificar la señal hasta un máximo de 5 [V]. Entonces, antes de comenzar el proceso de digitalización de la señal, el dispositivo afinador debe contar con un amplificador para dejar la respuesta analógica del micrófono en el rango correcto que utilizará el micro-controlador.

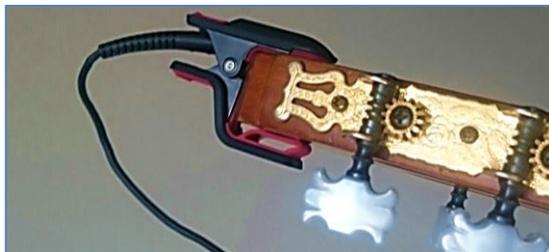


Figura 3-7: Micrófono de contacto montado en clavijero de guitarra.

3.2.1 Construcción del amplificador de señal

Se propone entonces un amplificador no inversor, haciendo uso de una resistencia, un potenciómetro, un OPAMP y dos baterías de 12 [V]:

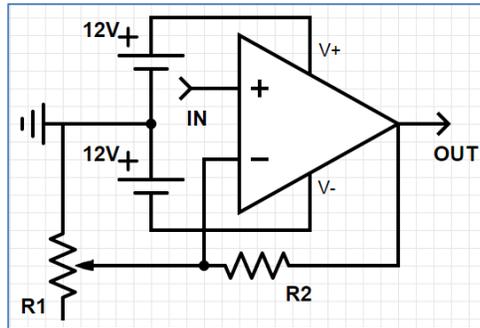


Figura 3-8: Esquema del amplificador no inversor.

Para calcular los efectos del amplificador en la señal de entrada, usamos las leyes de voltajes y corrientes.

Sea V_{in}^+ el voltaje en la entrada no inversora del OPAMP; V_{in}^- el voltaje en la entrada inversora del OPAMP; V_{out} el voltaje en la salida del OPAMP; $V_{gr} = 0 [V]$ el voltaje de referencia; R_2 la impedancia de la resistencia; y R_1 la impedancia del potenciómetro, se tiene que:

$$V_{in}^+ = V_{in}^- \quad (3-1)$$

$$\frac{V_{in}^- - V_{out}}{R_2} = \frac{V_{gr} - V_{in}^-}{R_1} \quad (3-2)$$

A partir de (3-2) obtenemos que:

$$V_{out} = V_{in}^- \times \left(1 + \frac{R_2}{R_1}\right) \quad (3-3)$$

Reemplazando (3-1) en (3-3) se obtiene que:

$$V_{out} = V_{in}^+ \times \left(1 + \frac{R_2}{R_1}\right) \quad (3-4)$$

De esta forma, si se usa una resistencia de $R_2 = 100[k\Omega]$ y un potenciómetro que varía el valor de R_1 entre 0 y $10[k\Omega]$ tendremos que cuando el potenciómetro está en su máximo valor el factor de amplificación será de $\left(1 + \frac{100}{10}\right) = 11$. En este escenario, la amplitud máxima de la señal analógica sería amplificada desde 200 [mV] hasta 2.2 [V]

Si el potenciómetro está girado hasta la mitad, entonces el factor de amplificación será de $\left(1 + \frac{100}{5}\right) = 21$. Este valor podría parecer exagerado, pero para situaciones de

ganancia más baja, cuando la señal que entrega el transductor tiene una amplitud del orden de los 100 [mV] entonces la señal amplificada sería del orden de 2.1 [V].

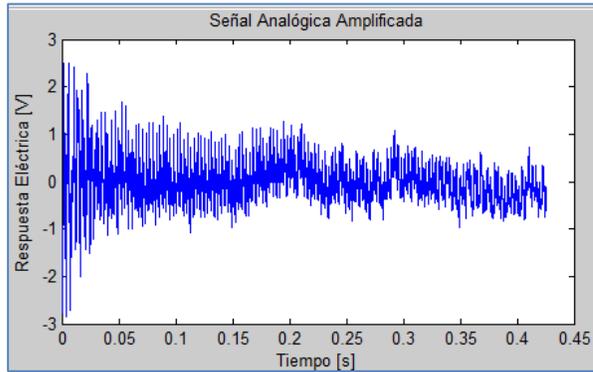


Figura 3-9: Respuesta eléctrica del micrófono luego de ser amplificada.

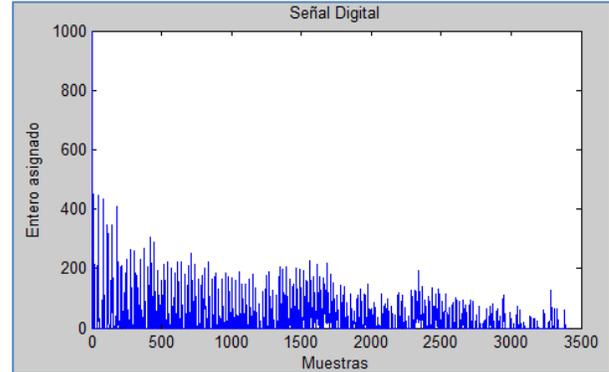


Figura 3-10: Digitalización de la respuesta del micrófono luego de ser amplificada.

La Figura 3-9 muestra una simulación de la respuesta del amplificador luego de recibir la misma señal eléctrica expuesta en la Figura 3-5. En este escenario se está usando un factor de amplificación de 11. La Figura 3-10 muestra una simulación del proceso de digitalización de la señal amplificada. Se observa que esta vez se tiene mayor resolución y sí se está recuperando información de la señal original. El proceso de amplificación de la señal analógica permite que una mayor cantidad de números enteros disponibles sean utilizados en la representación digital de la señal.

Hasta ahora sólo se ha propuesto un amplificador no inversor que lleva la amplitud de la señal analógica desde 200 [mV] hasta 2.2 [V]. En el apartado anterior se señaló que la tolerancia del micro-controlador es hasta de 5 [V]. Esto explica por qué la Figura 3-10 presenta también una cantidad de números enteros (aproximadamente la mitad de los 1024 enteros disponibles) que no están siendo asignados. En este caso se sigue viendo una representación de media onda, es decir, todos los valores negativos de la señal analógica de la Figura 3-9 están siendo ignorados y se les asigna arbitrariamente el valor cero. Este problema será abordado en el apartado 3.3.

Para la construcción del amplificador no inversor, se utilizó un OPAMP del fabricante STMicroelectronics modelo TL082 (ver anexo 7.6).

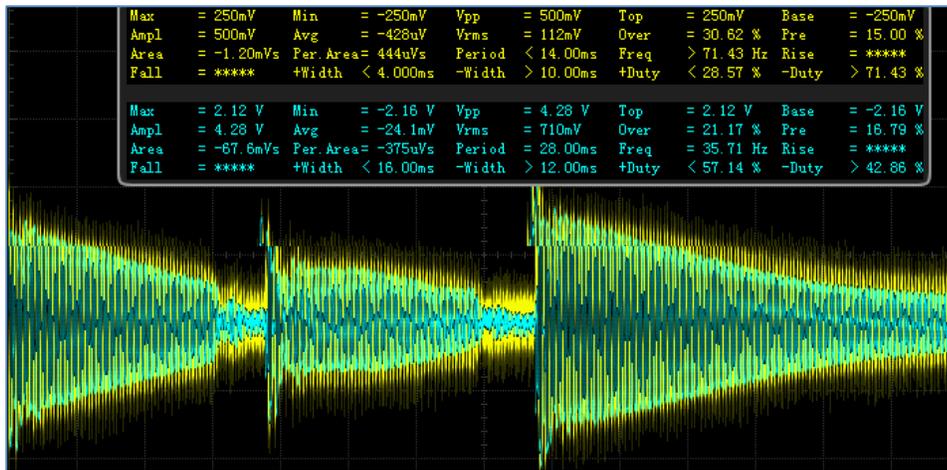


Figura 3-11: Señales sin amplificar (amarillo) y sin amplificar (cyan) medidas con osciloscopio.

La Figura 3-11 muestra una captura de pantalla del osciloscopio RIGOL MSO-4012 (ver anexo 7.5) obtenida en el laboratorio de electrónica del Departamento de Ingeniería Eléctrica de la Facultad de Ciencias Físicas y Matemáticas de la Universidad de Chile. Esta medición muestra la respuesta eléctrica del micrófono KORG CM-200 montado al clavijero de una guitarra acústica común y corriente sin amplificar (amarillo) y luego de tratar la señal con el amplificador no inversor aquí propuesto (cyan).

En la Figura 3-11 se nota claramente cada pulsación en las cuerdas de la guitarra durante un período de tiempo. Se observa en amarillo la señal obtenida directamente desde el micrófono KORG CM-200 (ver anexo 7.4) que tiene una amplitud de 250 [mV] alcanzando entonces un voltaje V_{pp} de 500 [mV]. En color cyan se observa la señal de salida del amplificador no inversor descrito al comienzo de este apartado, superpuesta con la original. La señal amplificada alcanza una amplitud de hasta 2.12 [V] y un valor V_{pp} de 4.28 [V]. En términos de amplitud, el voltaje V_{pp} es lo que interesa adaptar a la tolerancia de 5 [V] del micro-controlador. En este caso se está alcanzando un valor adecuado pues queda un margen de 0.72 [V] en caso de alguna fluctuación que deba ser medida.

3.3 Agregar componente continua

Ya se mencionó en la sección 3.2 que el uso de un amplificador corrige el problema del desperdicio de capacidad de procesamiento de un micro-controlador, pero todavía existe el problema de que la señal está siendo representada “a la mitad”. La señal analógica oscila en torno al valor 0 [V] incluso después de ser amplificada (ver Figura 3-11) y todos los valores de voltajes negativos que sean detectados en el proceso de digitalización son ignorados (ver Figura 3-10).

Para solucionar esto, es necesario agregar una componente continua de 2.5 [V] a la señal que ya ha sido amplificada, de forma que el valor central no sea 0 [V] sino 2.5 [V].

En otras palabras, el tratamiento en esta etapa será sumar una constante de 2.5 [V] a la señal analógica amplificada, esto provocará que la señal amplificada, en lugar de moverse entre los valores $-2.5[V]$ y $2.5[V]$, oscile entre los valores $0[V]$ y $5[V]$ que corresponde, en efecto, al rango de tolerancia que es capaz de detectar un micro-controlador común.

Si la señal amplificada es desplazada “hacia arriba” en 2.5 [V] entonces todos los valores negativos que antes eran ignorados en el proceso de digitalización, esta vez podrán ser detectados (ya que serán valores entre 0 y 5 [V]) y se les podrá asignar un número entero entre 0 y 1023.

3.3.1 Construcción del circuito offset

Se propone entonces un circuito *offset*, haciendo uso de dos resistencias, dos condensadores y una fuente de voltaje de 5 [V]:

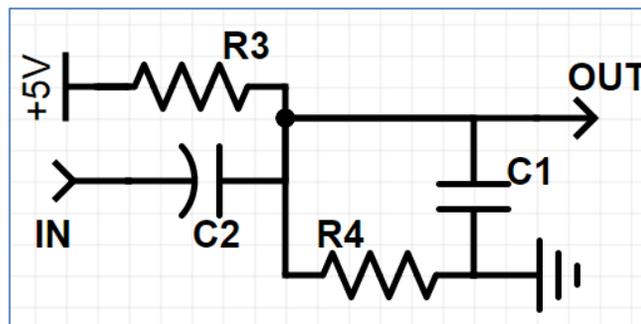


Figura 3-12: Esquema del circuito *offset*.

El efecto de este bloque de circuito en la señal analógica es simplemente agregar una componente continua de 2.5 [V], en otras palabras, la señal analógica es trasladada 2.5 [V] “hacia arriba”. En la entrada de este bloque se hace llegar la señal amplificada (ver sección 3.2) y en la salida se recupera una señal con la misma información pero, en lugar de oscilar entre -2.5 y 2.5 [V], el rango de voltajes alcanzados varía entre 0 y 5 [V].

El *offset* tiene dos elementos resistivos de igual resistencia eléctrica R_3 y R_4 conectadas entre la fuente de 5 [V] y la tierra (ver Figura 3-12). Esto corresponde a un divisor de voltaje y por lo tanto, la juntura entre R_3 y R_4 es igual a 2.5 [V].

La juntura de 2.5 [V] es conectada a la salida del amplificador no inversor por medio de un condensador C_2 . Mientras la señal analógica entregada por el amplificador incrementa y disminuye su voltaje (señal oscilante), ocurre una carga y descarga repetida de C_2 . Este efecto provoca que la juntura de 2.5 [V] incremente y disminuya (oscile en torno a 2.5 [V]) de la misma forma que la salida del amplificador.

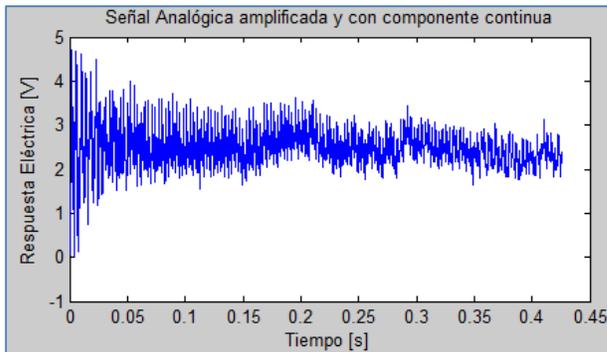


Figura 3-13: Respuesta eléctrica del micrófono luego de ser amplificada y trasladada.

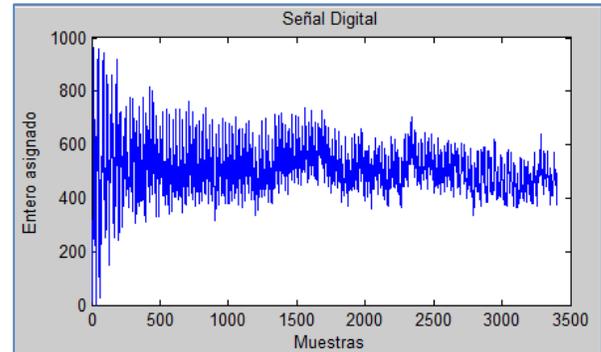


Figura 3-14: Digitalización de la respuesta del micrófono luego de ser amplificada y trasladada.

La Figura 3-13 muestra una simulación de la respuesta del circuito *offset* luego de recibir la señal amplificada expuesta en la Figura 3-9. Se observa que la forma de la respuesta eléctrica en el tiempo es exactamente la misma, siendo el rango de oscilación de la señal el único parámetro que ha variado.

La Figura 3-14 muestra una simulación del proceso de digitalización de la señal amplificada y trasladada. Se observa que esta vez se ha conservado la resolución obtenida en el apartado 3.2.1 y además se recupera la parte inferior de la onda. Como ya no hay valores negativos en la señal analógica, estos no son ignorados. El circuito *offset* permite que todo el rango de los 1024 números enteros disponibles sea utilizado en la representación digital de la señal.

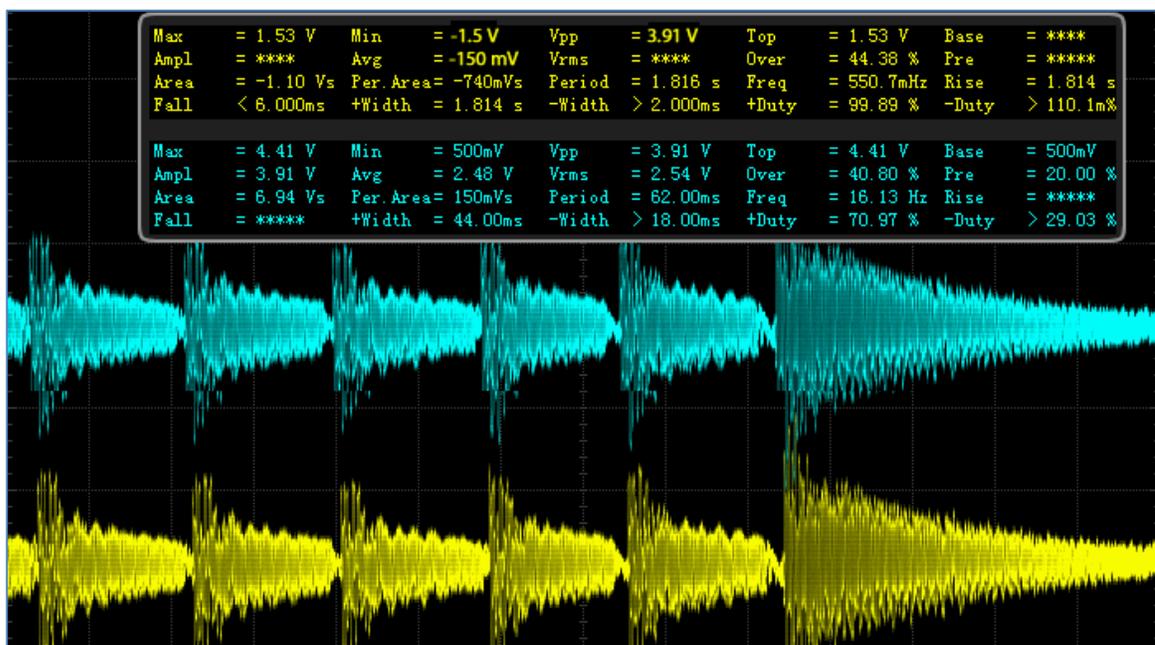


Figura 3-15: Señales de salida del amplificador (amarillo) y de salida del offset (cyan) medidas con osciloscopio.

La Figura 3-15 muestra una captura de pantalla del osciloscopio RIGOL MSO-4012 (ver anexo 7.5) obtenida en el laboratorio de electrónica del Departamento de Ingeniería Eléctrica de la Facultad de Ciencias Físicas y Matemáticas de la Universidad de Chile. Esta medición muestra en color amarillo, la respuesta del amplificador no inversor (ver sección 3.2) y en color cian, la misma señal luego de tratarla con el circuito *offset* aquí propuesto.

En la Figura 3-15 se puede notar cada pulsación en las cuerdas de la guitarra durante un período de tiempo. Se observa en amarillo la señal obtenida directamente desde el la salida del amplificador no inversor que tiene un voltaje V_{pp} de 3.91 [V]. En color cian se observa la señal de salida de bloque *offset* descrito al comienzo de este apartado. La señal desplazada es idéntica a la original y se diferencia sólo en que en lugar de oscilar en torno al valor 0 [V], lo hace a partir de 2.5 [V]. En términos de amplitud, el voltaje V_{pp} es idéntico. Esta señal está en condiciones de ser detectada y replicada por el micro-controlador, ya que en el proceso de digitalización no se perderá información y además los valores de voltajes estarán siempre entre 0 y 5 [V].

3.4 Interfaz de digitalización

El micro-controlador que se utiliza en el diseño del dispositivo afinador, corresponde al ATmega328, integrado en la placa REDBOARD del fabricante Sparkfun (ver anexo 7.7).



Figura 3-16: Placa de desarrollo Sparkfun REDBOARD.

¹ Imagen extraída del sitio web de SPARKFUN, productor de hardware libre para desarrollo.

El REDBOARD es una plataforma de hardware libre, basada en una placa con un micro-controlador y un entorno de desarrollo, diseñada para facilitar el uso de la electrónica en proyectos multidisciplinarios.

La Placa REDBOARD es una interfaz que deja a disposición las entradas y salidas digitales del micro-controlador ATmega328 (ver anexo 7.8) y que además permite programar el micro-controlador por medio del entorno de desarrollo Arduino, basado en lenguaje de programación C. Entonces, por medio de REDBOARD, es posible cargar un algoritmo en el ATmega 328 y hacer que funcione con las distintas señales de entrada y de salida.

Por medio del entorno de programación, es posible diseñar un algoritmo que sea ejecutado por el micro-controlador. Este entorno se llama *Arduino IDE* y se puede ejecutar desde un terminal de trabajo con Windows [18]. Desde el computador es posible escribir el código que será cargado mediante un cable USB, por medio del REDBOARD, al ATmega328.

3.4.1 Frecuencia de muestreo

Una de las limitantes del uso de un micro-controlador como el ATmega328 es la disponibilidad de la memoria de acceso aleatorio (RAM). En este caso particular sólo se dispone de $2[kB]$ de memoria para almacenar variables durante la ejecución de un algoritmo.

Cada número real de la serie discreta que representa una señal analógica usa un espacio de $2[B]$ de memoria RAM. Entonces, para hacer un buen uso de la memoria de acceso aleatorio disponible hay que limitar el vector que contiene cada muestra de la señal discreta a una dimensión razonable. Además, para usar el algoritmo de la Transformada Rápida de Hartley (FHT) es necesario que la cantidad de muestras N sea una potencia de 2 ($N = 2^x$) [10].

Una forma inteligente de trabajar con estas dos limitantes es elegir un número de muestras $N = 256$. De esta forma la memoria utilizada por el vector que almacena la señal digital en el micro-controlador no supera los $512[B]$ de RAM, quedando $\frac{3}{4}$ del total de la memoria disponible para almacenar otras variables, tales como contadores, acumuladores, señalizadores, etc.

En la sección 2.6 se estudió que las notas musicales de más baja frecuencia en el espectro audible requieren de mayor resolución en los bins de la Transformada para ser discriminadas de forma correcta. Además, la ecuación (2-8) indica que para aumentar la resolución (disminuir Δf) usando una cantidad de muestras N fija, es necesario disminuir la frecuencia de muestreo f_s con la que se obtienen las muestras de la señal digital a partir de la señal analógica.

La nota musical de frecuencia más baja reproducible por un instrumento de cuerda con sistema sinfín-corona corresponde a un Mi_1 , oscila a $41.2[Hz]$ (ver Tabla 2-1) y es el tono que produce la cuerda más gruesa de un bajo eléctrico o un contrabajo. La nota musical que le sigue hacia abajo en frecuencia es el $Re\#_1$ y oscila a $38.9[Hz]$, existiendo una diferencia de $2.31[Hz]$ entre estas dos notas. Para tener un margen de discriminación aceptable, se toma la mitad de la diferencia en frecuencia entre estas dos notas, de manera de tener a lo menos un bin libre entre cada nota musical. Para que la Transformada de Hartley pueda discriminar de forma correcta estas dos notas del espectro de bajas frecuencias, entonces es necesario que la resolución de los bins sea de al menos $\Delta f = 1.15[Hz]$. Nuevamente usando la ecuación (2-8), se obtiene que la frecuencia de muestreo para discriminar este rango de notas musicales es del orden de $f_s = 300[Hz]$.

Sin embargo, según el teorema del muestreo (ver sección 2.8) con una frecuencia de muestreo de $300[Hz]$ no se podrían detectar notas musicales más agudas que un Re_3 ¹. Con este orden de frecuencia de muestreo no hay problemas para afinar instrumentos de registro bajo, tales como el contrabajo, pero sería imposible lograr afinar, por ejemplo una guitarra, cuya cuerda más aguda vibra a $329.6[Hz]$ o una bandurria, cuya cuerda más aguda vibra a $880[Hz]$.

Para poder detectar correctamente las notas musicales del rango de frecuencias más agudas es necesario aumentar la frecuencia de muestreo f_s en el proceso de digitalización de la señal, lo que llevaría a una disminución de la resolución en los bins de la Transformada, sin embargo este es un costo que sí se puede abordar porque en el espectro de frecuencias más altas, las notas musicales son más distantes en frecuencia (ver Figura 2-15) y por lo tanto requieren menor resolución entre cada bin para ser detectadas.

Afortunadamente, la frecuencia de muestreo f_s que usa el ATmega328 es una variable que se puede ajustar por medio del entorno de programación, programando tareas secuenciales en sincronía con los relojes internos del micro-controlador.

3.4.2 Entorno de programación

En el entorno de programación que permite escribir el código del algoritmo que será ejecutado por el micro-controlador, siempre se definen dos funciones principales: *setup* y *loop*. La primera se usa para declarar variables de programación y definir qué puertos de entrada y salida digitales del REDBOARD serán utilizados para ejecutar las tareas. La segunda contiene el algoritmo en sí, es decir las distintas instrucciones que serán ejecutadas por el micro-controlador. Todas las instrucciones dentro de la función *loop* son ejecutadas indefinidamente mientras el micro-controlador se encuentre energizado.

¹ Un Re_3 oscila a $146.83[Hz]$. Esta es la nota musical que se produce al pulsar una cuarta cuerda de guitarra.

```
Circuit_01 $
void setup()
{
  pinMode(13, OUTPUT);
}

void loop()
{
  digitalWrite(13, HIGH);
  delay(1000);
  digitalWrite(13, LOW);
  delay(1000);
}
```

Figura 3-17: Ejemplo simple de código en el entorno de programación.

La Figura 3-17 muestra un ejemplo de código escrito en el entorno de programación. Dentro de la función *setup* existe sólo una instrucción, que define el pin 13 del REDBOARD como un puerto de salida. Dentro de la función *loop* primero se ordena que el pin 13 adquiera un voltaje alto (ya que es un sistema digital, las salidas sólo adquieren valores binarios HIGH para someterlo a 5 [V] y LOW para 0[V]), luego se da la orden se retener la ejecución del algoritmo durante 1000[ms], después se da la orden de que el pin 13 adquiera un voltaje bajo y finalmente se da la orden de que la ejecución se detenga nuevamente durante 1000[ms]. Estas 4 instrucciones serán ejecutadas de forma cíclica e indefinida mientras el micro-controlador se encuentre energizado.

En términos de hardware, si se conecta un LED al pin 13 del REDBOARD (ver Figura 3-18) mientras el micro-controlador ejecuta el código de la Figura 3-17, se observará que la luz se enciende y apaga de forma intermitente con intervalos de 1[s].

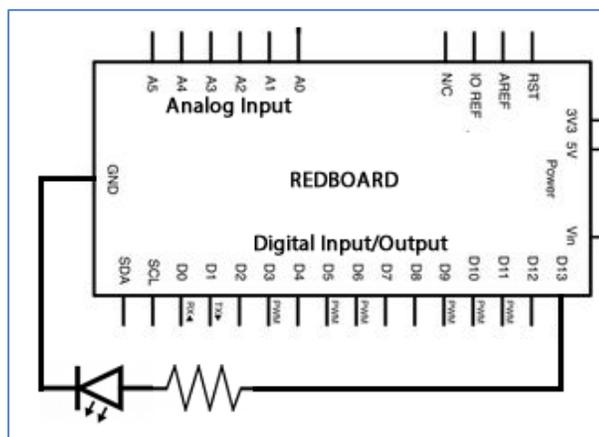


Figura 3-18: Circuito que ejecuta las acciones del código programado.

3.4.3 Relojes internos del ATmega328

El ATmega328 tiene un conjunto de características llamadas *Timer interrupts*, que se usan para ejecutar tareas en intervalos de tiempo estrictamente específicos, independiente de los demás procesos que estén programados en código del algoritmo [19]. Estas son las características que se utilizarán en el presente trabajo para regular la frecuencia de muestreo f_s en el proceso de digitalización de una señal eléctrica analógica.

En la sección anterior se mencionó que lo normal es que el micro-controlador ejecute ordenadamente todos los comandos dentro de la función *loop*, sin embargo es muy difícil saber cuánto tiempo tarda cada instrucción en ser ejecutada. Algunos comandos son más complejos que otros y por lo tanto tardan más milisegundos en ser completados, algunos también dependen de variables condicionales (como los típicos *if*) y algunas funciones características del entorno de programación para programar el micro-controlador (como *digitalWrite* o *analogRead*) dependen de un conjunto de otras funciones a nivel de máquina. Los *timer interrupts* permiten detener momentáneamente la ejecución de la secuencia de tareas de la función *loop* en intervalos de tiempo específicos, mientras se ejecuta una lista específica distinta de instrucciones. Una vez que esta lista de instrucciones ha sido ejecutada completamente, el micro-controlador vuelve a ejecutar las tareas originales de la función *loop* a partir de la última línea en que se detuvieron las operaciones.

De esta forma, los *Timer interrupts* son ideales para medir una señal entrante (por alguna de las entradas analógicas del REDBOARD en intervalos equiespaciados de tiempo).

El ATmega 328 cuenta con tres relojes internos: *timer0*, *timer1*, y *timer2*. Cada uno de los relojes posee un contador que es incrementado en cada paso del reloj (ver anexo 7.8). Los *timer interrupts* son ejecutados cada vez que el contador del reloj alcanza un valor específico llamado *compare match register (CMR)*. Cuando un contador de reloj alcanza el valor determinado, se restaura a cero en el siguiente paso del reloj, entonces volverá a contar cada paso del reloj hasta alcanzar nuevamente el valor *CMR*. Las variables que se pueden controlar en los relojes del ATmega328 son el valor del *CMR* y la velocidad con que el contador incrementa su valor en cada paso de reloj. Ergo, se puede controlar la frecuencia con que son ejecutados los *timer interrupts*.

El primer parámetro que entra en juego es la velocidad con que el reloj aumenta el contador. El ATmega328 funciona a $16[MHz]$, esta es la velocidad más rápida en que el contador puede incrementar su valor. A $16[MHz]$, cada paso del reloj ocurre a $\sim 63[ns]$ y un contador tardaría $\frac{10}{16 \times 10^6}$ segundos en alcanzar el valor 9 (el contador comienza en 0), así como tardaría $\frac{1}{16 \times 10^4}$ segundos para alcanzar el valor 99.

Los relojes *timer0* y *timer2* son relojes de 8 [bit], es decir, sus contadores pueden alcanzar un máximo valor de 255. El reloj *timer1* es de 16 [bit], o sea, su contador puede

alcanzar un valor máximo de 65 535. A 16[MHz], aunque se fije el *compare match* igual al máximo valor del contador, los *timer interrupts* ocurrirán cada $\frac{256}{16000000} \sim 16[\mu s]$ para contadores de 8[bit] y cada $\frac{65536}{16000000} \sim 4 [ms]$ para el contador de 16 [bit].

Para controlar la velocidad con que el contador incrementa su valor, se usa una característica llamada *prescaler*. El *prescaler* p determina la frecuencia de incremento del contador f_c , según la ecuación siguiente:

$$f_c = \frac{16 \times 10^6}{p} \quad (3-5)$$

Por ejemplo, usando un *prescaler* $p = 1$, se tendrá una velocidad de incremento en el contador de 16[MHz] y usando $p = 64$ se tendrá $f_c = 250[kHz]$. Cada uno de los tres relojes internos del ATmega328 tiene un *prescaler*. Cada *prescaler* sólo puede tomar los valores 1, 8, 64, 256 y 1024. El valor asignado a un *prescaler* depende de 3 bit específicos internos de cada reloj, llamados *Clock Select Bits*. La siguiente tabla muestra las posibles configuraciones para un reloj interno del ATmega328 y el valor que toma el *prescaler* en cada caso.

| Clock Select Bits | | | Descripción |
|-------------------|--------|--------|--|
| CS_2 | CS_1 | CS_0 | |
| 0 | 0 | 0 | Contador detenido |
| 0 | 0 | 1 | $p = 1$ |
| 0 | 1 | 0 | $p = 8$ |
| 0 | 1 | 1 | $p = 64$ |
| 1 | 0 | 0 | $p = 256$ |
| 1 | 0 | 1 | $p = 1024$ |
| 1 | 1 | 0 | Contador controlado por fuente externa. Incremento en bajo voltaje. |
| 1 | 1 | 1 | Contador controlado por fuente externa. Incremento en alto voltaje. |

Tabla 3-1: Posibles configuraciones para *Clock Select Bits* de un reloj interno.

Con esta información se puede determinar la frecuencia con que se ejecutan los *Timer interrupts* f_{Ti} , según la siguiente ecuación [19]:

$$f_{Ti} = \frac{16 \times 10^6}{p \times (CMR + 1)} \quad (3-6)$$

Reordenando, se obtiene:

$$CMR = \frac{16 \times 10^6}{(p \times f_{Ti})} - 1 \quad (3-7)$$

En el caso particular del presente trabajo de título, se desea que en cada ejecución de los *Timer interrupts* se adquiriera una muestra de la señal analógica que se está

digitalizando, luego la frecuencia de muestreo deseada es, en efecto, igual a la frecuencia de ejecución de los *Timer interrupts*, la ecuación del *compare match register* queda de la siguiente forma:

$$CMR = \frac{16 \times 10^6}{(p \times f_s)} - 1 \quad (3-8)$$

Otra vez, este valor no puede ser mayor que 256 para los relojes *timer0*; *timer2* y debe ser menor o igual que 65 536 para el reloj *timer1*.

Por ejemplo si se desea ejecutar una acción -puntualmente, medir una muestra de una señal eléctrica analógica para crear su representación digital- con una frecuencia del orden de $f_s = 300 [Hz]$ y usando $p = 256$, usando (3-8) se tiene que $CMR = \frac{16 \times 10^6}{(256 \times 300)} - 1 = 207.333$ Pero dado que el *compare match register* es un número entero (límite del contador), aproximamos al entero más cercano, obteniendo $CMR = 207$. Luego usamos (3-6) para calcular la frecuencia de ejecución de los *Timer interrupts*, obteniendo $f_{Ti} = \frac{16 \times 10^6}{256 \times (207 + 1)} = 300.48 [Hz]$. Este valor de frecuencia de muestreo es, como se estudió en el apartado 3.4.1, útil para identificar notas musicales más graves que un Re_3 .

Finalmente, las acciones que deben ser ejecutadas con la frecuencia de los *Timer interrupts* deben ser declaradas dentro de una función llamada $ISR(TIMER0_COMPA_vect)$; $ISR(TIMER1_COMPA_vect)$; $ISR(TIMER2_COMPA_vect)$ correspondientemente para cada uno de los relojes.

```

sketch_nov28a $
int muestra = 0;
void setup()
{
  cli(); //Detener los interrupts
  //set timer1 interrupt at 300.48Hz
  TCCR1A = 0; // set entire TCCR1A register to 0
  TCCR1B = 0; // same for TCCR1B
  TCNT1 = 0; //Inicializar el contador desde 0
  OCR1A = 207; // = (16*10^6) / (200*256) - 1 (debe ser <65536)
  // Habilitar CTC mode
  TCCR1B |= (1 << WGM12);
  // Activar bit CS12 para un prescaler=256
  TCCR1B |= (1 << CS12);
  // Activar timer compare interrupt
  TIMSK1 |= (1 << OCIE1A);
  sei(); //permitir interrupts
}
void loop(){}
ISR(TIMER1_COMPA_vect)
{
  muestra = analogRead(A0);
}

```

Figura 3-19: Código para medir una muestra de una señal analógica con $f_s \sim 300 [Hz]$.

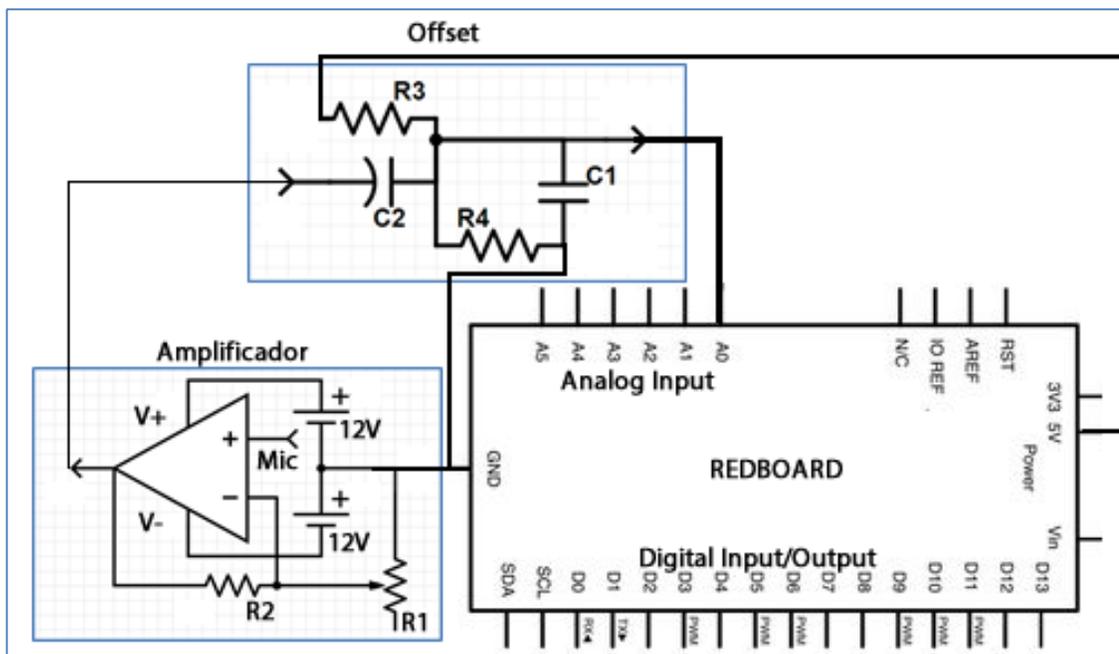


Figura 3-20: Esquema del circuito formado por el micrófono, el amplificador, el offset y el REDBOARD.

La Figura 3-19 muestra la implementación en el entorno de programación del código que permite fijar los *Timer interrupts* del *timer1* del ATmega328 a una frecuencia de muestreo. El micro-controlador, almacenará en la variable *muestra* un entero entre 0 y 1023 (tal como se explicó en las secciones 3.2 y 3.3) que representa un valor entre 0 y 5 [V] de la señal que llega a la entrada analógica A0 del REDBOARD cada vez que el reloj cumpla el correspondiente ciclo.

La Figura 3-20 muestra el esquema del montaje que incluye el micrófono que captura el sonido desde el instrumento que se desea afinar, el amplificador no inversor propuesto en la sección 3.2.1, el bloque offset propuesto en la sección 3.3.1 y la placa REDBOARD. Notar que los voltajes de referencia a tierra, tanto del amplificador como del offset, han sido conectados directamente al terminal GROUND del REDBOARD, de la misma forma que el pin de 5 [V] del divisor de voltaje del bloque offset ha sido reemplazado por la alimentación auxiliar del REDBOARD que entrega, justamente, una tensión eléctrica constante de 5 [V].

3.4.4 Indicador de saturación

En el apartado 3.2 se mencionó que durante el proceso de digitalización, cualquier valor de la señal analógica que esté por sobre los 5[V] será ignorado y se le asignará arbitrariamente el número entero 1023 (de la misma forma que los valores negativos de la señal analógica eran ignorados y se les asignaba el entero 0). Existe la posibilidad de que la señal analógica, durante el proceso de amplificación, sea sobre amplificada, ya sea porque el factor de amplificación del amplificador no inversor está ajustado a un valor muy alto, o porque la señal obtenida directamente del micrófono de contacto no es tan débil cómo se estimaba.

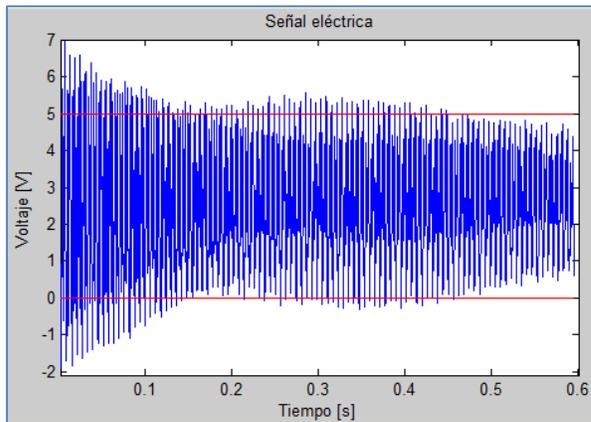


Figura 3-21: Señal analógica amplificada en exceso.

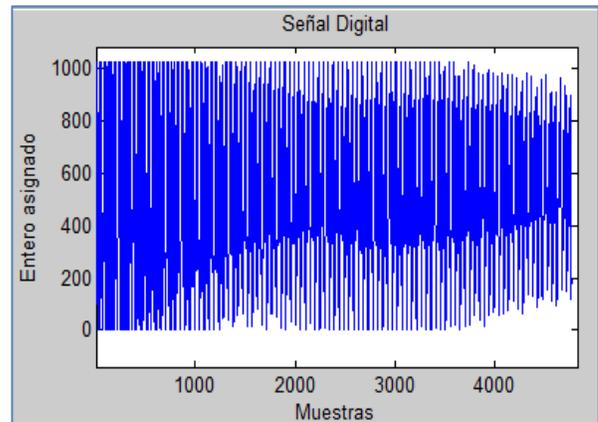


Figura 3-22: Digitalización distorsionada.

La Figura 3-21 muestra una simulación de una señal analógica tratada por los circuitos propuestos en las secciones 3.2.1 y 3.3.1, pero que en la etapa de amplificación existió un exceso de amplificación, es decir, el voltaje V_{pp} supera los 5[V]. Se observa en rojo los límites en voltaje que la señal analógica está sobrepasando. La Figura 3-22 muestra una simulación del proceso de digitalización de la señal sobre amplificada. Se observa que hay información que se pierde, todos los valores sobre 5[V] y bajo 0[V] son ignorados y se les asignan los enteros 1023 y 0 correspondientemente.

Para evitar este fenómeno, simplemente habría que ajustar el potenciómetro del amplificador no inversor (ver apartado 3.2) para disminuir el factor de amplificación. Lamentablemente, cuando se está usando el dispositivo afinador, no se cuenta con gráficos de la señal entrante como los que aquí han sido expuestos y no hay forma de saber en tiempo real si se está perdiendo información en el proceso de digitalización de la señal.

Sin embargo, lo que sí se puede hacer es incorporar un indicador de saturación, a través de un LED luminoso que se encienda cada vez que la señal analógica alcance los límites del rango de tolerancia de medición del micro-controlador.

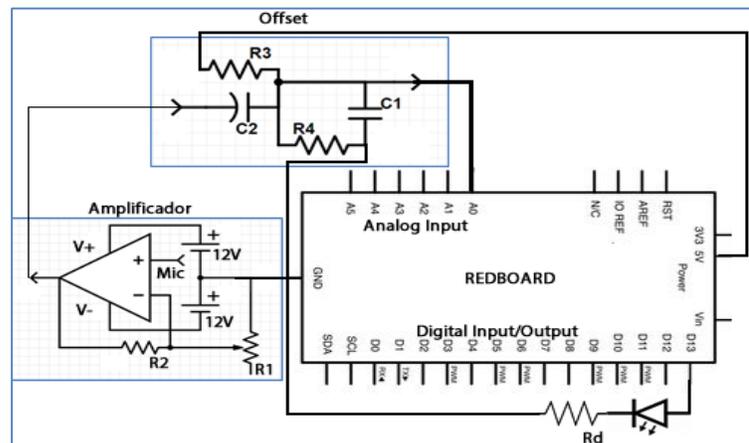


Figura 3-23: Esquema del circuito incluyendo indicador de saturación.

```

int muestra = 0;
void setup(){
  cli(); //Detener los interrupts
  //set timer1 interrupt at 300.48Hz
  TCCR1A = 0; // set entire TCCR1A register to 0
  TCCR1B = 0; // same for TCCR1B
  TCNT1 = 0; //Inicializar el contador desde 0
  OCR1A = 207; // = (16*10^6) / (200*256) - 1 (debe ser <65536)
  // Habilitar CTC mode
  TCCR1B |= (1 << WGM12);
  // Activar bit CS12 para un prescaler=256
  TCCR1B |= (1 << CS12);
  // Activar timer compare interrupt
  TIMSK1 |= (1 << OCIE1A);
  sei(); //permitir interrupts
  pinMode(13, OUTPUT);
}
void loop(){}
ISR(TIMER1_COMPA_vect){
  muestra = analogRead(A0);
  if(muestra==0 || muestra==1023)
    digitalWrite(13,HIGH);
  else
    digitalWrite(13,LOW);
}

```

Figura 3-24: Código que incorpora el indicador de saturación.

La Figura 3-23 muestra el esquema del montaje anterior, en que además se ha incluido un LED en el terminal digital 13 del REDBOARD, junto con una resistencia R_d cerrando en el terminal GROUND, cuando el terminal digital se activa con voltaje alto, se produce una tensión eléctrica de 5[V], por esta razón (para resguardar el LED) en el diseño de este trabajo de titulación se ha escogido una resistencia $R_d = 330[\Omega]$, luego la corriente que circula por el LED (ver anexo 7.9) no es mayor a 1.52[mA]. Este LED se encenderá cada vez que la señal digital pierda información o esté saturada.

La Figura 3-24 muestra el mismo código anterior, que ajusta la frecuencia de muestreo del ATmega328 a 300.48[Hz], pero esta vez en la función *setup* se ha establecido que el terminal 13 digital del REDBOARD funcionará como salida. Además, se ha incorporado en el *Timer interrupt* del reloj *timer1* que luego de medir una muestra de la señal que llega por la entrada analógica A0 del REDBOARD, se verifica si el valor asignado está en los extremos del rango de tolerancia, en tal caso se activa un voltaje alto en el pin 13 (se enciende el LED) o de lo contrario se activa un voltaje bajo (se apaga el LED).

Entonces, cuando el usuario del dispositivo afinador vea encender el LED que indica saturación de la señal, simplemente podrá ajustar el potenciómetro del amplificador no inversor hasta que el LED se apague.

3.5 Identificación de la frecuencia

Hasta ahora se han abordado los problemas de capturar el sonido del instrumento musical, tratar la señal capturada para prepararla para el proceso de digitalización y ajustar el micro-controlador con una frecuencia de muestreo adecuada que permita identificar notas musicales según su espectro de frecuencia (digitalización). En otras palabras, el diseño está cubierto hasta los 4 primeros bloques de la Figura 3-1. El siguiente paso es la identificación de la frecuencia fundamental de la señal digitalizada.

El dispositivo afinador que se propone en este trabajo de título debe ajustar la tensión de la cuerda del instrumento musical hasta que se produzca un sonido de frecuencia específica. Antes de tomar la decisión de mover la clavija y determinar en qué sentido o cuánto moverla es necesario tener una lectura de cuál es la frecuencia que se está produciendo antes de la afinación. En este apartado se atacará el concepto que envuelve el quinto bloque de la Figura 3-1.

Para usar la herramienta matemática de la Transformada de Hartley (ver sección 2.7) se carga una biblioteca informática¹ llamada *Arduino FHT Libray* [20] dedicada a calcular vectores con información de la densidad espectral de la Transformada. Esta librería usa el algoritmo *Fast Hartley Transform* (FHT) [10] que es análogo a la FFT (ver sección 2.6) con la única diferencia que no usa números complejos en el cálculo.

¹Una biblioteca es un conjunto de características, codificadas en un lenguaje de programación determinado, que pone a disposición una interfaz de funciones para ser utilizadas en otro código.

3.5.1 Uso de la biblioteca FHT

Las herramientas que pone a disposición esta biblioteca informática son bastante intuitivas, se cuenta con varias funciones que se deben invocar de forma secuencial y ordenada, las funciones que pone a disposición la *Arduino FHT Libray* son las siguientes:

| Función | Descripción |
|----------------------|--|
| fht_run() | Esta es la función principal de la biblioteca, esencialmente lo que hace es ejecutar el algoritmo de la Transformada Rápida de Hartley. No recibe variables de entrada y tampoco retorna variables de salida. Al usar esta función, se asume que ya existe un vector con las muestras de la señal adquirida almacenado en la memoria RAM del ATmega328 y que además este vector ha sido reordenado por medio de la función <i>fht_reorder()</i> . Las muestras deben ser almacenadas en un vector llamado <i>fht_input[]</i> , que contiene un valor de $2[B]$ por cada muestra de la señal discreta. En la sección 3.4.1 se estudió que con una cantidad de muestras $N = 256$ este vector usa $512[B]$ de memoria. El resultado final es almacenado en el mismo vector <i>fht_input[]</i> . Notar que se trata sólo de la Transformada de Hartley y no de la densidad de frecuencias. |
| fht_reorder() | Esta función ordena los datos del vector <i>fht_input[]</i> de una forma conveniente para ejecutar el algoritmo FHT. No recibe variables de entrada y tampoco retorna variables de salida. Las muestras de vector <i>fht_input[]</i> , deben ser completadas antes de ejecutar esta función. Esta función debe ser ejecutada antes de invocar <i>fht_run()</i> . |
| fht_mag_lin() | Esta función entrega la densidad espectral de frecuencias. Realiza las operaciones equivalentes a calcular el modulo de los números complejos de la Transformada de Fourier. Los datos, previamente ordenados y operados por la ventana, son tratados desde el vector <i>fht_input[]</i> . La densidad espectral es retornada en un vector llamado <i>fht_lin_out[]</i> . Este vector contiene $\frac{N}{2}$ ya que la Transformada es redundante es su última porción de datos [8]. |

Tabla 3-2: Funciones de la biblioteca FHT.

En el código propuesto en la Figura 3-24 se implementa un detector de saturación, pero en cada ejecución del *Timer interrupt* de reloj *timer1* se está almacenando una muestra de la señal que llega a la entrada analógica A0 del REDBOARD. En un contexto de uso del dispositivo afinador, el usuario no estará pulsando la cuerda siempre y en todo momento, es por eso que el dispositivo debe ser capaz de detectar cuando comenzar a digitalizar la señal, esto es, en qué momento comenzar a llenar con muestras el vector *fht_input[]*. Al pulsar la cuerda del instrumento que se desea afinar, la señal presenta un pico en su amplitud, este fenómeno será utilizado para gatillar el llenado del vector que será utilizado para la determinación de la frecuencia.

Se establece un umbral entonces, a partir de dos números enteros en el rango del proceso de digitalización, cuando la señal digital alcanza o sobrepasa alguno de estos dos números, comenzará el llenado del vector *fht_input[]*.

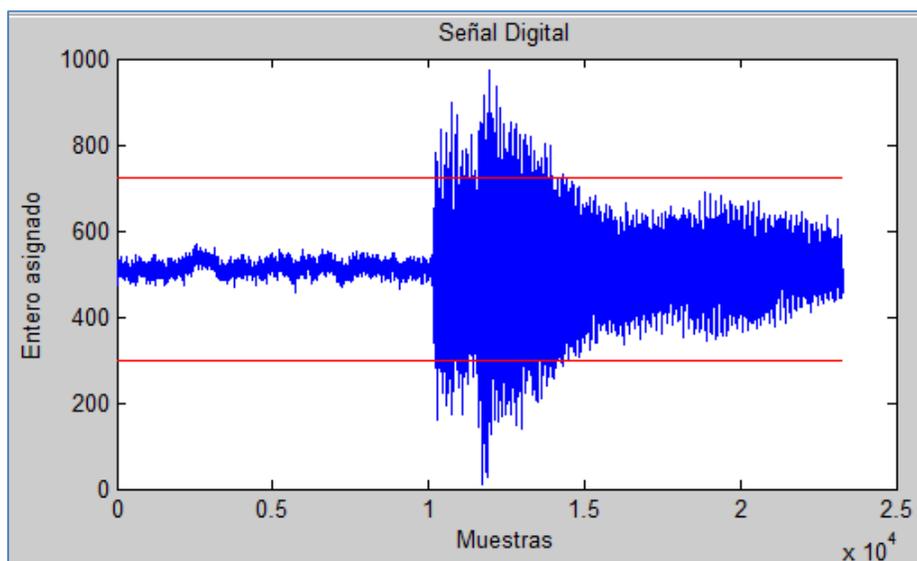


Figura 3-25: Umbral de detección de pulsación de la cuerda.

```

87 ISR(TIMERO_COMPA_vect){
88     muestra = analogRead(A0); //Se almacena muestra medida desde la entrada analógica.
89     if(muestra==0 || muestra==1023){ //Detector de saturación.
90         digitalWrite(13,HIGH);
91     }
92     else{
93         digitalWrite(13,LOW);
94         if(muestra<=420 || muestra>=1023-420){ //Umbral para llenar vector fht_input[]
95             umbral=1; //La cuerda ha sido pulsada.
96             if(umbral && !lleno){ // Si la cuerda fue pulsada y fht_input[] no ha sido completado.
97                 int margen = (int)0.3*Fs; //Margen para no capturar la pulsación de la cuerda.
98                 if(ind>=margen){ //Dejar pasar 0.3[s] después de la pulsación de la cuerda.
99                     fht_input[ind-margen]=muestra; //Se guarda la muestra.
100                    ind++; //Aumenta el índice auxiliar de llenado de fht_input[]
101                    if(ind-margen>=FHT_N) //Si ya se llenó el vector fht_input[]
102                    {
103                        ind=0; //Restauramos desde cero el índice auxiliar de llenado de fht_input[]
104                        umbral=0; // Se espera la próxima pulsación de la cuerda.
105                        lleno=1; //Boolean que indica que el vector fht_input[] ya ha sido completado.
106                    }
107                }
108            }
109        }
110    }
111 }

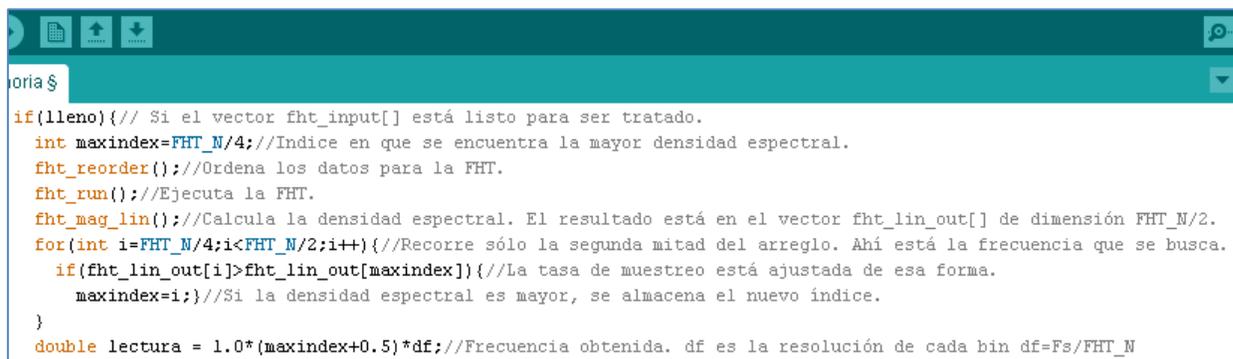
```

Figura 3-26: Implementación del algoritmo para llenar el vector *fht_input[]*.

La figura Figura 3-25 muestra una captura digital de sonido de la pulsación de la cuerda de una guitarra. En rojo se ha representado un umbral que debe sobrepasar la señal para que el micro-controlador comience el proceso de llenado del vector *fht_input[]* con muestras de la señal. Se observa que antes de la pulsación de la cuerda la señal es muy débil, este fenómeno le indica al algoritmo que sigue el micro-controlador, que esa parte de la señal son datos que deben ser ignorados porque no corresponden a sonido emitido por el instrumento musical.

La Figura 3-26 muestra el algoritmo que se ejecuta cada vez que el reloj *timer0* activa su *timer interrupt*, en esta porción de código se implementa tanto el detector de saturación (que enciende un LED en el pin digital 13 del REDBOARD cada vez que la señal alcanza el límite de tolerancia del micro-controlador) como el llenado del vector *fht_input[]* al que se le podrá calcular la Transformada de Hartley. Se observa que una vez que se detectó que la señal alcanza niveles sobre un umbral establecido, además se dejan pasar 300[ms]. Esto es para evitar almacenar información no deseada en el vector. Las frecuencias características y modos normales del material de la cuerda o del mediador con que se pulsó la cuerda podrían caer dentro del vector de medición y ser confundidas con la frecuencia fundamental de la nota musical que se desea identificar.

El proceso de llenado del vector *fht_input[]* fue programado dentro de la función *ISR(TIMERO_COMPA_vect)* porque era necesario hacerlo en intervalos regulares de tiempo (según la tasa de muestreo establecida). Una vez que este vector ha sido completado y ya tiene las 256 muestras, es necesario ejecutar las funciones explicadas en los párrafos anteriores para poder obtener la frecuencia de la señal medida.



```

oria $
if(!lleno){// Si el vector fht_input[] está listo para ser tratado.
    int maxindex=FHT_N/4;//Índice en que se encuentra la mayor densidad espectral.
    fht_reorder();//Ordena los datos para la FHT.
    fht_run();//Ejecuta la FHT.
    fht_mag_lin();//Calcula la densidad espectral. El resultado está en el vector fht_lin_out[] de dimensión FHT_N/2.
    for(int i=FHT_N/4;i<FHT_N/2;i++){//Recorre sólo la segunda mitad del arreglo. Ahí está la frecuencia que se busca.
        if(fht_lin_out[i]>fht_lin_out[maxindex]){//La tasa de muestreo está ajustada de esa forma.
            maxindex=i;}//Si la densidad espectral es mayor, se almacena el nuevo índice.
        }
    }
    double lectura = 1.0*(maxindex+0.5)*df;//Frecuencia obtenida. df es la resolución de cada bin df=Fs/FHT_N
}

```

Figura 3-27: Cálculo de la frecuencia de la señal medida.

La Figura 3-27 muestra la porción de código que va dentro de la función *loop()* en que se ejecutan las funciones de la biblioteca FHT correspondientes al cálculo de la Transformada de Hartley y luego su densidad espectral de frecuencias. Notar que el vector de salida que contiene la densidad espectral es de dimensión $\frac{N}{2}$ (ver Tabla 3-2) y sólo se recorre su segunda mitad en búsqueda del bin con la mayor densidad espectral. Esto es una técnica para acelerar el algoritmo. La tasa de muestreo f_s está ajustada de tal manera que la frecuencia que se desea medir está siempre en la segunda mitad. Detalles sobre la programación de la tasa de muestreo se encuentran en los siguientes apartados.

3.6 Motor SERVO

Un motor SERVO es un motor eléctrico que tiene la particularidad de poder ser controlado para ubicar su eje rotacional en una posición específica dentro de un rango de funcionamiento y fijar esa posición hasta que se le dé una orden diferente. Los motores SERVO más comunes tienen un rango de funcionamiento de 180° , esto es, no pueden dar un giro completo sino que su posición debe ser fijada dentro del rango desde 0 hasta 180° . Un motor SERVO puede ser controlado por un micro-controlador como el ATmega328 que se está usando en este trabajo. El motor SERVO que se utiliza en el diseño del afinador electromecánico corresponde al modelo MG995 del fabricante TOWER PRO (ver anexo 7.10).

Un motor SERVO funciona con una serie de engranajes que provocan un movimiento más lento y más preciso del eje de rotación, junto con un potenciómetro que modifica el valor de su resistencia eléctrica con el movimiento de los engranajes. El motor cuenta con 3 terminales eléctricos, uno va conectado al voltaje de referencia GROUND, otro a una alimentación que puede operar en el rango de los $4.8[V]$ y $7.2[V]$, y un tercer terminal que se usa para transmitir el parámetro que indica la posición deseada desde el micro-controlador. La orden que se le da al motor SERVO desde el micro-controlador indica el valor que debe alcanzar la resistencia del potenciómetro para que el motor deje de girar. De esta forma, es posible dar un parámetro indicando el ángulo de la posición deseada al motor SEVO, para que este gire hasta fijar su eje de rotación a un ángulo específico.

Existe una biblioteca informática para poder programar órdenes en el entorno Arduino e incluir instrucciones dentro del algoritmo que seguirá el ATmega328 que servirán para controlar el motor SERVO.

3.6.1 Uso de la biblioteca SERVO

Las herramientas que pone a disposición la biblioteca informática SERVO son de fácil uso y se invocan como métodos implementados en una clase llamada Servo, más adelante se verán ejemplos que clarifican qué significa esto. Se cuenta con varias funciones que se deben invocar de forma secuencial y ordenada, las funciones que pone a disposición la *SERVO Libray* son las siguientes:

| Función | Descripción |
|---------------------------------|--|
| <i>attach(int pin)</i> | Se usa para indicar al micro-controlador cuál será el terminal digital del REDBOARD desde el que se controlará el motor SERVO, esto es, en qué pin serán enviadas las órdenes de control del motor. Recibe como parámetro un entero correspondiente al número del pin del REDBOARD correspondiente al control del motor. |
| <i>write(int angulo)</i> | Es la función que da la orden de posicionamiento al motor. Recibe como parámetro un entero entre 0 y 180, indicando en grados la posición en que será fijado el eje de rotación. Al ejecutar esta función el motor SERVO cambia su posición hasta el valor indicado en el parámetro <i>angulo</i> . |
| <i>detach()</i> | Se usa para liberar el pin del REDBOARD desde el que el motor SERVO está siendo controlado. No recibe parámetros y al ejecutar esta función, el motor deja de recibir órdenes desde el micro-controlador y se detiene, dejando de ejercer torque. |

Tabla 3-3: Funciones de la biblioteca SERVO.

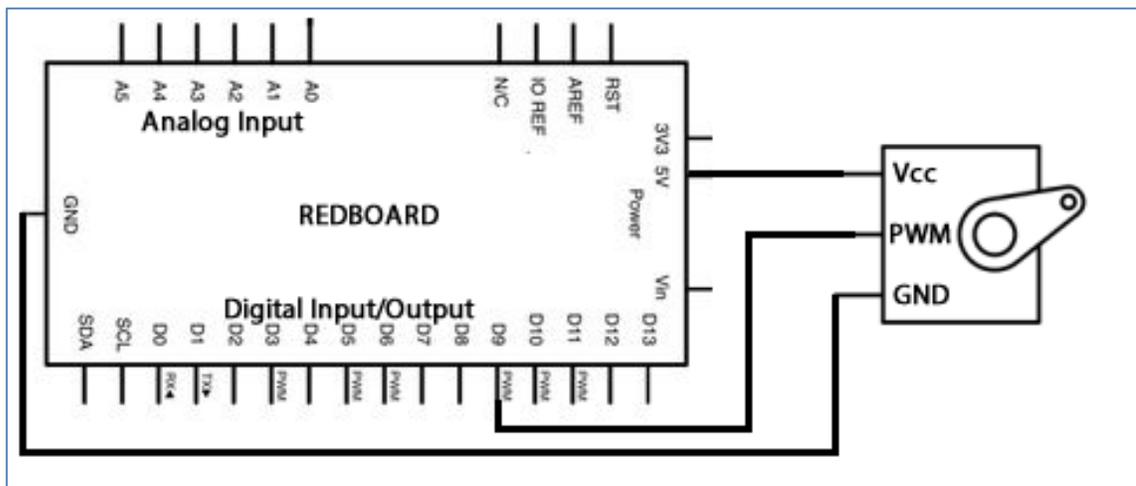
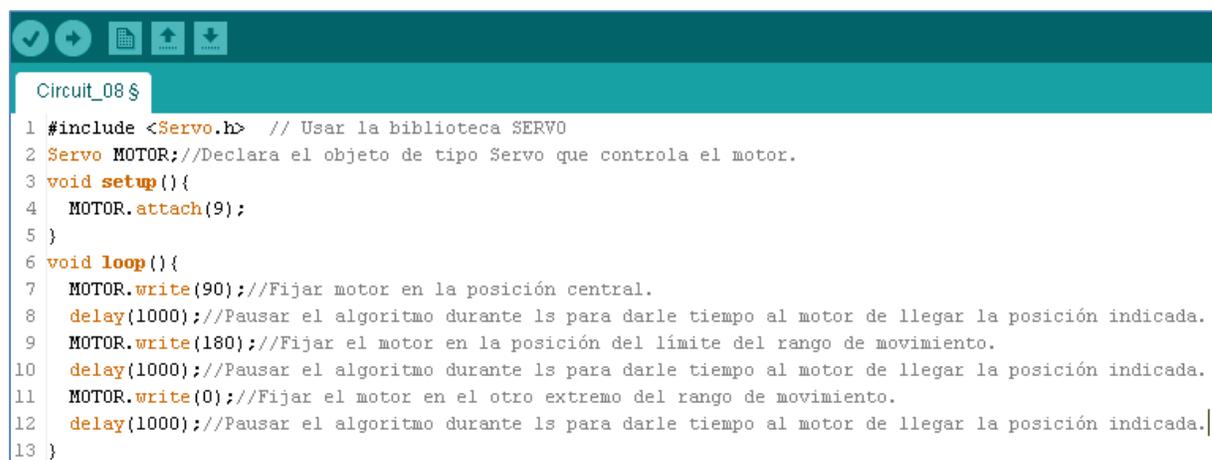


Figura 3-28: Esquema de un ejemplo de montaje entre el REDBOARD y el motor SERVO.



```
Circuit_08 $
1 #include <Servo.h> // Usar la biblioteca SERVO
2 Servo MOTOR;//Declara el objeto de tipo Servo que controla el motor.
3 void setup(){
4     MOTOR.attach(9);
5 }
6 void loop(){
7     MOTOR.write(90);//Fijar motor en la posición central.
8     delay(1000);//Pausar el algoritmo durante 1s para darle tiempo al motor de llegar la posición indicada.
9     MOTOR.write(180);//Fijar el motor en la posición del límite del rango de movimiento.
10    delay(1000);//Pausar el algoritmo durante 1s para darle tiempo al motor de llegar la posición indicada.
11    MOTOR.write(0);//Fijar el motor en el otro extremo del rango de movimiento.
12    delay(1000);//Pausar el algoritmo durante 1s para darle tiempo al motor de llegar la posición indicada.
13 }
```

Figura 3-29: Ejemplo de uso de la biblioteca SERVO.

La Figura 3-28 muestra un ejemplo de montaje para controlar el motor SERVO a través del REDBOARD. La Figura 3-29 muestra un código que se puede ejecutar usando tal montaje. El algoritmo muestra que para controlar el motor, primero es necesario declarar un objeto de clase *Servo* (en este ejemplo llamado *MOTOR*) y luego referenciarlo para hacer uso de las distintas funciones. En este ejemplo, el motor es controlado desde el pin 9 del REDBOARD y lo que se debería observar es que el motor mueva su eje de rotación hasta la posición central, luego de un segundo el motor se comienza a mover hasta el extremo de su rango de funcionamiento (180°), luego de un segundo el motor se mueve hasta el extremo opuesto (0°) y luego de un segundo comienza la rutina nuevamente.

Hasta ahora se ha visto cómo se controla un motor SERVO por medio de un código para programar el ATmega328, pero este tipo de motor tiene la gran limitante de no poder realizar giros completos. En un contexto de ajuste de tensión de una cuerda de un instrumento musical no sería útil usar este tipo de motores, porque muchas veces es necesario hacer giros completos de la clavija para obtener el tono deseado. Afortunadamente, es posible modificar el motor SERVO para alterar el control de su funcionamiento [21] y poder hacer que responda de forma distinta a las funciones estudiadas en el presente apartado, logrando giros continuos en 360° convenientes para la aplicación que en este trabajo se presenta.

3.6.2 Modificación del motor SERVO para giro continuo

En el apartado anterior se explicó que un motor SERVO funciona con un potenciómetro interno que cambia su resistencia para establecer el límite de giro y llegar hasta una posición deseada. Si se cambia el potenciómetro por un par de resistencias fijas, entonces el límite de giro ya no podrá ser establecido y el motor podrá continuar haciendo girar su eje de forma indefinida. Además, también es necesario eliminar un tope físico que existe en los engranajes internos del motor que impiden su giro continuo. El motor MG995 utilizado en el diseño del dispositivo afinador propuesto en este trabajo ha sido modificado de esta manera para poder ajustar, por medio del giro de la clavija, la tensión de una cuerda de un instrumento musical. Los detalles sobre cómo se realizó la modificación se encuentran claramente explicados en la referencia bibliográfica [21].

Luego de la modificación explicada en este apartado, no se altera la forma de controlar el motor SERVO, es decir, las funciones de la biblioteca informática siguen siendo las mismas. Lo único que cambia es la forma en que el motor responde frente a la ejecución de la función *write()*. Originalmente, lo que hacía *write()* era fijar una posición para el eje de rotación, indicado en un parámetro angular. Para un motor SERVO modificado, lo que se interpreta de tal parámetro no es una posición, sino una velocidad.

El rango del parámetro que recibe la función *write()* sigue siendo el mismo, esto es, no se puede ingresar un valor menor que 0 o mayor que 180. La diferencia es que luego de haber modificado el motor, este parámetro será interpretado como una velocidad de giro en ambos sentidos escalada en el rango desde 0 hasta 180. Por ejemplo, si se recibe el como parámetro el número 0, entonces el motor girará a su máxima velocidad en sentido horario (torque negativo, movimiento para “desatornillar”), este movimiento aplicado en una clavija de un instrumento musical provocaría un aumento en la tensión de la cuerda y por lo tanto un aumento en su frecuencia de vibración. Si se recibe como parámetro de la función *write()* el número 180, entonces el motor SERVO modificado girará a su máxima velocidad en sentido anti-horario (torque positivo, movimiento para “atornillar”), este movimiento aplicado en la clavija de un instrumento musical provocaría una disminución de la tensión de la cuerda y por lo tanto una nota musical más grave. Si la función *write()* recibe como parámetro el número 90, entonces el motor detendría su movimiento.

De esta forma, al ejecutar el código expuesto en la Figura 3-29 usando un motor SERVO modificado, como el que se utiliza en este trabajo, se debería observar un ciclo de 3 segundos repetirse indefinidamente: durante el primer segundo el motor está detenido; durante el segundo segundo el motor gira a su máxima velocidad en sentido anti-horario; y durante el tercer segundo el motor gira a su máxima velocidad en sentido horario.

La velocidad máxima de giro del motor utilizado en este trabajo es de $300[^\circ/s]$, es decir, el motor tarda $1.2[s]$ en dar una vuelta completa.

3.6.3 Adaptación del motor para clavijas

Finalmente, en este apartado se comienza a cubrir la etapa indicada en el sexto bloque de la Figura 3-1. En efecto, luego de poder controlar el motor SERVO adecuadamente, es necesario atender la interfaz mecánica que ligará el movimiento del eje de rotación del motor, con el movimiento del tornillo sinfín de la clavija del instrumento musical que se desea afinar.

Existe una herramienta utilizada por luthiers y músicos llamada bobinadora (ver Figura 3-30). Este tipo de herramientas se encuentra disponible en cualquier tienda de música y se usa para acelerar el proceso de encordado de un instrumento musical. Básicamente se trata de una manivela de plástico que permite ejercer un torque mayor (y por lo tanto, giros más rápidos) sobre la clavija del instrumento musical (ver Figura 3-31).



Figura 3-30: Bobinadora para encordar guitarras.



Figura 3-31: Uso de bobinadora en ukelele.

Para permitir que el motor transmita su torque y movimiento sobre la clavija del instrumento musical que se desea afinar, se desarmó una bobinadora común y corriente, recuperando el extremo que encaja sobre la clavija del instrumento y se fijó esta pieza al eje de rotación del motor servo. De esta forma queda cubierto el diseño de la interfaz mecánica que puede interactuar con el instrumento que se desea afinar. El resultado del montaje se puede observar en la Figura 3-32 y en la Figura 3-33.



Figura 3-32: Motor SERVO adaptado para mover clavijas de instrumentos musicales.



Figura 3-33: Motor SERVO siendo utilizado en las clavijas de un Laudón.

¹ Imagen extraída del sitio web de FENDER, fabricante de instrumentos musicales.

3.6.4 Inclusión del motor en el montaje del dispositivo afinador

Ahora es necesario hacer que el micro-controlador haga girar el motor SEVO modificado en función de la frecuencia medida de la señal digital capturada. La velocidad y el sentido de giro del motor, deben estar controlados por la diferencia que existe entre la frecuencia de la nota musical correspondiente a la cuerda que se está afinando y la frecuencia de la señal medida. Para lograr esto, primero se implementa una simple función matemática en el código del algoritmo, que básicamente corresponde a la versión informática de la ecuación (2-5). Esta función es llamada *freq()* y recibe como parámetros dos números enteros correspondientes a la nota y la octava en que se desea calcular la frecuencia. La implementación de esta función se puede ver en la Figura 3-34.

```
119 float freq(int n, int o){
120     return 440.0*pow(2.0, (1.0*(o-4)*12+(n-10))/12.0);
121 }
```

Figura 3-34: Función *freq()*.

El siguiente paso es agregar el motor SEVO al montaje completo, incluyendo los circuitos propuestos en las secciones 3.2.1 y 3.3.1 junto con el REDBOARD. El esquema de este montaje se puede observar en la Figura 3-35.

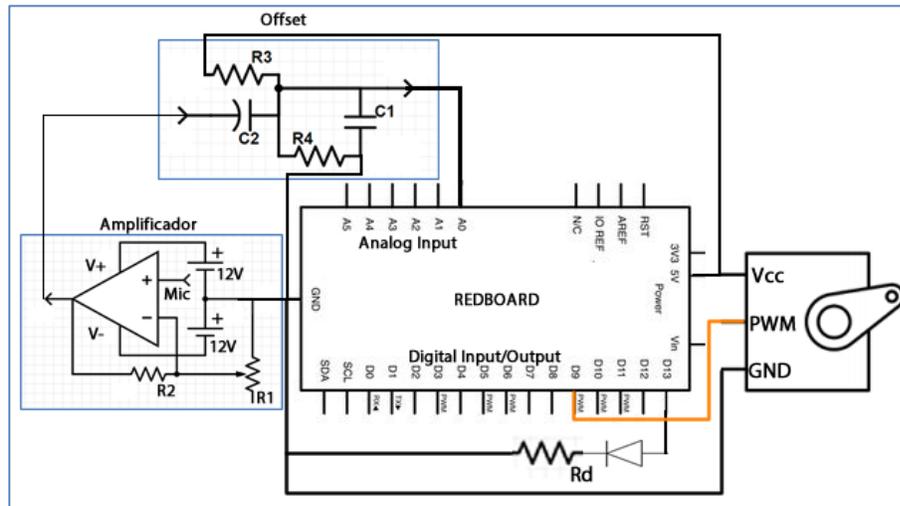


Figura 3-35: Esquema del montaje con REDBOARD, amplificador, offset y motor SERVO.

El código expuesto en la Figura 3-27 definía el cálculo de la frecuencia de la señal capturada haciendo uso de la biblioteca FHT. Ahora, a partir de ese valor, es necesario hacer uso de la biblioteca SERVO para controlar el giro del motor. En la Figura 3-36 se muestra el mismo código, esta vez completado para hacer girar el motor SERVO luego de haber almacenado la frecuencia de la señal medida en la variable *lectura*.

```

if(lleno){//Si el vector fht_input[] está listo para ser tratado.
    int maxindex=FHT_M/4;//Indice en que se encuentra la mayor densidad espectral.
    fht_reorder();//Ordena los datos para la FHT.
    fht_run();//Ejecuta la FHT.
    fht_mag_lin();//Calcula la densidad espectral
    for(int i=FHT_M/4;i<FHT_M/2;i++){//Recorre sólo la segunda mitad del arreglo. Ahí está la frecuencia que se busca.
        if(fht_lin_out[i]>fht_lin_out[maxindex]){//La tasa de muestreo está ajustada de esa forma.
            maxindex=i;//Si la densidad espectral es mayor, se guarda el nuevo índice.
        }
        double lectura = 1.0*(maxindex+0.5)*df;//Frecuencia obtenida. df es la resolución de cada bin.
        double inferior = freq(nota-3,octava);//Frecuencia de 3 semitonos más grave que la nota deseada.
        double superior=freq(nota+3,octava);//Frecuencia de 3 semitonos más agudo que la nota deseada.
        double lec = log(lectura)/log(2);//Logaritmo en base2 de las últimas 3 variables calculadas
        double inf = log(inferior)/log(2);//Esto se usa para linealizar la distancia en frecuencia entre cada nota.
        double sup=log(superior)/log(2);
        lleno=0;//Ya se usó el vector fht_input y está listo para ser completado nuevamente.
        if(lectura > inferior && lectura < superior){// Si la frecuencia detectada está dentro del rango
            MOTOR.attach(9);//El motor recibe ordenes dese el PIN 9.
            int giro = Escala(lec,inf,sup);//Transforma linealmente para obtener una orden.
            MOTOR.write(giro);//Da la orden de girar.
        }
        else{//Si la lectura está fuera de los límites definidos.
            MOTOR.detach();//Se detiene el motor.
        }
    }
}

```

Figura 3-36: Control del motor usando la frecuencia de la señal medida.

Además se definen dos variables: *superior* e *inferior*. Estas variables definen un rango dentro del cual debe encontrarse la frecuencia de la señal capturada para que se tome alguna decisión de giro del motor. Si *lectura* está fuera de este rango, significa que la frecuencia medida es demasiado lejana a la frecuencia deseada, esto puede implicar que se esté pulsando una cuerda distinta a la que se desea afinar o que la frecuencia deseada ha sido mal ajustada, en ese escenario se detiene el motor y no se realiza ninguna acción. Las variables *superior* e *inferior* son calculadas por medio de la función *freq()* y corresponden a las frecuencias características que tienen las notas musicales tres semitonos más grave y tres semitonos más agudo con respecto a la nota musical deseada.

A continuación se definen las variables *lec*, *inf* y *sup*. Que corresponden a el $\log_2()$ de *lectura*, *inferior* y *superior*. Esto es para llevar los valores de estas tres frecuencias a una escala lineal (ver apartado 2.3). Luego se cambia el valor de verdad de la variable *lleno*, para indicar que el vector *fht_input[]* ya ha sido utilizado y puede volver a ser completado en otra medición. La variable *giro* contiene en parámetro que dará la orden de girar al motor SERVO y se obtiene usando la función *Escala()* (ver Tabla 3-5), esta función realiza una transformación lineal desde el rango del intervalo definido por (*inf;sup*) hacia el rango que define el intervalo (*0;180*).

Si la señal medida tiene una frecuencia más baja que la frecuencia deseada (es necesario aumentar la tensión de la cuerda), la variable *giro* tendrá un valor entre 0 y 90, por lo tanto se realizará un torque negativo (ver apartado 3.6.2) proporcional a la distancia entre la frecuencia medida y la frecuencia deseada. Si la señal medida tiene una frecuencia más alta que la frecuencia deseada (es necesario disminuir la tensión de la cuerda), la variable *giro* tendrá un valor entre 90 y 180, entonces se realizará un torque positivo. Aquí concluye el diseño en aspectos funcionales del dispositivo.

3.7 Sistema controlado

Dentro del marco de la teoría de control, el diseño propuesto hasta este punto, junto con el flujo que circula por los elementos en cascada y sus distintas funciones corresponde a un lazo de control (ver apartado 2.9).

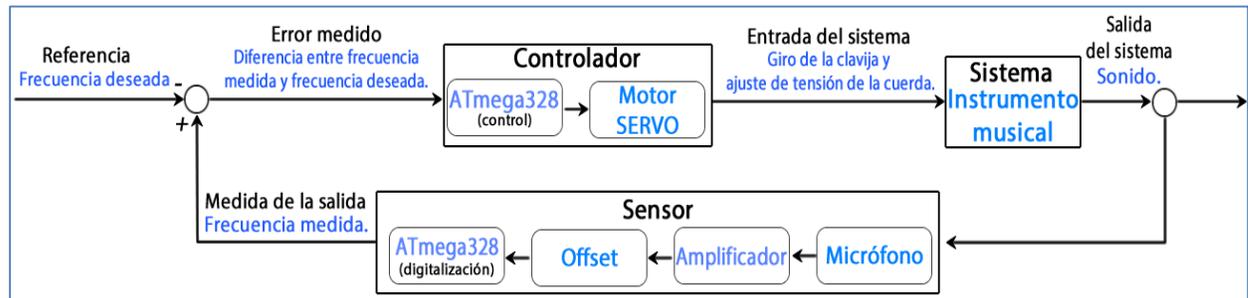


Figura 3-37: Lazo de control del dispositivo afinador.

La Figura 3-37 muestra el lazo de control que conforma el funcionamiento del dispositivo afinador diseñado hasta este punto del presente trabajo. La referencia del sistema sería la frecuencia deseada; el error medido corresponde a la diferencia entre la frecuencia medida y la frecuencia deseada; el controlador es, naturalmente el ATmega328 junto con el motor SERVO; el sistema corresponde al instrumento musical que se está afinando; y el sensor corresponde al conjunto formado por el micrófono, el amplificador, el offset y las funcionalidades del ATmega328 que digitalizan la señal y calculan su frecuencia. Notar que en este caso se trata de un control con realimentación negativa. En efecto, al valor censado se le resta el valor deseado (similar al lazo de control de la Figura 2-29). El diseño se ha dispuesto de esta forma para que el giro del motor tenga sentido con respecto al movimiento de la clavija que tensa o afloja la cuerda del instrumento (ver apartado 3.6). Otro detalle particular de este lazo cerrado es que el ATmega328 aparece tanto en el cuadro de controlador como en el cuadro de sensor. Esto se explica porque el micro-controlador participa tanto en el proceso de toma de decisiones al dar órdenes de giro al motor SERVO, como en el proceso de digitalización al convertir la señal analógica en una señal digital para encontrar la frecuencia. Desde el punto de vista de la teoría de control, la medida de la salida es el valor de la frecuencia calculado luego de ejecutar la FHT, entonces el ATmega328, que realiza estas operaciones debe estar, forzosamente, en el rol de Sensor del lazo de control.

Por otro lado, sería ideal poder encontrar una función de transferencia para cada bloque del control, cuantizando el efecto de cada etapa del proceso de control. Sin embargo, en este contexto es prácticamente imposible realizar tal estudio debido a la sensibilidad variable del sistema, por ejemplo, el giro que provoca el motor en una clavija no causa un cambio de tensión lineal en la cuerda, y más aún, el cambio tensión depende de demasiados factores, entre los cuales se encuentran: nivel de tensión; densidad lineal de masa de la cuerda; largo de la cuerda; material de la cuerda; temperatura del ambiente; calibración del instrumento; tamaño de la corona de la clavija.

3.8 Interfaz de utilización

La siguiente etapa, a nivel de utilización en el proceso de diseño del dispositivo afinador es implementar una interfaz que el usuario pueda entender y mediante la cual también se le entreguen datos al dispositivo afinador. Un ejemplo de dato que el usuario debe entregar al dispositivo, es el valor de la frecuencia deseada. Un ejemplo de dato que el dispositivo debe informar al usuario, es el valor de la frecuencia medida.

3.8.1 Pantalla LCD

Una forma sencilla de mostrar información para que el usuario pueda verla, es usando una pantalla LCD para desplegar datos. Existen módulos de pantallas LCD que pueden ser controlados, al igual que el motor SERVO, por el micro-controlador ATmega328 por medio de una biblioteca informática. En esta sección se estudiará el funcionamiento y control de un módulo con pantalla LCD ADM1602K (ver anexo 7.11), este módulo se incluirá en el diseño del dispositivo afinador. El módulo LCD cuenta con un micro-controlador interno propio (de menor complejidad que el ATmega328), este micro-controlador puede recibir instrucciones desde el ATmega328 y entonces operar la pantalla LCD desplegando caracteres y controlando los circuitos internos para que se vea información coherente en la pantalla.

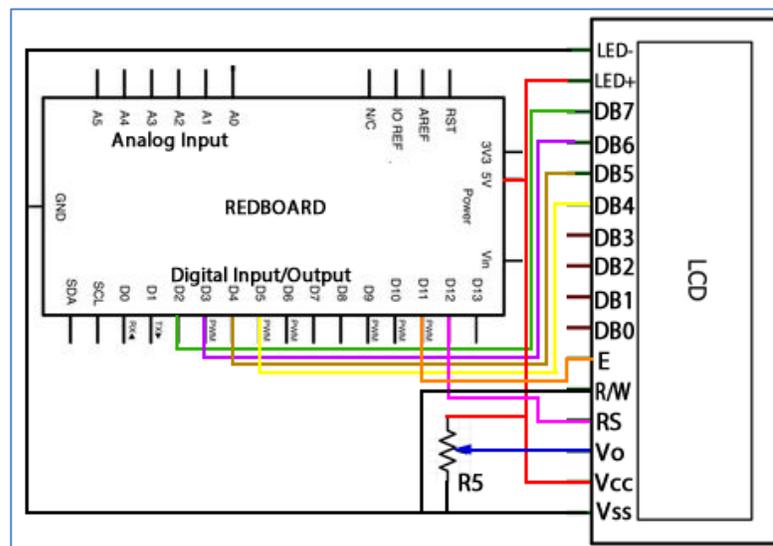


Figura 3-38: Montaje del módulo LCD al REDBOARD.

La Figura 3-38 muestra el montaje necesario para poder controlar el módulo LCD usando el ATmega328, conectado a través del REDBOARD. A primera vista parece ser un cableado de alta complejidad, pero su estudio se simplifica al aclarar que en el módulo LCD se están alimentando tanto la pantalla LCD como el micro-controlador interno que recibe las órdenes. De esta forma, los pin V_{ss} y V_{cc} del módulo LCD se usan para alimentar el micro-controlador interno, mientras que los pin LED^- y LED^+ sirven para alimentar la iluminación de la pantalla LCD. Esto explica por qué estos 4 terminales están conectados a la alimentación de 5[V] proveída por el REDBOARD y al voltaje de referencia 0[V] del

pin GROUND. Además, el pin V_0 está conectado al terminal central de un potenciómetro R_5 , esto actúa simplemente como un divisor de voltaje y al cambiar el valor de la resistencia eléctrica del potenciómetro, se hace cambiar también la intensidad luminosa de la pantalla LCD. Los pin $RS; E; DB_4; DB_5; DB_6; DB_7$. Son usados por el microcontrolador interno para recibir órdenes y enviar datos al ATmega328. El REDBOARD necesita entonces 6 terminales digitales –en este ejemplo del 2 al 6 junto con 11 y 12- libres para poder usar el módulo LCD en un montaje.

3.8.1.1 Uso de la biblioteca *LiquidCrystal*

A nivel de software y al igual que con el motor SERVO, para facilitar la programación del ATmega328 y el control del módulo LCD, existe una biblioteca informática llamada *LiquidCrystal*. Análogo a la biblioteca SERVO, para usar las funciones de *LiquidCrystal* es necesario en primera instancia declarar un objeto de tipo *LiquidCrystal*, a partir del cual se pueden invocar las funciones en el código. Las funciones disponibles en esta biblioteca son:

| Función | Descripción |
|---|---|
| <i>begin(int col, int fil)</i> | Se usa para establecer las dimensiones de la pantalla (el ATmega328 necesita contar con esta información para poder controlar el módulo LCD). Recibe dos variables, la primera indica el número de caracteres que soporta una fila de la pantalla y la segunda indica la cantidad de filas disponibles en la pantalla. |
| <i>clear()</i> | No recibe ninguna variable de entrada. Se usa para limpiar por completo la pantalla, esto es, borrar todos los caracteres que puedan estar siendo mostrados. Es recomendable usar esta función dentro de la función <i>setup()</i> del código, para evitar que existan caracteres residuales en la pantalla, producto de la ejecución de un algoritmo previo al que se esté trabajando. |
| <i>setCursor(int col, int fil)</i> | Se usa para indicar en qué posición exacta de la pantalla se desea escribir una cadena de caracteres. Es recomendable usar esta función antes de invocar <i>print()</i> para evitar que los mensajes sean escritos en posiciones erradas. Recibe dos variables, la primera indica en qué carácter de una fila específica de la pantalla se desea comenzar a escribir y la segunda indica la fila correspondiente. |
| <i>print(data)</i> | Se usa para mostrar datos en la pantalla LCD a partir de la posición indicada con la función <i>setCursor()</i> . Recibe un parámetro de entrada correspondiente a la información que será mostrada en pantalla, pueden ser letras o números. |

Tabla 3-4: Funciones de la biblioteca *LiquidCrystal*.

La Figura 3-39 muestra un código de ejemplo que hace uso de las funciones de la Tabla 3-4. Este código puede ser usado para programar el ATmega328 usando el montaje de la Figura 3-38.

```
Circuit_15 $
1 #include <LiquidCrystal.h> //Usar biblioteca LCD.
2 LiquidCrystal lcd(12,11,5,4,3,2); //Se le indica qué pins del REDBOARD se usan para controlar el módulo LCD.
3 void setup(){
4   lcd.begin(16, 2); //Establecer las dimensiones de la pantalla [2 líneas de 16 caracteres].
5   lcd.clear(); //Limpia la pantalla, en caso de que haya caracteres residuales de una ejecución anterior.
6   lcd.print("Ejemplo");
7 }
8 void loop(){
9   lcd.setCursor(0,1); //Indicar en qué caracter se desea escribir.
10  lcd.print(millis()/1000); //Millis entrega la cantidad de milisegundos que han pasado desde que se inició el algoritmo.
11 }
```

Figura 3-39: Código que usa la biblioteca LiquidCrystal.

Este código primero ordena al módulo LCD escribir el mensaje *Ejemplo* en la primera fila de la pantalla LCD y luego, en la segunda fila sobre-escribir la cantidad de segundos que han pasado desde que se inició el algoritmo. La Figura 3-40 muestra el resultado de ejecutar el código usando el montaje correspondiente.



Figura 3-40: Módulo LCD funcionando.

3.8.2 Botones

Hasta el momento, ya se cuenta con una herramienta o interfaz “de salida” para interactuar con el usuario. En efecto, la pantalla LCD puede ser usada para que el usuario se entere de datos importantes durante el proceso de afinación al usar el dispositivo propuesto en este trabajo. Sin embargo, hace falta una interfaz “de entrada” para que el dispositivo pueda interactuar con el usuario, esto es, contar con una herramienta con la que el usuario pueda ingresar datos al dispositivo.

El método de interacción de entrada más simple e intuitivo es el uso de pulsadores momentáneos o botones. Básicamente un botón es un switch que en estado de inercia se encuentra abierto y al presionarlo se encuentra cerrado. Si se arma un circuito usando botones, es posible detectar voltajes altos (botón presionado, switch cerrado) y voltajes bajos (botón suelto, switch abierto, interrupción del flujo de corriente) por medio de los terminales digitales del REDBOARD.

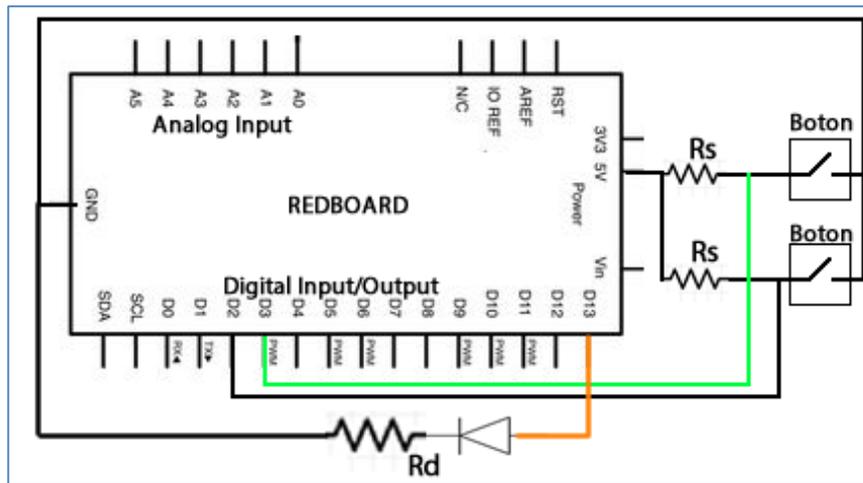


Figura 3-41: Ejemplo de montaje usando botones.

La Figura 3-41 muestra el esquema de un montaje que implementa el uso de dos botones y un LED luminoso. Si un botón es presionado, entonces se cierra el circuito que conecta la alimentación de 5[V], por medio de la resistencia R_s (para resguardo contra corto circuito), con el terminal de voltaje de referencia 0[V] GND. Luego, el pin digital correspondiente (2 o 3 en este ejemplo) del REDBOARD detectará un voltaje bajo. Si el botón no es presionado, el circuito queda abierto y el pin detectará un voltaje alto.

```

sketch_dec03a $
1 void setup() {
2   pinMode(2, INPUT); //Pin digital 2 programado como entrada.
3   pinMode(3, INPUT); //Pin digital 3 programado como entrada.
4   pinMode(13, OUTPUT); //Pin digital 13 programado como salida.
5 }
6 void loop() {
7   int Boton1, Boton2;
8   Boton1 = digitalRead(2); //Leer voltaje del pin 2
9   Boton2 = digitalRead(3); //Leer voltaje del pin3
10  boolean alguno = (Boton1 == LOW) || (Boton2 == LOW); //Primera condicion, es verdad si B1 es presionado o si B2 es presionado.
11  boolean ambos = (Boton1 == LOW) && (Boton2 == LOW); //Segunda condicion, es verdad si B1 y B2 son presionados.
12  if (alguno && !ambos) { //Si sólo un boton es presionado
13    digitalWrite(13, HIGH); // Encender el LED.
14  }
15  else {
16    digitalWrite(13, LOW); //Apagar el LED.
17  }
18 }

```

Figura 3-42: Ejemplo de código que utiliza botones.

La Figura 3-42 muestra un ejemplo de código que puede ser ejecutado usando el montaje presentado en la Figura 3-41. Al programar el ATmega328 con este algoritmo y usando el montaje indicado se debería observar que el LED luminoso se enciende sólo cuando un botón es presionado, si ningún botón es presionado o si ambos son presionados al mismo tiempo, entonces el LED se apaga.

Un tipo de botón que se comporta como switch eléctrico y que además será utilizado en el diseño del dispositivo afinador es el SMS/PSM (ver anexo 7.12).

3.8.3 Inclusión de interfaz de entrada y salida para uso del dispositivo.

Ya se cuenta con las herramientas necesarias para que un usuario del dispositivo afinador diseñado en este trabajo pueda interactuar viendo información en una pantalla o ingresando parámetros al presionar botones. En este apartado se propone un modelo de utilización que permite al usuario elegir la nota musical que debe producir la cuerda del instrumento que se desea afinar y además poder observar una lectura de la frecuencia medida.

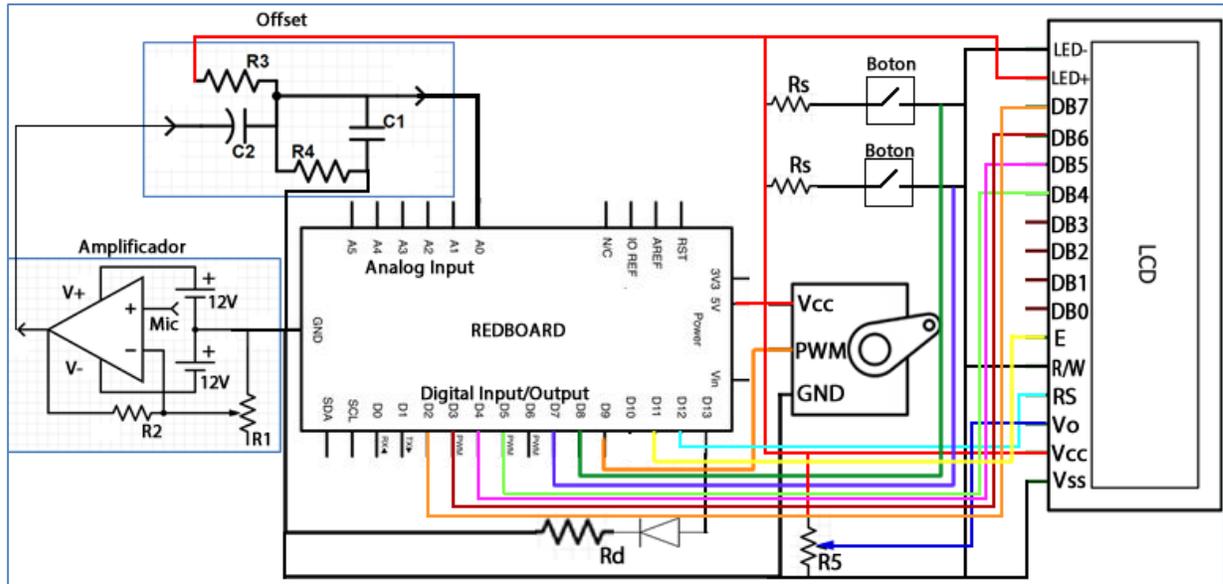


Figura 3-43: Esquema del circuito final del dispositivo afinador diseñado.

La Figura 3-43 muestra el esquema del circuito final del dispositivo afinador, incluyendo todos los elementos que se han revisado en este capítulo: Micrófono, que transforma el sonido acústico del instrumento en una señal eléctrica; Amplificador, que trata la señal analógica dejándola con una amplitud adecuada; Sumador de componente continua, que traslada el nivel en torno al cual oscila la señal; REDBOARD, que contiene el micro-controlador que digitaliza la señal y controla otros elementos del circuito; motor SERVO, que transmite torque a la clavija del instrumento musical que se desea afinar; módulo LCD, que permite mostrar información al usuario; y botones, que permiten ingresar información al usuario. Los valores de las nuevas resistencias usadas en el diseño definitivo son $R_s = 10[k\Omega]$ y R_5 es un potenciómetro de $10[k\Omega]$.

La última etapa es programar el comportamiento de los botones y la pantalla LCD, para lograr este cometido se han programado varias funciones informáticas, esto es el equivalente a crear una biblioteca similar a las estudiadas en este capítulo. En la Tabla 3-5 se presentan las funciones implementadas en el entorno de programación Arduino.

| Función | Descripción |
|---|---|
| SetSampleRate() | Sirve para ajustar la tasa de muestreo con que se está digitalizando la señal que llega a la entrada analógica del REDBOARD. No recibe ningún parámetro de entrada porque trabaja con las variables globales <i>nota</i> y <i>octava</i> , que indican la nota musical (y por lo tanto su frecuencia) que debe producir la cuerda que se está afinando. Lo que hace esta función es ajustar la frecuencia con que se ejecuta el <i>timer interrupt</i> (ver sección 3.4.3) del reloj <i>timer0</i> . Esta frecuencia queda ajustada a un valor conveniente: lo suficientemente bajo para tener una resolución de bins de la Transformada de Hartley (ver secciones 2.6 y 2.7) suficiente para discriminar la nota específica y también lo suficientemente alto para lograr captar la frecuencia deseada (ver sección 2.8). Esta función es invocada por la función <i>DisplayNota()</i> . |
| DisplayNota() | Esta función es ejecutada cada vez que el usuario presiona un botón (detalles más adelante). Sirve para mostrar en pantalla el nombre de la nota musical objetivo, es decir, la nota que debería producir la cuerda que se está afinando junto con la su frecuencia característica. Para calcular tal frecuencia, esta función invoca a <i>freq()</i> y hace uso de las variables globales <i>nota</i> y <i>octava</i> . No recibe parámetros de entrada. Esta función invoca a <i>SetSampleRate()</i> , es decir cada vez que el usuario presiona un botón se ajusta también la tasa de muestreo. |
| freq(int n, int o) | Ver Figura 3-34. |
| Escala(double x, double a, double b) | Realiza una transformación lineal de una variable desde una escala específica hacia la escala definida por el intervalo (0,180). Esta función recibe 3 parámetros de entrada: La primera <i>x</i> es el número que se desea transformar; la segunda y tercera variables <i>a, b</i> definen un intervalo representando la escala original de la primera variable. Retorna un valor entero correspondiente al número transformado en la nueva escala. Esta función básicamente es la implementación informática de la fórmula $180 \times \frac{(x-a)}{(b-a)}$. Se usa para traducir el logaritmo de la frecuencia medida en un número adecuado para usar la función <i>write()</i> de la biblioteca SERVO. |

Tabla 3-5: Funciones implementadas.

La Figura 3-44 muestra una porción de código programada dentro de la función *loop()* que define las acciones gatilladas al presionar los botones incluidos en el dispositivo afinador. El comportamiento es simple: En la pantalla se está mostrando el nombre de la nota deseada junto con el número de la octava a la que pertenece, al pulsar un botón se incrementa la nota en un proceso cíclico, esto es, si se alcanzó la nota más alta (en general, *Si*) entonces la búsqueda regresa a la nota más grave (en genera, *Do*). Lo mismo ocurre al presionar el segundo botón, pero modificando la octava en que la se está trabajando.

En la sección 2.3 se estudió que hay 10 octavas en el espectro audible, sin embargo muchas de estas frecuencias simplemente no son producidas por los instrumentos musicales de interés de este trabajo. El rango posible de notas musicales deseadas entonces va desde (ver Tabla 2-1) un Mi_1 , que es la nota más baja que puede producir un bajo eléctrico o un contrabajo, hasta un La_5 , que es la nota que produce la cuerda más aguda de una bandurria.

El dispositivo puede afinar cualquier cuerda que deba producir una nota musical dentro de este rango de frecuencias, pero es necesario que el usuario indique qué nota musical se está afinando. En otras palabras, se requiere que el usuario tenga los conocimientos básicos sobre el instrumento musical que está usando y sepa qué nota musical le corresponde a cada cuerda.

```

Memoria
29 void loop(){
30   B1Actual = digitalRead(7); //Leer el estado del boton1. LOW es presionado, HIGH es sin presionar.
31   if(B1Actual!=B1Anterior){ //Si cambió su estado desde la ultima lectura.
32     if(B1Actual==HIGH){ //El cambio fue que se soltó el botón.
33       nota=nota+1; //Aumenta el valor de la nota deseada.
34       if(nota>12){ //Si supero un SI, entonces regresa al D0.
35         nota=1;
36         DisplayNota(); //Muestra la nota deseada en pantalla.
37       }
38       B1Anterior=B1Actual; //Actualiza el ultimo estado del boton1 registrado.
39     }
40   B2Actual = digitalRead(8); //Leer el estado del boton2.
41   if(B2Actual!=B2Anterior){ //Si cambió su estado desde la ultima lectura.
42     if(B2Actual==HIGH){ //El cambio fue que se soltó el botón.
43       octava=octava+1; //Aumenta el valor de la octava de nota deseada.
44       if(octava>5){ //Si se superó la quinta octava, volver a la primera.
45         octava=1;
46         DisplayNota(); //Muestra la nota deseada en pantalla.
47       }
48       B2Anterior=B2Actual; //Actualiza el ultimo estado del boton2 registrado.
49     }
50   }
51   if(!lleno){ //Si el vector fbt_input[] está listo para ser tratado

```

Figura 3-44: Comportamiento de los botones.

Adicionalmente, se observa que cada vez que un botón es presionado, es decir, cada vez que la nota deseada es ajustada por el usuario, se invoca a la función *DisplayNota()*, la que a su vez gatilla la ejecución de la función *SetSampleRate()*. Esto provoca que la frecuencia de muestreo cambie según la nota musical que se desea afinar, asegurando que no se pierda resolución para discriminar distintas notas, aprovechando al máximo las limitadas 256 muestras disponibles para ejecutar la Transformada de Hartley.

El nombre de la nota musical deseada y la octava en que se encuentra aparecen en el extremo superior izquierdo de la pantalla. Bajo esta indicación –en el extremo inferior izquierdo de la pantalla- se muestra la frecuencia de la nota objetivo. El usuario puede leer el valor de la frecuencia medida en el extremo superior derecho de la pantalla. Y en extremo inferior derecho de la pantalla, se observa la tasa de muestreo activada, que cambia según la nota musical deseada.

El código completo utilizado para programar el ATmega328 en el funcionamiento del afinador electromecánico para instrumentos de cuerda con sistema sinfín-corona se puede ver en el anexo 7.17. Con esto termina el diseño final del dispositivo y en el apartado siguiente se discutirán resultados y distintos comportamientos al hacer uso del afinador electro-mecánico.

4 RESULTADOS

En este apartado se expone el desempeño del afinador electro-mecánico diseñado luego de realizar pruebas en 5 instrumentos distintos, luego se discuten las posibles razones que explican el comportamiento observado y se proponen soluciones para los problemas detectados.

4.1 Pruebas realizadas

Uno de los objetivos específicos de este proyecto plantea que el afinador electro-mecánico debe ser versátil y compatible con la mayor cantidad de instrumentos posibles. Por esta razón se eligieron 5 ejemplares distintos, que en conjunto cubren todo el espectro de frecuencias involucradas en las notas musicales que pueden producir los instrumentos de cuerda con sistema sinfín-corona, esto es, desde un Mi_1 , que oscila a $41.2[Hz]$ hasta un La_5 , que vibra a $880[Hz]$.

4.1.1 Afinación de una guitarra

La guitarra es un instrumento de 6 cuerdas que tiene su completo registro dentro de las 4 octavas que es capaz de afinar el dispositivo diseñado. A continuación se presenta una tabla con datos obtenidos al afinar una guitarra acústica con un encordado de nylon y bronce (ver anexo 7.13). Las tres primeras cuerdas del instrumento usado son de nylon puro y las últimas tres están fabricadas con un núcleo de nylon recubierto por un entorchado (bobinado) de bronce.

| Cuerda | Nota | Frecuencia [Hz] | Material | Δf [Hz] | Tiempo [s] | Precisión [Hz] | Precisión [¢] |
|--------|---------|--------------------|----------------|-----------------|---------------|-------------------|------------------|
| 1 | Mi_4 | 329.63 | Nylon | 3.1289 | 7.52 | 0.47 | 2.4667 |
| 2 | Si_3 | 246.94 | Nylon | 2.3437 | 8.2 | -2.02 | -14.2199 |
| 3 | Sol_3 | 196 | Nylon | 1.9531 | 5.23 | 0.29 | 2.5597 |
| 4 | Re_3 | 146.83 | Nylon y bronce | 1.5625 | 6.02 | 0.83 | 9.7586 |
| 5 | La_2 | 110 | Nylon y bronce | 1.5625 | 8.28 | 0.16 | 2.5163 |
| 6 | Mi_2 | 82.41 | Nylon y bronce | 1.1718 | 9.56 | 0.21 | 4.4061 |

Tabla 4-1: Resultados al afinar una guitarra.

Los datos de las 4 primeras columnas de la Tabla 4-1 son característicos del instrumento que se está afinando. La quinta columna contiene un valor característico del diseño del dispositivo: la resolución en frecuencia, o sea, la mínima diferencia de frecuencias que puede detectar al afinar la correspondiente cuerda. La quinta columna contiene el tiempo que tardó el dispositivo en afinar la correspondiente cuerda. La sexta columna muestra la diferencia entre la frecuencia final obtenida y la frecuencia deseada. La octava columna indica la precisión alcanzada en cents (ver ecuación (2-6)). Las últimas 3 columnas son promedios calculados al realizar 10 mediciones usando el mismo instrumento musical.

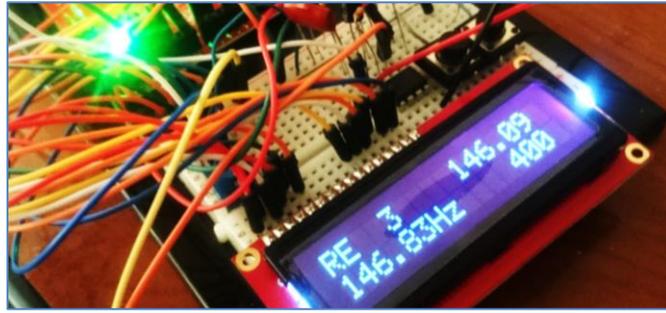


Figura 4-1: Afinador electro-mecánico afinando cuarta cuerda de guitarra.

4.1.2 Afinación de un laúd español

El Laúd español es un instrumento de 12 cuerdas afinadas en pares, también se dice que es un instrumento de “seis cuerdas dobles” o de 12 cuerdas con seis ordenanzas [22]. Esto significa que dos cuerdas continuas producen la misma nota musical. Los dos primeros pares de cuerdas de este instrumento son de acero, mientras que el resto está construido usando un núcleo de acero, recubierto con un bobinado de plata (ver anexo 7.14).

| Cuerda | Nota | Frecuencia [Hz] | Material | Δf [Hz] | Tiempo [s] | Precisión [Hz] | Precisión [¢] |
|--------|-----------|-----------------|---------------|-----------------|------------|----------------|---------------|
| 1a | La_4 | 440 | Acero | 3.9062 | 10.12 | -0.55 | -2.1654 |
| 1b | La_4 | 440 | Acero | 3.9062 | 11.23 | -0.55 | -2.1654 |
| 2a | Mi_4 | 329.63 | Acero | 3.1289 | 8.65 | 0.47 | 2.4667 |
| 2b | Mi_4 | 329.63 | Acero | 3.1289 | 5.02 | 3.6 | 18.8050 |
| 3a | Si_3 | 246.94 | Acero y plata | 2.3437 | 4.28 | 0.33 | 2.3120 |
| 3b | Si_3 | 246.94 | Acero y plata | 2.3437 | 3.56 | 0.33 | 2.3120 |
| 4a | $Fa\#_3$ | 185 | Acero y plata | 1.9531 | 12.63 | 1.52 | 14.1663 |
| 4b | $Fa\#_3$ | 185 | Acero y plata | 1.9531 | 15.45 | -0.43 | -4.0287 |
| 5a | $Do\#_3$ | 138.59 | Acero y plata | 1.5625 | - | - | - |
| 5b | $Do\#_3$ | 138.59 | Acero y plata | 1.5625 | - | - | - |
| 6a | $Sol\#_2$ | 103.83 | Acero y plata | 1.5625 | - | - | - |
| 6b | $Sol\#_2$ | 103.83 | Acero y plata | 1.5625 | - | - | - |

Tabla 4-2: Resultados al afinar un laúd español.

La Tabla 4-2 muestra los resultados obtenidos al afinar este instrumento. Los valores en blanco representan medidas en que el dispositivo detectaba frecuencias erróneas o distintas a la vibración fundamental de la cuerda. Más adelante se discutirá este comportamiento.

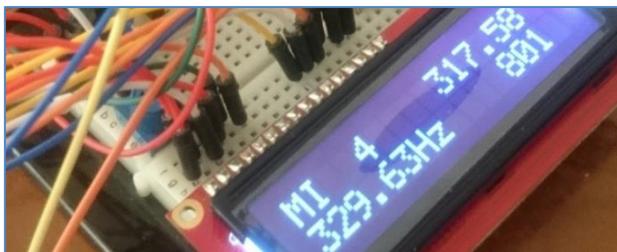


Figura 4-2: Dispositivo afinando segundo par de cuerdas de laúd.



Figura 4-3: Dispositivo ajustando tensión de las cuerdas del laúd.

4.1.3 Afinación de una bandurria

La bandurria es otro instrumento de seis cuerdas dobles. Su afinación es análoga a la del laúd español, con la diferencia que cada cuerda vibra al doble de la frecuencia original, esto es, las notas están en una octava superior [22], alcanzando a producir las notas musicales más más agudas existentes dentro del espectro que abarcan los instrumentos de cuerda con sistema sinfín-corona. Al igual que el laúd español, los dos primeros pares de cuerdas son de acero y el resto están construidas con un núcleo de acero y un entorchado de plata (ver anexo 7.15).

| Cuerda | Nota | Frecuencia [Hz] | Material | Δf [Hz] | Tiempo [s] | Precisión [Hz] | Precisión [¢] |
|--------|-----------|-----------------|---------------|-----------------|------------|----------------|---------------|
| 1a | La_5 | 880 | Acero | 7.0742 | 8.62 | 0.74 | 1.4552 |
| 1b | La_5 | 880 | Acero | 7.0742 | 5.23 | 0.74 | 1.4552 |
| 2a | Mi_5 | 659.26 | Acero | 5.4843 | 8.65 | 1.61 | 4.2228 |
| 2b | Mi_5 | 659.26 | Acero | 5.4843 | 6.57 | 1.61 | 4.2228 |
| 3a | Si_4 | 493.88 | Acero y plata | 3.9063 | 5.25 | 0.26 | 0.9112 |
| 3b | Si_4 | 493.88 | Acero y plata | 3.9063 | 5.55 | 4.17 | 14.5559 |
| 4a | $Fa\#_4$ | 369.99 | Acero y plata | 3.1289 | 3.25 | 0.79 | 3.6925 |
| 4b | $Fa\#_4$ | 369.99 | Acero y plata | 3.1289 | 4.25 | 0.79 | 3.6925 |
| 5a | $Do\#_4$ | 277.18 | Acero y plata | 2.3438 | 12.52 | -1.79 | -11.2163 |
| 5b | $Do\#_4$ | 277.18 | Acero y plata | 2.3438 | 15.69 | -4.13 | -25.9894 |
| 6a | $Sol\#_3$ | 207.65 | Acero y plata | 1.9531 | 12.25 | -1.6 | -13.3911 |
| 6b | $Sol\#_3$ | 207.65 | Acero y plata | 1.9531 | 13.52 | 0.36 | -2.9988 |

Tabla 4-3: Resultados al afinar una bandurria.

Para este instrumento se registraron tiempos de afinación notablemente elevados con respecto al resto, en sus últimos dos pares de cuerdas. La explicación para este fenómeno se cubrirá en la sección 4.2.3.

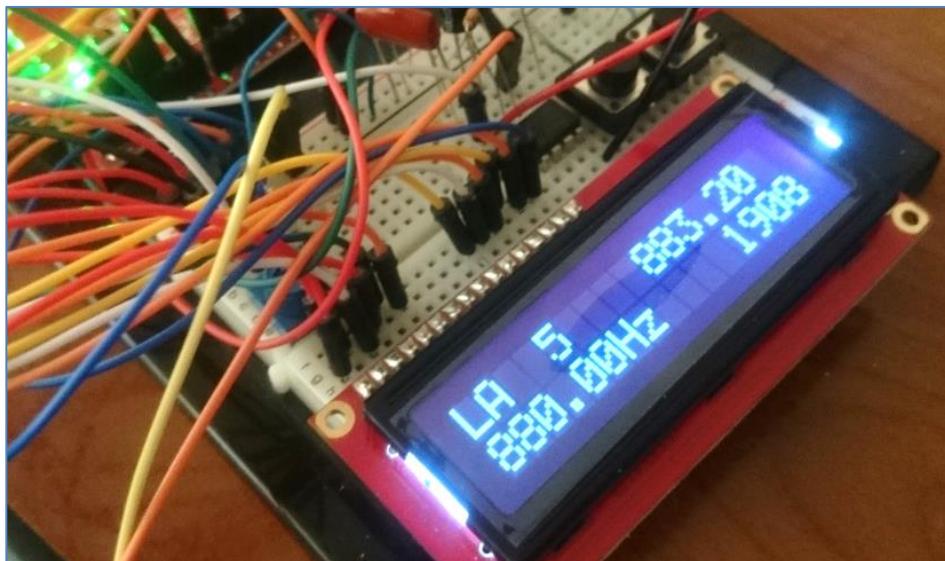


Figura 4-4: Afinador electro-mecánico detectando la nota más alta que es capaz de discriminar.

4.1.4 Afinación de un laudón

El laudón es un instrumento que originalmente fue diseñado con 6 ordenanzas dobles más una simple [22], pero el ejemplar que se usó para realizar las pruebas en este trabajo es una restauración del maestro lutier Antonio Zurita y ha sido modificado para ser encordado usando siete cuerdas dobles (ver Figura 4-5). Este instrumento fue encordado usando cuerdas de acero para sus dos primeras ordenanzas y el resto de las cuerdas tiene núcleo de acero y entorchado de níquel. Como no existen encordados para este tipo de instrumento en el mercado, se usó una combinación de cuerdas de guitarra eléctrica (primeros cinco pares de cuerdas) con cuerdas de bajo eléctrico (últimas dos ordenanzas).

| Cuerda | Nota | Frecuencia [Hz] | Material | Δf [Hz] | Tiempo [s] | Precisión [Hz] | Precisión [¢] |
|--------|----------|-----------------|----------------|-----------------|------------|----------------|---------------|
| 1a | Sol_4 | 392 | Acero | 3.12 | 3.25 | -2.44 | -10.8340 |
| 1b | Sol_4 | 392 | Acero | 3.12 | 2.18 | 0.68 | 3.0207 |
| 2a | Re_4 | 293.66 | Acero | 2.34 | 5.28 | 0.47 | 2.7994 |
| 2b | Re_4 | 293.66 | Acero | 2.34 | 6.14 | 0.47 | 2.7994 |
| 3a | La_3 | 220 | Acero y níquel | 1.95 | 8.54 | 1.68 | 13.1701 |
| 3b | La_3 | 220 | Acero y níquel | 1.95 | 10.21 | -0.27 | -2.1260 |
| 4a | Mi_3 | 164.81 | Acero y níquel | 1.56 | 3.5 | 0.02 | 0.2754 |
| 4b | Mi_3 | 164.81 | Acero y níquel | 1.56 | 4.2 | -1.5 | -16.1865 |
| 5a | Si_2 | 123.47 | Acero y níquel | 1.17 | 15.28 | 1.16 | 2.2304 |
| 5b | Si_2 | 123.47 | Acero y níquel | 1.17 | 12.32 | 1.16 | 2.2304 |
| 6a | $Fa\#_2$ | 92.5 | Acero y níquel | 0.78 | - | - | - |
| 6b | $Fa\#_2$ | 92.5 | Acero y níquel | 0.78 | - | - | - |
| 7a | Do_2 | 65.41 | Acero y níquel | 0.78 | 20.23 | -0.17 | -4.6752 |
| 7b | Do_2 | 65.41 | Acero y níquel | 0.78 | 18.18 | 0.61 | 16.1658 |

Tabla 4-4: Resultados al afinar un laudón.



Figura 4-5: Laudón restaurado para usar 14 cuerdas dobles.

4.1.5 Afinación de un bajo eléctrico.

El bajo eléctrico es un instrumento de cuatro cuerdas que se afina igual que un contrabajo y alcanza a producir las notas musicales más graves dentro del espectro que abarcan los instrumentos de cuerda con sistema sinfín-corona¹.

| Cuerda | Nota | Frecuencia [Hz] | Material | Δf [Hz] | Tiempo [s] | Precisión [Hz] | Precisión [¢] |
|--------|---------|-----------------|----------------|-----------------|------------|----------------|---------------|
| 1 | Sol_2 | 92 | Acero y níquel | 0.78 | * | 0.05 | 0.902 |
| 2 | Re_2 | 73.42 | Acero y níquel | 0.78 | * | 0.41 | 9.7306 |
| 3 | La_1 | 55 | Acero y níquel | 0.78 | * | 0.86 | 26.8607 |
| 4 | Mi_1 | 41.2 | Acero y níquel | 0.39 | 15.82 | 0.86 | 17.9736 |

Tabla 4-5: Resultados al afinar un bajo eléctrico.

La Tabla 4-5 muestra los resultados obtenidos al afinar este instrumento. Los valores con * representan medidas en que el dispositivo detectaba correctamente, y daba el giro correcto, pero el motor no producía el torque suficiente para hacer girar la clavija y se llegó a la nota deseada ajustándola manualmente.

4.2 Discusión de resultados.

A continuación se realiza un análisis de los resultados favorables junto con los casos en que el afinador no tuvo un buen desempeño, explicando las posibles causas y proponiendo soluciones que no fueron implementadas en este proyecto.

4.2.1 Precisión

Se observa que en todos los resultados obtenidos, siempre el valor de Δf es más grande que la diferencia en frecuencia entre el valor obtenido y el valor deseado. Esto significa que el ancho de banda que cubre cada bin de la Transformada de Hartley es superior al error en frecuencia obtenido, en otras palabras, el dispositivo está siempre discriminando las notas a su máxima precisión posible. El valor de Δf depende del número de muestras utilizadas, que en el caso del diseño propuesto es siempre igual a 256 debido a las características del micro-procesador ATmega328, que sólo tiene $2[kB]$ ver apartado 3.4.1.

En términos de precisión y considerando las limitantes de memoria RAM disponible, el nivel de detección es un éxito, ya que no hay ningún error superior a la resolución que alcanza el dispositivo. Además, la Figura 4-6 muestra que en ninguna de las mediciones realizadas se alcanza un error absoluto en frecuencia superior a los $44.5[Hz]$.

¹ Existen bajos eléctricos particulares capaces de producir notas más graves, incorporan una quinta cuerda que vibra a $30.86[Hz]$, esto es un Si_0 . Este tipo de instrumentos son llamados “de registro extendido” y escapan los alcances de este proyecto.

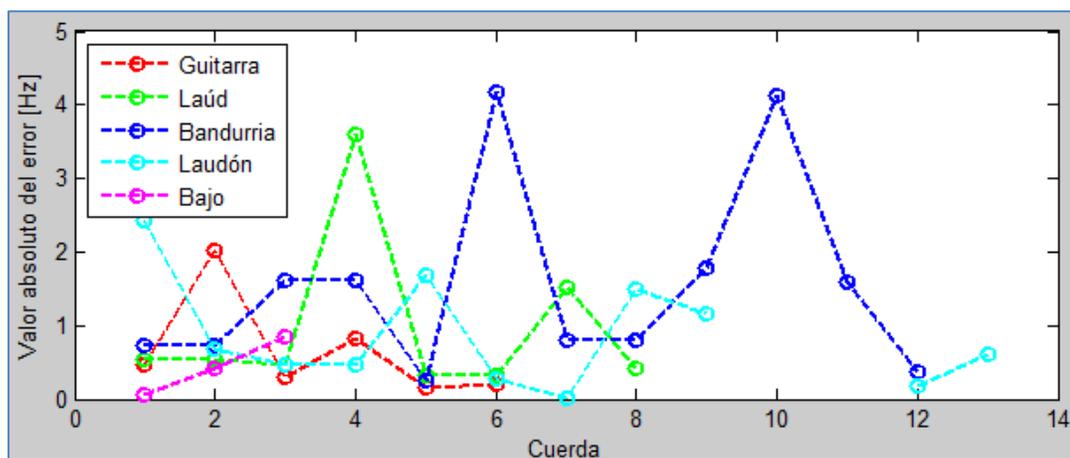


Figura 4-6: Errores absolutos en frecuencia.

Por otro lado, usando la escala de igual temperamento para todas las notas musicales, existen varios estudios y es muy difícil establecer un umbral en cents estableciendo un límite perceptible por el oído humano, pero un número estandarizado que se usa como referencia [5] es de 20[¢], esto equivale a un décimo de tono. Naturalmente pueden haber personas con oído más sensible, pero usando tal límite y observando los errores absolutos obtenidos, se puede apreciar en la Figura 4-7 que hay un solo valor que supera los 20[¢] de error.

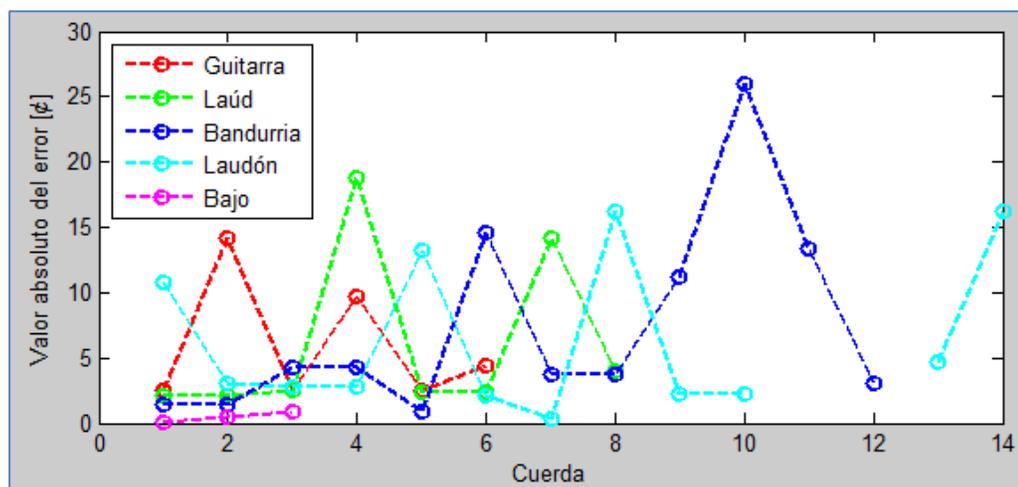


Figura 4-7: Errores absolutos en escala de igual temperamento para todas las notas musicales.

4.2.2 Número de muestras

En el diseño propuesto, se estudió que la cantidad de muestras capturadas para calcular la Transformada de Hartley y luego identificar la frecuencia más significativa de la señal es muy limitada. En todos los casos, por razones de memoria informática disponible se usan sólo 256 muestras.

La ecuación (2-8) indica que una de las variables para alcanzar mejor resolución es la tasa de muestreo f_s , en efecto, es ese el parámetro que se hace variar al seleccionar una nota musical deseada distinta. Sin embargo, pero si se usara un micro-controlador con mayor capacidad al ATmega328 se podrían usar más muestras para calcular la Transformada, lo que se podría traducir en tasas de muestreo más altas sin tener que sacrificar resolución.

Además, una reconstrucción de una señal con mayor número de muestras es una representación más fiel del sonido a partir del cual fue generada. La Figura 4-8 muestra el gráfico de la densidad espectral de una señal reconstruida usando 256 muestras, mientras que la Figura 4-9 muestra la densidad espectral de la misma señal, esta vez calculada usando 4096 muestras. Se observa que la frecuencia más importante detectada es distinta para los dos casos. Este fenómeno podría explicar también los datos en blanco de los apartados de la sección 4.1, en que el dispositivo obtenía lecturas erradas.

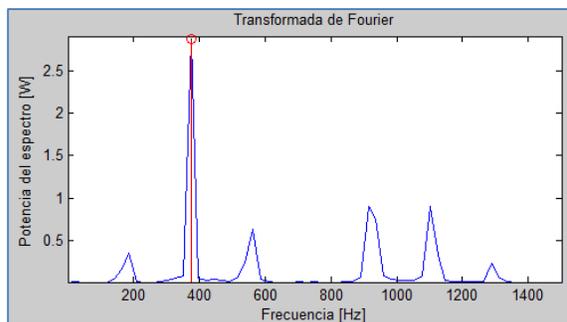


Figura 4-8: Densidad espectral usando 256 muestras.

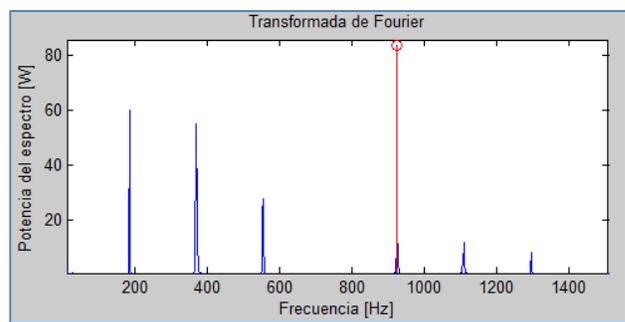


Figura 4-9: Densidad espectral usando 4096 muestras.

4.2.3 Intensidad de giro

Las tablas de resultados expuestas en los apartados de la sección 4.1 presentan una tendencia a que las ordenanzas más graves de los instrumentos tarden más tiempo en ser afinadas.

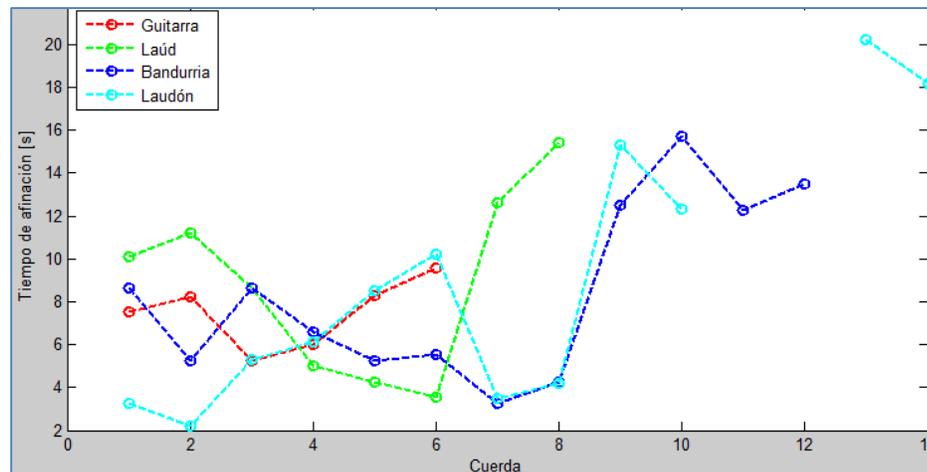


Figura 4-10: Tiempos de afinación de cada cuerda.

La Figura 4-10 muestra un gráfico en que se puede apreciar esta tendencia. La explicación para este fenómeno es simple: las cuerdas de calibre más grueso o últimas ordenanzas, en general para cualquier instrumento de cuerdas, son mucho más sensibles al giro de la clavija, o sea, un pequeño giro se traduce en una gran variación de tensión y por lo tanto una gran variación en la frecuencia emitida.

Esto provoca que cuando el motor gira la clavija, la nota producida por la cuerda varía más allá del objetivo deseado, entonces el motor debe comenzar a girar en sentido contrario para llegar a la nota musical deseada, esto ocurre varias veces y en ambos sentidos, lo que significa un mayor tiempo de afinación para este tipo de cuerdas.

La ecuación (2-4) relaciona la tensión de la cuerda con la frecuencia de vibración, pero no se ha estudiado la relación existente entre el giro del tornillo sinfín de la clavija y la variación de tensión en la cuerda. Esta materia se aleja de los alcances del proyecto, porque la relación existente es distinta para cada instrumento en particular (por ejemplo, dos bandurrias pueden tener relaciones de giro y tensión distintas), y además también es distinta para cada cuerda de un mismo instrumento.

Es posible hacer un estudio para un instrumento en particular, midiendo sus tensiones y el torque que realizan sus clavijas, para incorporar estos datos al diseño del dispositivo afinado, pero entonces se estaría sacrificando la versatilidad del mismo. En ese contexto se estaría diseñando un afinador dedicado específicamente para el instrumento estudiado, y entonces ya no se estaría cumpliendo el objetivo específico (ver apartado 1.3) que plantea la compatibilidad del afinador con la mayor cantidad de instrumentos posibles.

Otra alternativa para corregir el problema del giro exagerado sería utilizar un motor SERVO distinto que se pueda controlar con mayor precisión. Esto significaría contar con un mayor presupuesto para el desarrollo del proyecto. El MG995 utilizado en el diseño propuesto tiene un costo del orden de los \$12.000.- mientras que otros motores SERVO capaces de realizar mayor torque y que reciben órdenes con parámetros más finos (generalmente usados en aeromodelismo) pueden llegar a costar \$80.000.-

4.2.4 Tiempo de afinación

Se mencionó en la sección anterior que existen casos particulares en que el tiempo de afinación se extiende debido al torque excesivo aplicado por el motor SERVO sobre la clavija. Además, la respuesta de variación de tono no tiene la misma forma para todos los instrumentos ni para todas las cuerdas de un mismo instrumento. A continuación se presentan gráficos mostrando la evolución en el tiempo del proceso de afinación de algunos de los casos estudiados en el apartado 4.1.

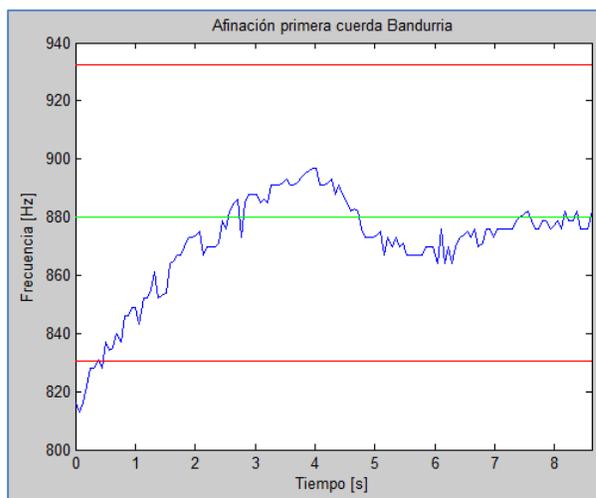


Figura 4-11: Tiempo de afinación bandurria.

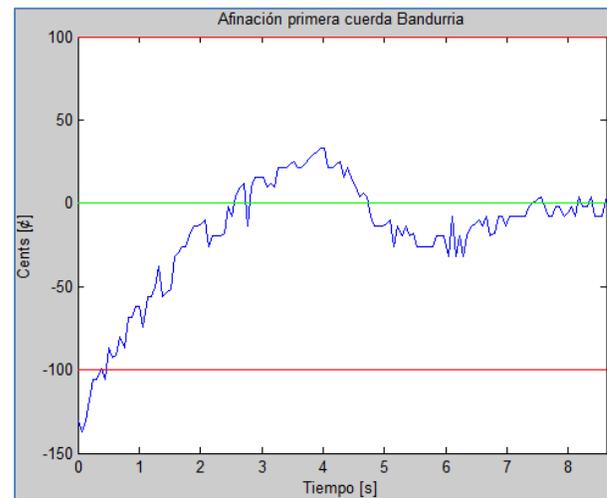


Figura 4-12: Tiempo de afinación bandurria.

La Figura 4-11 muestra un gráfico de frecuencia en el tiempo, evidenciando la evolución del tono que produce la primera cuerda de una bandurria mientras es afinada con el dispositivo. En verde se ha marcado la frecuencia deseada, que en este caso corresponde a $880[Hz]$ producidos por un La_5 y en rojo se han marcado las frecuencias que emiten las notas adyacentes al tono deseado, esto es un $La\#_5$ en la parte superior y un $Sol\#_5$ por debajo de la curva. La Figura 4-12 muestra exactamente la misma información, pero en lugar de identificar la frecuencia producida por la cuerda en el tiempo, se mide en cents la diferencia entre la altura emitida y la altura deseada. Se observa en ambas imágenes, que el tono deseado fue alcanzado alrededor de los 3 segundos, pero luego la cuerda alcanzó una frecuencia de vibración superior a la deseada (hubo torque en exceso), luego la cuerda fue aflojada y nuevamente la intensidad del giro de la clavija sobrepasó el objetivo, para finalmente converger a la altura deseada.

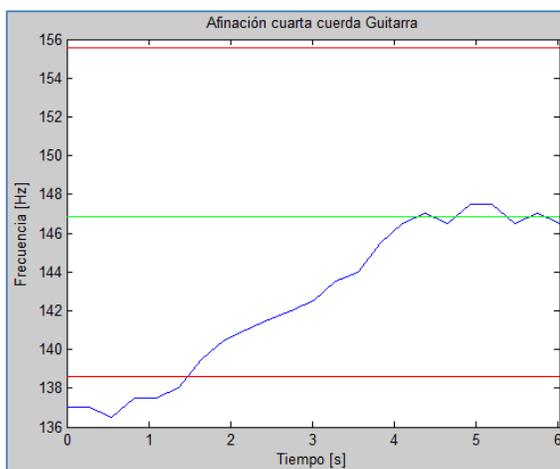


Figura 4-13: Tiempo de afinación guitarra.

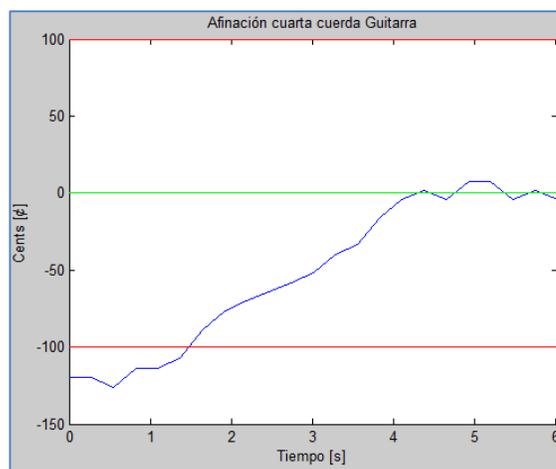


Figura 4-14: Tiempo de afinación guitarra.

La Figura 4-13 y la Figura 4-14 muestran de la misma forma, la evolución del tono producido por una cuarta cuerda de guitarra al ser afinada. Se observa que en este caso la curva es mucho más suave, esto se debe a que las lecturas y cálculos de frecuencia en el tiempo son más consistentes y tienen cambios menos bruscos, tal fenómeno evidencia que una cuarta cuerda de guitarra es mucho menos sensible al giro de la clavija (y por lo tanto más fácil de afinar) que una primera cuerda de bandurria. Se observa en estos gráficos, que la frecuencia objetivo fue alcanzada casi de forma lineal y no fue necesario girar en el sentido opuesto pues el tono producido no se alejó hacia una nota más aguda que la nota desea, en este caso un Re_3 .

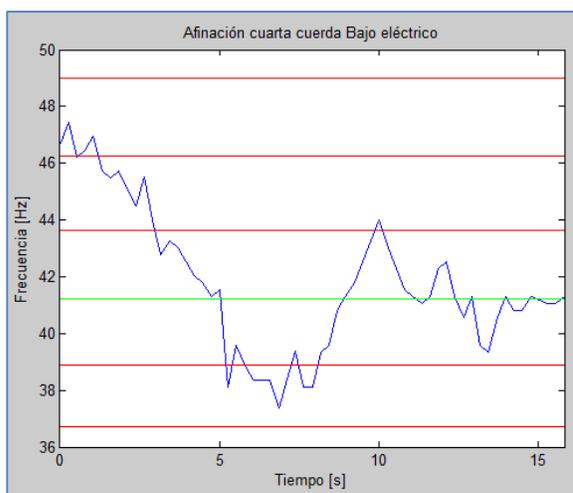


Figura 4-15: Tiempo de afinación bajo eléctrico.

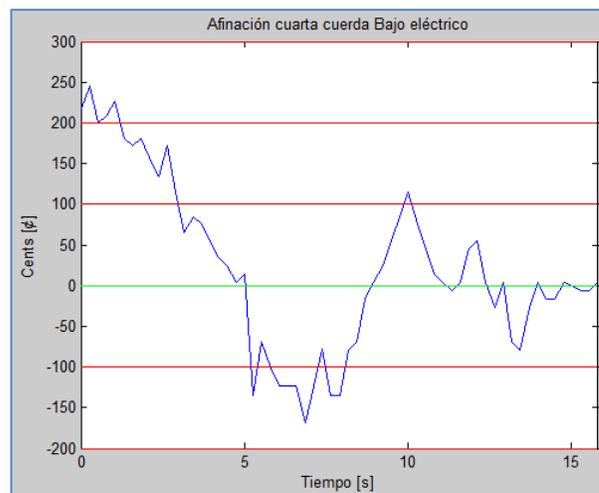


Figura 4-16: Tiempo de afinación bajo eléctrico.

La Figura 4-15 y la Figura 4-16 muestran el comportamiento en el tiempo al afinar una cuarta cuerda de bajo eléctrico. En este caso se han marcado en rojo los intervalos correspondientes hasta dos notas musicales más graves y tres notas más agudas. Al igual que con la bandurria, se sobre pasa el objetivo (esta vez, tres veces) antes de

converger hacia la nota musical deseada. Esta cuerda es mucho más sensible al giro de la clavija que los dos casos anteriores.

4.2.5 Torque insuficiente

Un comportamiento que limitó las mediciones en las pruebas realizadas con bajo eléctrico es que el motor SERVO no tenía la fuerza suficiente para hacer girar la clavija. Esto depende mucho de la tensión particular de las cuerdas que se esté afinando y de la calidad de las clavijas mismas. El ejemplar de bajo eléctrico que se usó para estas pruebas tiene unas clavijas ligeramente más grandes que la interfaz mecánica tomada de la bobinadora para adaptar el motor SERVO (ver Figura 4-17). Sin embargo, la interfaz mecánica cuenta con una rendija que le permite ejercer torque sobre este tipo de clavijas (ver Figura 4-18).



Figura 4-17: Tamaño de clavija de bajo eléctrico e interfaz mecánica del motor.

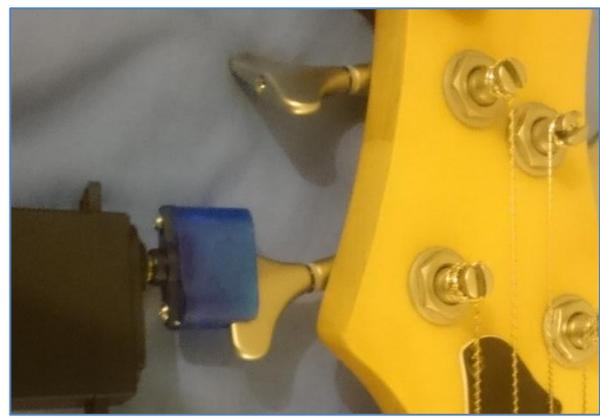


Figura 4-18: Motor ejerciendo torque sobre clavija de bajo eléctrico.

El motor MG995 utilizado en el diseño del dispositivo es capaz de realizar un torque de levante de $8.5[kg\ cm]$, eso significa que el motor es capaz de levantar un peso de $8.5[kg]$ a una distancia de $1\ [cm]$ desde su eje de rotación. La clavija del bajo eléctrico tiene un ancho total de $3.4[cm]$, en otras palabras, el motor no puede ejercer una fuerza mayor a $\frac{8.5[kgcm]}{\frac{3.4}{2}[cm]} = 5[kg]$ al realizar el torque sobre esta clavija. Según los resultados observados, esto no es suficiente para hacer girar el tornillo sinfín del bajo eléctrico. Nuevamente, la solución para corregir esta situación es cambiar el motor usado en el diseño por un SERVO que pueda ejercer mayor torque y no se implementó en el diseño actual porque se trata de un proyecto que no cuenta con respaldo económico más que el individual.

4.2.6 Modos normales y armónicos

En la sección 2.1.3 se mencionó que el timbre característico de un sonido se configura por la combinación de las frecuencias armónicas que acompañan a la fundamental de este. Es posible entonces, que la señal que esté detectando el afinador no tenga su mayor potencia espectral concentrada en la frecuencia fundamental de la

vibración de la cuerda, sino que en algún armónico relativo o en algún modo normal característico del material con que está construida la cuerda.

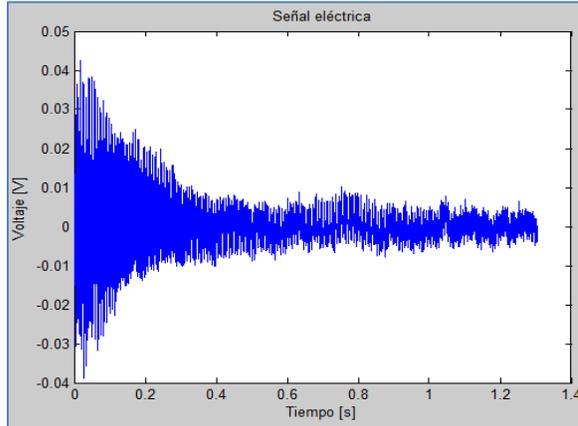


Figura 4-19: Captura de un $Fa\#_3$ producido por un laúd.

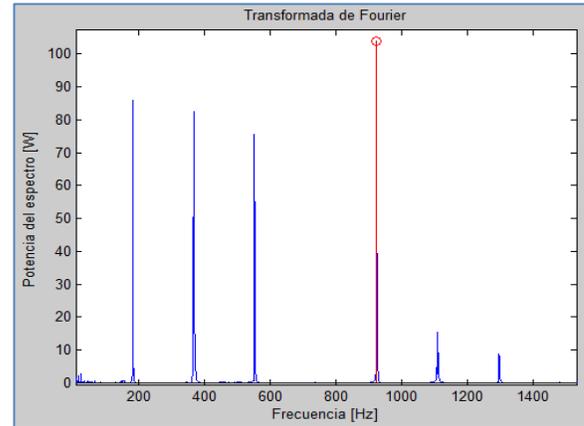


Figura 4-20: Densidad espectral de un $Fa\#$ producido por un laúd.

La Figura 4-19 muestra la captura de un $Fa\#$ producido por la cuarta ordenanza de un laúd español y la Figura 4-20 muestra su densidad espectral de frecuencias. La nota musical tiene una frecuencia característica de $185[Hz]$, pero en este caso, vemos que existen 6 picos de concentración de potencia espectral: $f_1 = 185[Hz]$; $f_2 = 370[Hz]$; $f_3 = 554.4[Hz]$; $f_4 = 929.9[Hz]$; $f_5 = 1108.7[Hz]$; $f_6 = 1296[Hz]$.

La combinación de todas estas frecuencias juntas, determinan el timbre característico del instrumento al reproducir esa nota. Las frecuencias f_1, f_2, f_3, f_5 y f_6 detectadas en la señal son notas musicales: $Fa\#_3$, que es la frecuencia fundamental; $Fa\#_4$, que es el armónico de la octava superior; $Do\#_5$, que es el armónico del quinto intervalo en la segunda octava superior; y $Do\#_6$, que es el armónico del quinto intervalo en la tercera octava superior. Las frecuencias f_4 y f_5 sin embargo, son curiosas porque no corresponden a ninguna nota musical armónica en concordancia con la fundamental de la cuerda vibrante, pero podrían tratarse perfectamente de modos normales ya sea del material con que se construyeron las cuerdas o de las maderas del cuerpo del instrumento mismo.

En este ejemplo, la mayor potencia espectral está concentrada en f_4 y por lo tanto es la lectura que recibirá el dispositivo afinador. En cualquier caso, puede darse que este fenómeno origine detecciones erróneas. La forma de corregir este fenómeno que conduce a error sería agregar un filtro pasa bandas al tratamiento previo de la señal, para suprimir las frecuencias armónicas o los modos normales que entorpecen la lectura de la fundamental. Sin embargo, si se realiza esta operación, al igual que con el caso anterior, se estaría sacrificando versatilidad del dispositivo. Agregar un filtro implicaría tener certeza de cuál es la banda que hay que suprimir. Al suprimir una banda de frecuencias determinada, se cancela la detección de todas las notas musicales que se encuentren

dentro de esa banda y el dispositivo ya no serviría para el espectro completo de notas musicales.

4.2.7 Caídas de tensión

Otro comportamiento poco favorable para el desempeño y la utilización del dispositivo es que durante el proceso de ajuste de tensión de una cuerda específica, y si el torque que debe ser aplicado es positivo (la cuerda necesita aumentar su tensión), la pantalla LCD tiende a disminuir su brillo y a perder los caracteres o la información desplegada. Esto se debe a que en un proceso de aumento de tensión de la cuerda, se debe ejercer un torque mayor y el motor realiza un esfuerzo eléctrico superior a la media utilizada durante todo el proceso de funcionamiento del dispositivo.

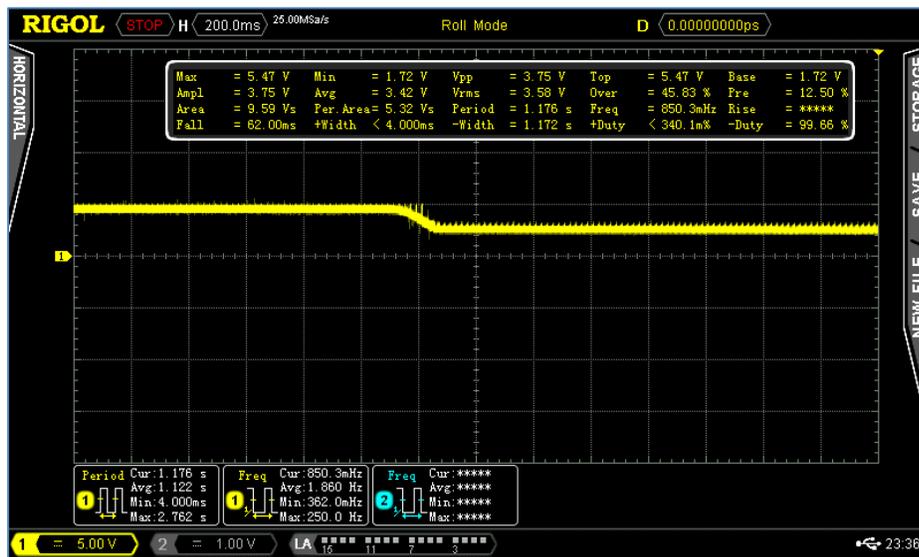


Figura 4-21: Caída de tensión del motor SERVO.

La Figura 4-21 muestra captura de pantalla del osciloscopio RIGOL MSO-4012 (ver anexo 7.5) obtenida en el laboratorio de electrónica del Departamento de Ingeniería Eléctrica de la Facultad de Ciencias Físicas y Matemáticas de la Universidad de Chile. En esta imagen se observa que la alimentación del motor SERVO (en amarillo) tiene un valor constante de 5[V], y súbitamente presenta una caída de tensión que le hace disminuir hasta los 1.72[V]. Esto es lo que ocurre cuando el motor es enfrentado a un torque de levante superior a su capacidad. Este valor de bajo voltaje también es percibido por los terminales del módulo LCD, lo que puede provocar que el micro-controlador interno deje de funcionar.

La Figura 3-43 muestra que el divisor de voltaje del offset; la pantalla LCD; el micro-controlador del módulo LCD; el motor SERVO; los botones; y el potenciómetro que regula el contraste de la pantalla están todos conectados a la alimentación de 5[V] proveída por el REDBOARD. Cuando el motor realiza un esfuerzo mayor al que es capaz de levantar ocurre una baja de tensión importante en la alimentación de los demás componentes del circuito.

En el caso extremo de bajas de tensión eléctrica en la alimentación se ha observado que el módulo LCD pierde sincronía en la comunicación con el ATmega328 y la pantalla deja de mostrar la información que debería desplegar. Otro comportamiento no deseado en un caso extremo de baja de tensión, es que el motor SERVO queda girando indefinidamente, ignorando futuras instrucciones del micro-controlador. Afortunadamente estos casos de errores son muy raros de ver y se presentan en situaciones de torque extremo, que el motor no es capaz de controlar. En tales ocasiones, se debe reiniciar por completo el dispositivo, esto es, cortar la alimentación energética del REDBOARD y comenzar nuevamente desde el principio la manipulación del afinador.

Una posible solución para corregir este problema, sería usar fuentes de voltaje independientes para el motor SERVO o para el módulo LCD, de manera que el funcionamiento a grandes esfuerzos eléctricos de uno, no altere el desempeño del otro.

5 CONCLUSION

Como resultado de este trabajo, se concluye que el objetivo general ha sido logrado: se cuenta con un prototipo de afinador electro-mecánico para instrumentos de cuerda con sistema sinfín-corona que cumple la tarea, en términos generales, de afinar eficazmente las cuerdas del instrumento utilizado. La mayoría de los tiempos de afinación de cada cuerda oscilan entre los 4 y 5 segundos, lo que es un valor razonable para preparar el uso de un instrumento antes de una presentación en vivo, por ejemplo 24 segundos en afinar una guitarra completamente.

Los objetivos específicos también han sido logrados: se propone una solución en base al conocimiento adquirido luego de observar el estado del arte; se hace uso de herramientas existentes para aplicaciones alternativas a las que fueron diseñadas; se implementa una interfaz de utilización de fácil uso (apretar botones y leer una pantalla); y se incluye todo el espectro de notas musicales reproducibles por los instrumentos de interés. Todos los resultados negativos son casos particulares que dependen de las condiciones bajo las que se realizaron las pruebas y tienen una razón técnica que las explica, o sea, pueden ser corregidos.

Este proyecto demuestra que es posible establecer un vínculo entre disciplinas radicalmente distintas como lo son la programación de algoritmos y la música. La investigación de distintas áreas de trabajo permite concebir aplicaciones y productos nuevos que no podrían ver la luz concentrándose en un solo campo de estudio.

Se expone también un ejemplo de que el uso de complejas herramientas matemáticas, como la FFT y la FHT; avanzados conceptos informáticos, como la programación orientada a objetos; propuesta circuitos eléctricos, como el amplificador no inversor y el sumador de componente continua, pueden todos en conjunto ser llevados a un contexto de aplicación cotidiana, como lo es la simple práctica de afinar una guitarra.

5.1 Trabajo futuro propuesto

Para finalizar este trabajo, se deja una ventana abierta que permita continuidad en los estudios aquí abordados. En primera instancia, se podrían mejorar los aspectos de funcionamiento según las siguientes sugerencias:

- Incluir más funcionalidades al dispositivo afinador. Por ejemplo, agregar dos botones más que hagan girar el motor SERVO a su máxima velocidad en ambos sentidos independiente de las lecturas que se estén obteniendo. Esta sería una herramienta útil para ayudar a encordar un instrumento y acelerar el proceso de enrollar la cuerda en la clavija antes de afinarla.
- Mejorar el desempeño del diseño aquí propuesto. Existen varias formas de lograr continuidad del proyecto abarcando este punto, por ejemplo, se puede comenzar el diseño de una versión dedicada del afinador propuesto en este trabajo. Si se construye un afinador, por ejemplo, dedicado sólo a afinar guitarras, se tiene un espectro de frecuencias mucho más limitado para detectar, lo cual permite aumentar la resolución de los bins, liberar memoria utilizada por el código que se carga en el micro-controlador. Además, con una propuesta de este tipo, se podrían incluir filtros pasa banda en el pre-tratamiento de la señal que eliminen las frecuencias residuales o armónicas que no sean de interés. Otra alternativa es, en la medida que el presupuesto lo permita, reemplazar los módulos usados en este diseño por elementos con mayor capacidad (ver sección 4.2), motor con mayor torque, micro-controlador con más memoria, fuentes de voltaje controladas para pantalla, etc.
- Incluir programas dedicados en el diseño propuesto. De manera que el usuario pueda, por ejemplo, seleccionar qué instrumento desea afinar y comenzar un algoritmo secuencial que afine las notas específicas de cada cuerda. Esto implicaría almacenar una base de datos de distintos tipos de instrumentos y sus afinaciones específicas.
- Investigar sobre métodos alternativos de detección de frecuencia que funcionen en tiempo real y no involucren almacenamiento de muestras periódicas que usen recursos limitados.
- El afinador propuesto en este trabajo cuenta con un solo motor y afina una cuerda a la vez, esto permite mayor versatilidad y permite su uso en distintos tipos de instrumentos. Sería interesante también construir un modelo con más de un motor, con el objetivo de afinar varias cuerdas al mismo tiempo, expandiendo el campo de estudio de procesamiento de señales a capturas de sonido con más de una nota sonado al mismo tiempo.

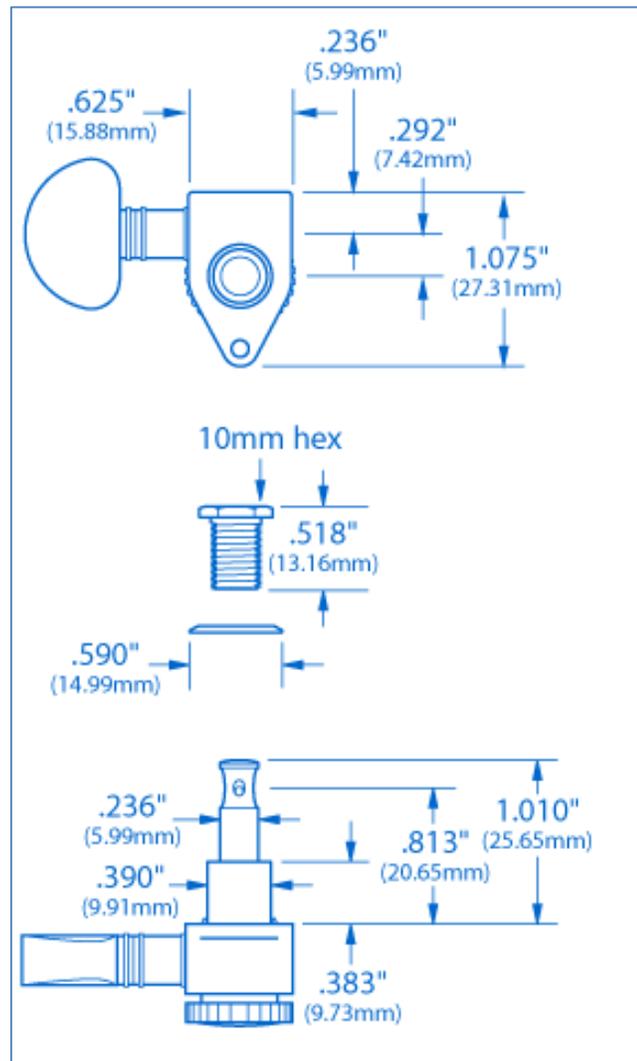
6 BIBLIOGRAFÍA

- [1] R. O. P. Javier Luzuriaga, *La física de los instrumentos musicales*, Buenos Aires: Universidad de Buenos Aires, 2007.
- [2] J. B. Marion, «La cuerda vibrante,» de *Dinámica Clásica de las Partículas y Sistemas*, Barcelona, Reverté, 2003, pp. 515-549.
- [3] R. W. Young, *Terminology for Logarithmic Frequency Units*, 1939.
- [4] O. M. Peralta, «Memoria de Titulación: Arpa Eólica,» Pontificia Universidad Católica de Valparaíso, Valparaíso, 1981.
- [5] M. Olmo y R. Nave, «HyperPhysics,» 2014. [En línea]. Available: <http://hyperphysics.phy-astr.gsu.edu/hbasees/music/cents.html>. [Último acceso: 11 agosto 2014].
- [6] A. Pulido, «Aprendemos Tecnología,» 2014. [En línea]. Available: <http://aprendemostecnologia.org/2009/03/19/el-tornillo-sinfin-y-la-rueda-dentada/>. [Último acceso: mayo 2014].
- [7] L. Stokowski, «Bell Laboratories et le développement,» [En línea]. Available: http://www.stokowski.org/Developpement_de_Enregistrement_Electrique.htm. [Último acceso: Junio 2014].
- [8] E. Brigham, *The Fast Fourier Transform*, New York: Prentice-Hall, 2002.
- [9] S. Hilbert, «FFT Zero Padding,» 22 Abril 2013. [En línea]. Available: <http://www.bitweenie.com/listings/fft-zero-padding/>. [Último acceso: 14 Agosto 2014].
- [10] F. Piccinin, «The Fast Hartley Transform as an alternative to the Fast Fourier Transform,» Department Of Defense, Salisbury, 1988.
- [11] R. J. Marks, *Introduction to Shannon Sampling and Interpolation Theory*, New York: Springer-Verlag, 1991.
- [12] D. Lavry, *Sampling Theory For Digital Audio*, Lavry Engineering Inc., 2004.
- [13] K. J. Aström y M. Richard, *Feedback Systems: An Introduction for Scientist and Engineers*, Princeton and Oxford: Princeton University Press, 2012.

- [14] North Staffordshire Royal Infirmary, Stoke-on-Trent., «The origin of the tuning fork,» *Journal of the Royal Society of Medicine*, vol. 80, pp. 771-773, 1987.
- [15] E. Szigety, M. A. Tintori Ferreira, J. Viau y L. Moro, «Estroboscópio Mecánico: Una experiencia visual,» Universidad Nacional de Mar del Plata, Buenos Aires, 2010.
- [16] N. Burgos, S. Sepúlveda y C. Torres, «Experiencia electiva: Transductores electroacústicos,» Facultad de Ciencias Físicas y Matemáticas de la Universidad de Chile, Santiago, 2014.
- [17] B. Weiss and G. Gridling, *Introduction To Microcontrollers*, Vienna: University of Technology, 2007.
- [18] A. SA, «Arduino,» Arduino SA, 2014. [En línea]. Available: <http://arduino.cc/>. [Último acceso: 25 Septiembre 2014].
- [19] A. Ghassaei, «Instructables,» [En línea]. Available: <http://www.instructables.com/id/Arduino-Timer-Interrupts/>. [Último acceso: 25 Septiembre 2014].
- [20] O. M. Labs, «Arduino FHT Libray,» 8 Julio 2014. [En línea]. Available: <http://wiki.openmusiclabs.com/wiki/ArduinoFFT>. [Último acceso: 12 Octubre 2014].
- [21] F. Trenaman, «Modifying a Servo Motor for Continuous Rotation,» 25 11 2013. [En línea]. Available: <http://www.youtube.com/watch?v=6G1nHfvI4al>. [Último acceso: 15 Octubre 2014].
- [22] J. J. Rey y A. Navarro, *Los instrumentos de púa en España: Bandurri, cítola y "laúdes españoles"*, Madrid: Alianza Editorial, 1993.

7 ANEXOS

7.1 Dimensiones de una clavija de guitarra GROVER



7.2 Datasheet SHURE PG48

Product Specifications

PG48 Cardioid Dynamic Vocal Microphone

Overview

The PG48 Dynamic Vocal Microphone is a professional quality microphone designed to be used for spoken word and karaoke performance applications. With rugged durability and excellent sound, the PG48 comes with an on/off switch, swivel adapter, carrying pouch and choice of XLR cable, QTR (¼") cable or without cable.

Features

- Tailored frequency response is smooth and extended. Tuned specifically for vocal applications.
- Cardioid polar pattern picks up the most sound from in front of the microphone and some sound from the sides. Less susceptible to feedback in high volume settings.
- Dynamic cartridge has a simple, rugged coil. Handles extreme volume levels without distortion.
- Durable metal construction.
- On/off switch for onstage control.
- Steel mesh ball grille with integral "pop" filter reduces explosive breath sounds and wind noise.

Available Models

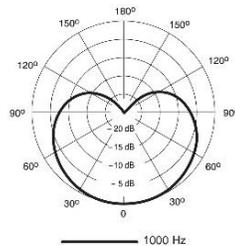
| | |
|-----------------|--|
| PG48-QTR | Includes a 15 ft (4.57 m) XLR to ¼" Cable, Microphone Clip, 5/8" to 3/8" Threaded Adapter, and a Storage Bag. |
| PG48-XLR | Includes a 15 ft (4.57 m) XLR to XLR Cable, Microphone Clip, 5/8" to 3/8" Threaded Adapter, and a Storage Bag. |
| PG48-LC | Includes a Microphone Clip, 5/8" to 3/8" Threaded Adapter, and a Storage Bag. |

Specifications

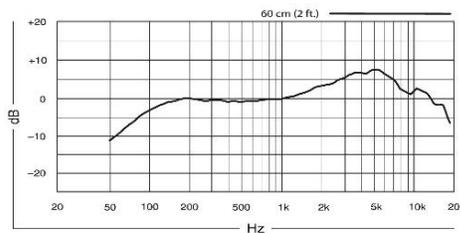
| | |
|---------------------------------|---|
| Microphone Type | Dynamic (moving coil) |
| Polar Pattern | Cardioid (unidirectional) |
| Frequency Response | 70 to 15,000 Hz |
| Polarity | Positive pressure on diaphragm produces positive voltage on pin 2 relative to pin 3 of microphone output connector |
| Output Impedance (at 1000 Hz) | 600 ohms |
| Sensitivity (at 1000 Hz) | -52 dBV/Pa, (2.5 mV), 1 Pascal = 94 dB SPL |
| Electromagnetic Hum Sensitivity | 22 dB equivalent SPL in a 1 millioersted field (60 Hz) |
| Connector | Three-pin professional audio connector (male XLR type) |
| Switch | Built-in On/Off switch |
| Environmental Conditions | This microphone operates over a temperature range of -29 to 57 degrees Celsius (-20 to 135 degrees Fahrenheit), and at a relative humidity of 0 to 95%. |
| Case | Die cast metal handle, black matte finish; hardened silver coated, spherical steel mesh grille |
| Weight | Net: 314 grams (11.1 oz) Packaged: 890 g (1 lb 1.5 oz) |



PG48



Polar Pattern
Measured at 1000 Hz



Frequency Response

SHURE[®]
LEGENDARY
PERFORMANCE™

www.shure.com

©2008 Shure Incorporated

7.3 Datasheet RODE NT-A

NT2-A

Multi Pattern 1" Condenser Microphone

RODE
MICROPHONES



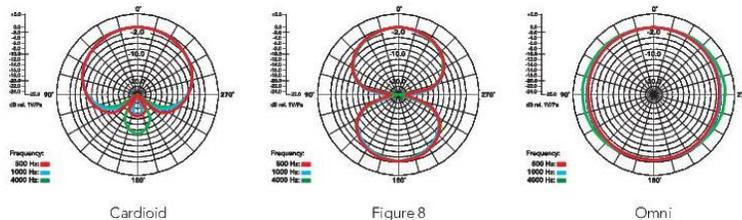
Features

- Large 1" (25mm) HF1 gold sputtered capsule
- On body control of polar pattern, HPF and PAD
- Three position variable polar pattern - Omni, Cardioid or Figure 8
- Three position variable High-Pass Filter - Flat, 40Hz or 80Hz
- Three position PAD - 0dB, -5dB or -10dB
- Ultra low noise, transformerless surface mount circuitry
- Includes SM6 shock mount with integrated pop filter, 3m XLR cable, dustcover and exclusive training DVD.
- Internal capsule shock mounting
- Designed and manufactured in Australia
- 10 year warranty*

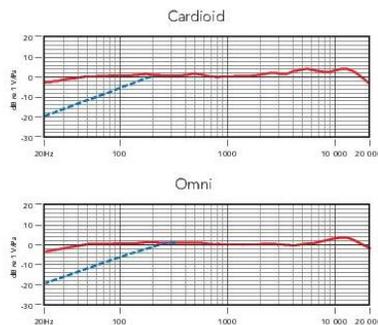
Specifications

| | | | |
|--------------------|--|---------------------|---|
| acoustic Principle | Pressure, Pressure gradient | Directional Pattern | Three position variable - Omni, Cardioid or Figure 8 |
| frequency Range | 20 Hz-20 kHz | Sensitivity | -36 dB re 1 Volt/Pascal (16 mV @ 94 dB SPL) +/- 2 dB @ 1kHz |
| Output Impedance | 200Ω | Dimensions | Length - 209mm (8.2283") Diameter - 55mm (2.1653") |
| Output Connection | 3 pin XLR, balanced output between Pin 2 (+), Pin 3 (-) and Pin 1 (ground) | Shipping Weight | 1kg |
| Net Weight | 860g | | |

Polar Pattern



Frequency Response



Included Accessories

- SM6 Shock mount
- 3m (10') XLR cable
- Microphone dustcover
- Exclusive NT2-A training DVD



www.rodemic.com



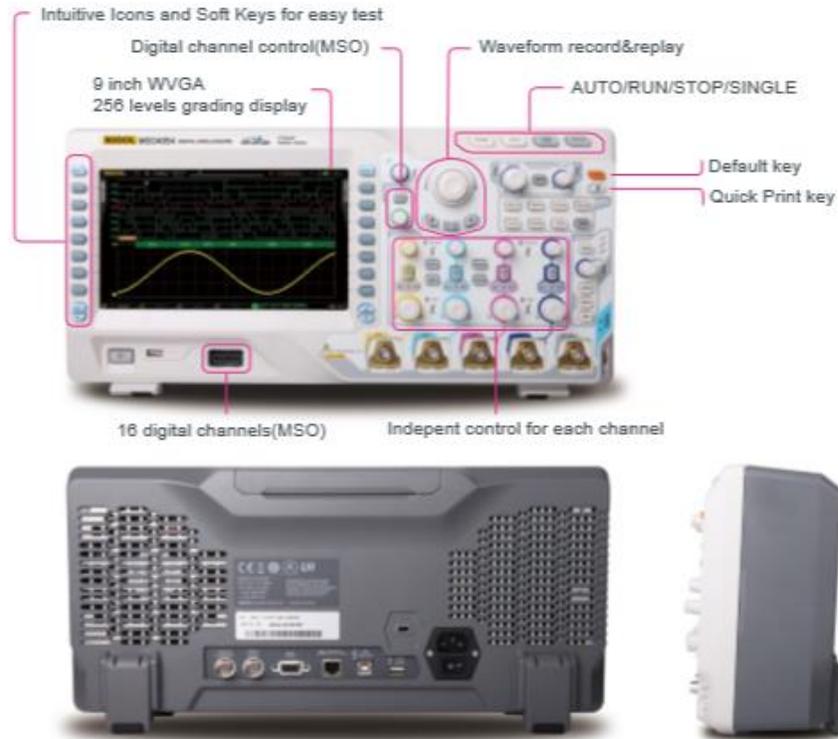
*After online registration

Australia
107 Carnarvon st, Silverwater NSW 2128 Australia
Ph: +61 2 9648 5855 Fx: +61 2 9648 2455

USA
PO Box 4189 Santa Barbara CA 93140 USA
Ph: +1 805 566 7777 Fx: +1 805 566 0071

7.5 Datasheet RIGOL MSO4012

MSO/DS4000 Series Digital Oscilloscope



Product Dimensions: Width X Height X Depth = 440.0mm X 218.0 mm X 130.0 mm Weight: 4.6 kg ± 0.2 kg (Without Package)

► Innovative UltraVision technology(Analog Channel)



- Deeper Memory Depth(Std.140M pts)
- Higher Waveform capture rate (Up to 110,000wfms/s)
- Real Time waveform record & replay(Up to 200,000 frames)
- Multi-level intensity grading display(Up to 256 levels)

► Models and key Specs

| Model Number | DS4054 | DS4052 | DS4034 | DS4032 | DS4024 | DS4022 | DS4014 | DS4012 |
|---|--|---------|---------|---------|---------|---------|---------|---------|
| | MSO4054 | MSO4052 | MSO4034 | MSO4032 | MSO4024 | MSO4022 | MSO4014 | MSO4012 |
| Analog BW | 500 MHz | | 350 MHz | | 200 MHz | | 100 MHz | |
| Analog Channels | 4 | 2 | 4 | 2 | 4 | 2 | 4 | 2 |
| Digital Channels(MSO) | 16 | | | | | | | |
| Max. Sample rate | Analog Channel: Max. 4 GSa/s single channel, 2GSa/s dual channel Digital Channel: Max. 1 GSa/s per channel | | | | | | | |
| Max. Memory Depth | Analog Channel: Std. up to 140 Mpts single channel,70 Mpts dual channel Digital Channel: Std. up to 28 Mpts per channel | | | | | | | |
| Waveform Capture rate | Up to 110,000 wfms/s(Digital channel Closed); 85,000 wfms/s(Digital channel Opened) | | | | | | | |
| Real Time waveform Record, Replay and Analysis function | Analog channel:Up to 200,000 frames(Std.) Digital channel:Up to 64,000 frames(Std.) | | | | | | | |
| Std. Probes | 2 or 4 sets RP3500A 500MHz BW Passive Probe; 1 set RPL2316 LA Probe(MSO only) | | | | | | | |

RIGOL

► Specifications

All the specifications are guaranteed except parameters marked with "Typical" and the oscilloscope needs to operate for more than 30 minutes under the specified operation temperature.

Sample

| | |
|-----------------------|--|
| Sample Mode | Real-time sample |
| Real-time Sample Rate | Analog channel: 4.0 Gsa/s (single-channel); 2.0 Gsa/s (dual-channel) Digital channel: 1.0 Gsa/s |
| Peak Detect | Analog channel: 250 ps (single-channel); 500 ps (dual-channel) Digital channel: 1 ns |
| Averaging | After all the channels finish N samples at the same time, N can be 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096 or 8192. |
| High Resolution | 12 bit of resolution When $\geq 5 \mu\text{s}/\text{div}$ @ 4 Gsa/s (or $\geq 10 \mu\text{s}/\text{div}$ @ 2 Gsa/s). |
| Memory Depth | Analog channel: Single-channel: Auto, 14k pts, 140k pts, 1.4M pts, 14M pts and 140M pts are available Dual-channel: Auto, 7k pts, 70k pts, 700k pts, 7M pts and 70M pts are available Digital channel: maximum 28 M pts |

Input

| | |
|----------------------------------|--|
| Number of Channels | MSO40X4: four-analog-channel + 16-digital-channel MSO40X2: dual-analog-channel + 16-digital-channel DS40X4: four-channel DS40x2: dual-channel |
| Input Coupling | DC, AC or GND |
| Input Impedance | Analog channel: $(1\text{M}\Omega \pm 1\%) \parallel (14 \text{ pF} \pm 3 \text{ pF})$ or $50 \Omega \pm 1.5\%$ Digital channel: $(101 \text{ k}\Omega \pm 1\%) \parallel (9 \text{ pF} \pm 1 \text{ pF})$ |
| Probe Attenuation Coefficient | Analog channel: 0.01X-1000X 1-2-5 step |
| Max Input Voltage (1M Ω) | Maximum input voltage of the analog channel CAT I 300 Vrms, CAT II 100 Vrms, transient overvoltage 1000 Vpk With RP2200 10:1 probe: CAT II 300 Vrms With RP3300 10:1 probe: CAT II 300 Vrms With RP3500 10:1 probe: CAT II 300 Vrms With RP5600 10:1 probe: CAT II 300 Vrms Digital channel: CAT I 40Vrms, transient overvoltage 800 Vpk |

Horizontal

| | |
|------------------------------------|--|
| Time Base Scale | MSO405X/DS405X: 1 ns/div to 1000 s/div MSO403X/DS403X: 2 ns/div to 1000 s/div MSO402X/DS402X: 2 ns/div to 1000 s/div MSO401X/DS401X: 5 ns/div to 1000 s/div |
| Time Base Accuracy | $\leq \pm 4 \text{ ppm}$ |
| Time Base Drift | $\leq \pm 2 \text{ ppm/Year}$ |
| Delay Range | Pre-trigger (negative delay): ≥ 1 screen width Post-trigger (positive delay): 1 s to 1000 s |
| Time Base Mode | Y-T, X-Y, Roll, Delayed |
| Number of X-Ys | 2 paths at the same time (four-channel model) |
| Waveform Capture Rate ¹ | 110,000 wfms/s(Dots display, Digital channel Closed); 85,000 wfms/s(Digital channel Opened) |

Vertical(Analog Channel)

| | |
|---------------------|--|
| Bandwidth (-3dB) | MSO405X/DS405X: DC to 500 MHz MSO403X/DS403X: DC to 350 MHz MSO402X/DS402X: DC to 200 MHz MSO401X/DS401X: DC to 100 MHz |
| Single Bandwidth | MSO405X/DS405X: DC to 500 MHz MSO403X/DS403X: DC to 350 MHz MSO402X/DS402X: DC to 200 MHz MSO401X/DS401X: DC to 100 MHz |
| Vertical Resolution | Analog channel: 8 bit, two channels sample at the same time Digital channel: 1bit |

RIGOL

| | |
|--|--|
| Vertical Scale | 1 mV/div to 5 V/div (1 M Ω) 1 mV/div to 1 V/div (50 Ω) |
| Offset Range | 1 mV/div to 124 mV/div: ± 1.2 V (50 Ω) 126 mV/div to 1 V/div: ± 12 V (50 Ω) 1 mV/div to 225 mV/div: ± 2 V (1M Ω) 230 mV/div to 5 V/div: ± 40 V (1M Ω) |
| Bandwidth Limit ² | M50405X/M50403X/DS405X/DS403X: 20 MHz/100 MHz/200 MHz M50402X/DS402X: 20 MHz/100 MHz M50401X/DS401X: 20 MHz |
| Low Frequency Response (AC coupling, -3dB) | ≤ 5 Hz (on BNC) |
| Rise Time ² | M50405X/DS405X: 700 ps M50403X/DS403X: 1 ns M50402X/DS402X: 1.8 ns M50401X/DS401X: 3.5 ns |
| DC Gain Accuracy | $\pm 2\%$ full scale |
| DC Offset Accuracy | 200 mV/div to 5 V/div: 0.1 div ± 2 mV $\pm 0.5\%$ offset 1 mV/div to 195 mV/div: 0.1 div ± 2 mV $\pm 1.5\%$ offset |
| ESD Tolerance | ± 2 kV |
| Channel to Channel Isolation | DC to maximum bandwidth: >40 dB |

Vertical (Digital Channel)

| | |
|---------------------|--|
| Threshold | 1 group with 8 channels adjustable threshold |
| Threshold selected | TTL (1.4 V) 5.0 V CMOS (+2.5 V), 3.3 V CMOS (+1.65 V) 2.5 V CMOS (+1.25 V), 1.8 V CMOS (+0.9 V) ECL (-1.3 V) PECL (+3.7 V) LVDS (+1.2 V) 0 V User |
| Threshold range | ± 20.0 V, with 10 mV step |
| Threshold accuracy | ± 100 mV+3% of threshold setting |
| Dynamic range | ± 10 V + threshold |
| Min Voltage Swing | 500 mVpp |
| Vertical resolution | 1 bit |

Trigger

| | | |
|---------------------------------------|--|---|
| Trigger Level Range | Internal | ± 6 div from the center of the screen |
| | EXT | ± 0.8 V |
| Trigger Mode | Auto, Normal, Single | |
| Holdoff Range | 100 ns to 10 s | |
| High Frequency Rejection ² | 50 kHz | |
| Low Frequency Rejection ² | 5 kHz | |
| Edge Trigger | | |
| Edge Type | Rising, Falling, Rising&Falling | |
| Pulse Trigger | | |
| Pulse Condition | Positive Pulse Width (greater than, lower than, within specified interval) Negative Pulse Width (greater than, lower than, within specified interval) | |
| Pulse Width Range | 4 ns to 4 s | |
| Runt Trigger | | |
| Pulse Condition | None, > (greater than), < (lower than), <> (within the specified interval) | |
| Polarity | Positive, Negative | |
| Pulse Width Range | 4 ns to 4 s | |
| Nth Edge Trigger | | |
| Edge Type | Rising, Falling | |
| Idle Time | 40 ns to 1 s | |
| Number of Edges | 1 to 85535 | |
| Slope Trigger | | |
| Slope Condition | Positive Slope (greater than, lower than, within specified interval) Negative Slope (greater than, lower than, within specified interval) | |
| Time Setting | 10 ns to 1 s | |

RIGOL

| | | |
|------------------------------|---|---|
| Video Trigger | | |
| Signal Standard | Support standard NTSC, PAL and SECAM broadcasting standards Support 480P, 576P, 720P, 1080P and 1080I HDTV standards | |
| Pattern Trigger | | |
| Pattern Setting | H, L, X, Rising Edge, Falling Edge | |
| RS232/UART Trigger | | |
| Trigger Condition | Start, Error, Check Error, Data | |
| Polarity | Normal, Invert | |
| Baud | 2400 bps, 4800 bps, 9600 bps, 19200 bps, 38400 bps, 57600 bps, 115200 bps, User | |
| Data Bits | 5 bit, 6 bit, 7 bit, 8 bit | |
| I2C Trigger | | |
| Trigger Condition | Start, Restart, Stop, Missing Ack, Address, Data, A&D | |
| Address Bits | 7 bit, 8 bit, 10 bit | |
| Address Range | 0 to 127, 0 to 255, 0 to 1023 | |
| Byte Length | 1 to 5 | |
| SPI Trigger | | |
| Trigger Condition | CS, TimeOut | |
| Timeout Value | 100 ns to 1 s | |
| Data Bits | 4 bit to 32 bit | |
| Data Line Setting | H, L, X | |
| Clock Edge | Rising edge, Falling edge | |
| CAN Trigger | | |
| Signal Type | Rx, Tx, CAN_H, CAN_L, Differential | |
| Trigger Condition | SOF, EOF, Frame Type, Frame Error | |
| Baud | 10 kbps, 20 kbps, 33.3 kbps, 50 kbps, 62.5 kbps, 83.3 kbps, 100 kbps, 125 kbps, 250 kbps, 500 kbps, 800 kbps, 1 Mbps, User | |
| Sample Point | 5% to 95% | |
| Frame Type | Data, Remote, Error, OverLoad | |
| Error Type | Bit Fill, Answer Error, Check Error, Format Error, Random Error | |
| FlexRay Trigger | | |
| Baud | 2.5 Mb/s, 5 Mb/s, 10 Mb/s | |
| Trigger Condition | Frame, Symbol, Error, TSS | |
| USB Trigger | | |
| Signal Speed | Low Speed, Full Speed | |
| Trigger Condition | SOP, EOP, RC, Suspend, ExitSuspend | |
| Measure | | |
| Cursor | Manual mode | Voltage deviation between cursors (ΔV) Time deviation between cursors (ΔT) Reciprocal of ΔT (Hz) ($1/\Delta T$) |
| | Track mode | Voltage and time values of the waveform point |
| | Auto mode | Allow to display cursors during auto measurement |
| Auto Measurement | Measurements of Maximum, Minimum, Peak-Peak Value, Top Value, Bottom Value, Amplitude, Average, Mean Square Root, Overshoot, Pre-shoot, Area, Period Area, Frequency, Period, Rise Time, Fall Time, Positive Pulse Width, Negative Pulse Width, Positive Duty Cycle, Negative Duty Cycle, Delay A→B $\bar{\#}$, Delay A→B $\bar{\#}$, Phase A→B $\bar{\#}$, Phase A→B $\bar{\#}$ | |
| Number of Measurements | Display 5 measurements at the same time | |
| Measurement Range | Screen Region or Cursor Region | |
| Measurement Statistic | Average, Max, Min, Standard Deviation, Number of Measurements | |
| Counter | Hardware 8 bits counter (channels are selectable) | |
| Math | | |
| Waveform Operation | A+B, A-B, A×B, A/B, FFT, Editable Advanced Operation, Logic Operation | |
| FFT Window | Rectangle, Hanning, Blackman, Hamming | |
| FFT Display | Split, Full Screen | |
| FFT Vertical Scale | dB, Vrms | |
| Logic Operation | AND, OR, NOT, XOR | |
| Math Function | Intg, Diff, Log, Exp, Sqrt, Sine, Cosine, Tangent | |
| Number of Buses for Decoding | 2 | |
| Decoding Type | Parallel (standard), RS232 /UART (option), I2C (option), SPI (MSO4XX4/DS4XX4 option), CAN (option), FlexRay (option) | |
| RIGOL | | |

Display

| | |
|--------------------|---|
| Display Type | 9 inches (229 mm) TFT LCD display |
| Display Resolution | 800 horizontal×RGB×480 vertical pixel |
| Display Color | 160,000 color |
| Persistence Time | Min, 50 ms, 100 ms, 200 ms, 500 ms, 1 s, 2 s, 5 s, 10 s, 20 s, Infinite |
| Display Type | Dots, Vectors |
| Real-time Clock | Time and Date (user adjustable) |

I/O

| | |
|----------------|--|
| Standard Ports | Dual USB HOST, USB DEVICE, LAN, VGA output, 10MHz input/output, Aux output (TrigOut, Fast, GND, PassFail, Calibration) |
|----------------|--|

General Specifications

| | | |
|---|--|-----------------|
| Probe Compensation Output | | |
| Output Voltage ² | About 3 V, peak-peak | |
| Frequency ² | 1 kHz | |
| Power | | |
| Power Voltage | 100-120 V/50Hz/60Hz/400Hz 100-240 V/50 Hz/60Hz | |
| Power | Maximum 120 W | |
| Fuse | 3 A, T degree, 250 V | |
| Environment | | |
| Temperature Range | In operation: 0°C to +50°C Out of operation: -40°C to +70°C | |
| Cooling Method | Fan | |
| Humidity Range | 0°C to 30°C : ≤95% relative humidity +30°C to +40°C : ≤75% relative humidity +40°C to +50°C : ≤45% relative humidity | |
| Altitude | In operation: under 3,000 meters Out of operation: under 15,000 meters | |
| Mechanical | | |
| Dimensions ² | Width×Height×Depth =440.0 mm× 218.0 mm×130.0 mm | |
| Weight ⁴ | Without package | 4.8 kg ± 0.2 kg |
| | With package | 7.1 kg ± 1.0kg |
| Adjustment Interval | | |
| The recommended calibration interval is one year. | | |
| Regulation Standards | | |
| Electromagnetic Compatibility | 2004/108/EC Execution standard EN 61326-1:2006 EN 61326-2-1:2006 | |
| safety | UL 61010-1:2004; CAN/CSA-C22.2 NO. 61010-1-2004; EN 61010-1:2001; IEC 61010-1:2001 | |

1. Maximum value. In single-channel mode, sine signal with 10 ns horizontal scale, 4 div input amplitude and 10 MHz frequency, edge trigger.
2. Typical.
3. Tilt tabs and handle folded, knob height included, front panel cover excluded.
4. Standard configuration.

7.6 Datasheet TL082



TL082

TL082A - TL082B

GENERAL PURPOSE J-FET DUAL OPERATIONAL AMPLIFIER

- WIDE COMMON-MODE (UP TO V_{CC}^+) AND DIFFERENTIAL VOLTAGE RANGE
- LOW INPUT BIAS AND OFFSET CURRENT
- OUTPUT SHORT-CIRCUIT PROTECTION
- HIGH INPUT IMPEDANCE J-FET INPUT STAGE
- INTERNAL FREQUENCY COMPENSATION
- LATCH UP FREE OPERATION
- HIGH SLEW RATE : $16V/\mu s$ (typ)



N
DIP8
(Plastic Package)

D
SO8
(Plastic Micropackage)



P
TSSOP8
(Thin Shrink Small Outline Package)

DESCRIPTION

The TL082, TL082A and TL082B are high speed J-FET input dual operational amplifiers incorporating well matched, high voltage J-FET and bipolar transistors in a monolithic integrated circuit.

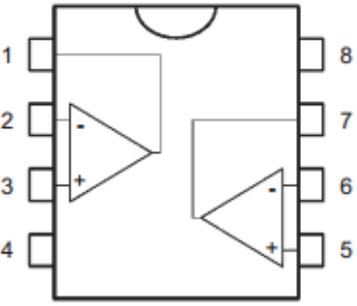
The devices feature high slew rates, low input bias and offset current, and low offset voltage temperature coefficient.

ORDER CODES

| Part Number | Temperature Range | Package | | |
|--------------|-------------------|---------|---|---|
| | | N | D | P |
| TL082M/AM/BM | -55°C, +125°C | • | • | • |
| TL082I/AI/BI | -40°C, +105°C | • | • | • |
| TL082C/AC/BC | 0°C, +70°C | • | • | • |

Examples : TL082CD, TL082IN

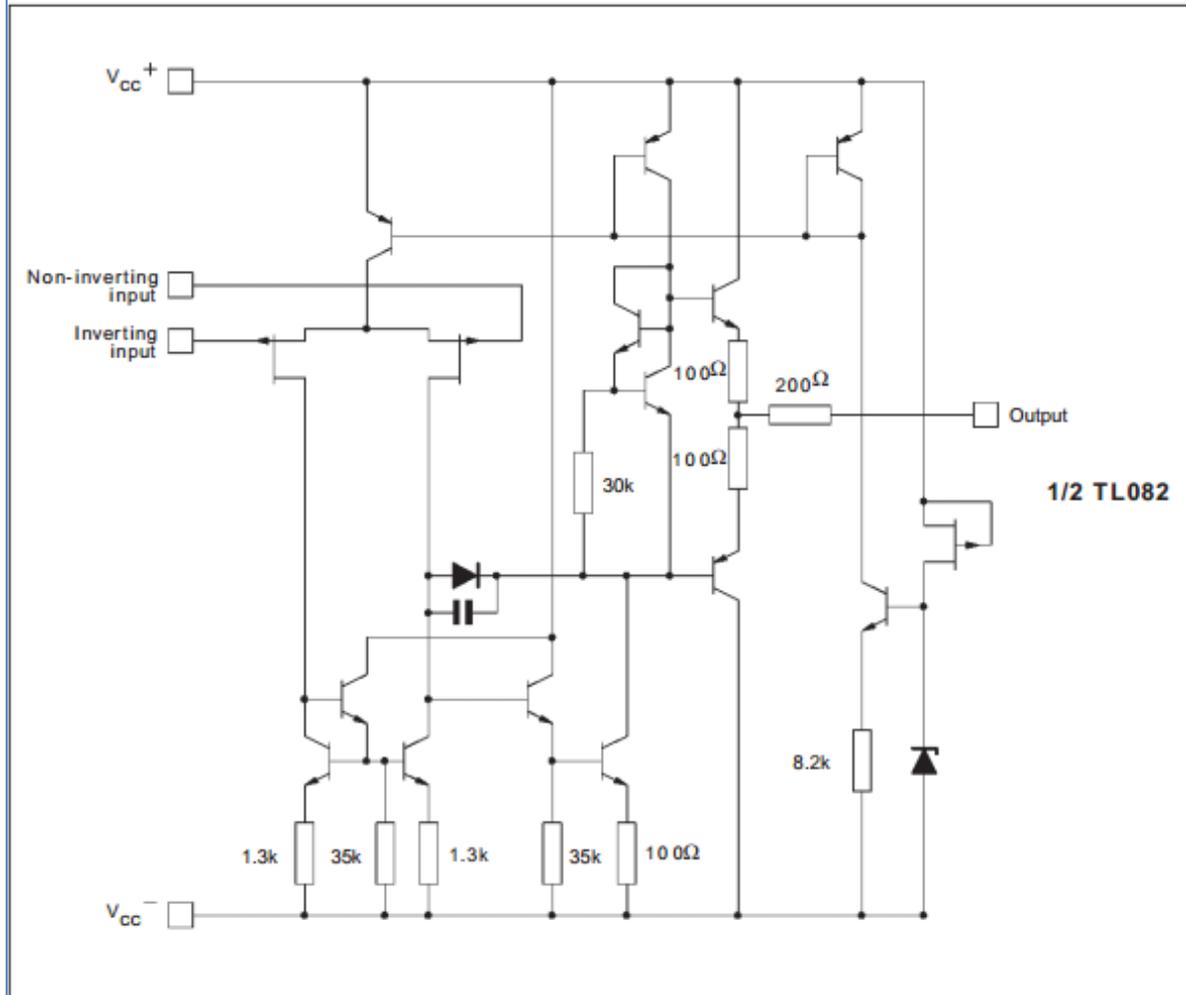
PIN CONNECTIONS (top view)



1 - Output 1
 2 - Inverting input 1
 3 - Non-inverting input 1
 4 - V_{CC}^-
 5 - Non-inverting input 2
 6 - Inverting input 2
 7 - Output 2
 8 - V_{CC}^+

TL082 - TL082A - TL082B

SCHEMATIC DIAGRAM



ABSOLUTE MAXIMUM RATINGS

| Symbol | Parameter | Value | Unit | |
|------------|--|--|---|--------------------|
| V_{CC} | Supply Voltage - (note 1) | ± 18 | V | |
| V_i | Input Voltage - (note 3) | ± 15 | V | |
| V_{id} | Differential Input Voltage - (note 2) | ± 30 | V | |
| P_{Tot} | Power Dissipation | 680 | mW | |
| | Output Short-circuit Duration - (note 4) | Infinite | | |
| T_{oper} | Operating Free Air Temperature Range | TL082C,AC,BC TL082I,AI,BI TL082M,AM,BM | 0 to 70 -40 to 105 -55 to 125 | $^{\circ}\text{C}$ |
| T_{stg} | Storage Temperature Range | | -65 to 150 | $^{\circ}\text{C}$ |

- Notes :
1. All voltage values, except differential voltage, are with respect to the zero reference level (ground) of the supply voltages where the zero reference level is the midpoint between V_{CC}^+ and V_{CC}^- .
 2. Differential voltages are at the non-inverting input terminal with respect to the inverting input terminal.
 3. The magnitude of the input voltage must never exceed the magnitude of the supply voltage or 15 volts, whichever is less.
 4. The output may be shorted to ground or to either supply. Temperature and /or supply voltages must be limited to ensure that the dissipation rating is not exceeded.

TL082 - TL082A - TL082B

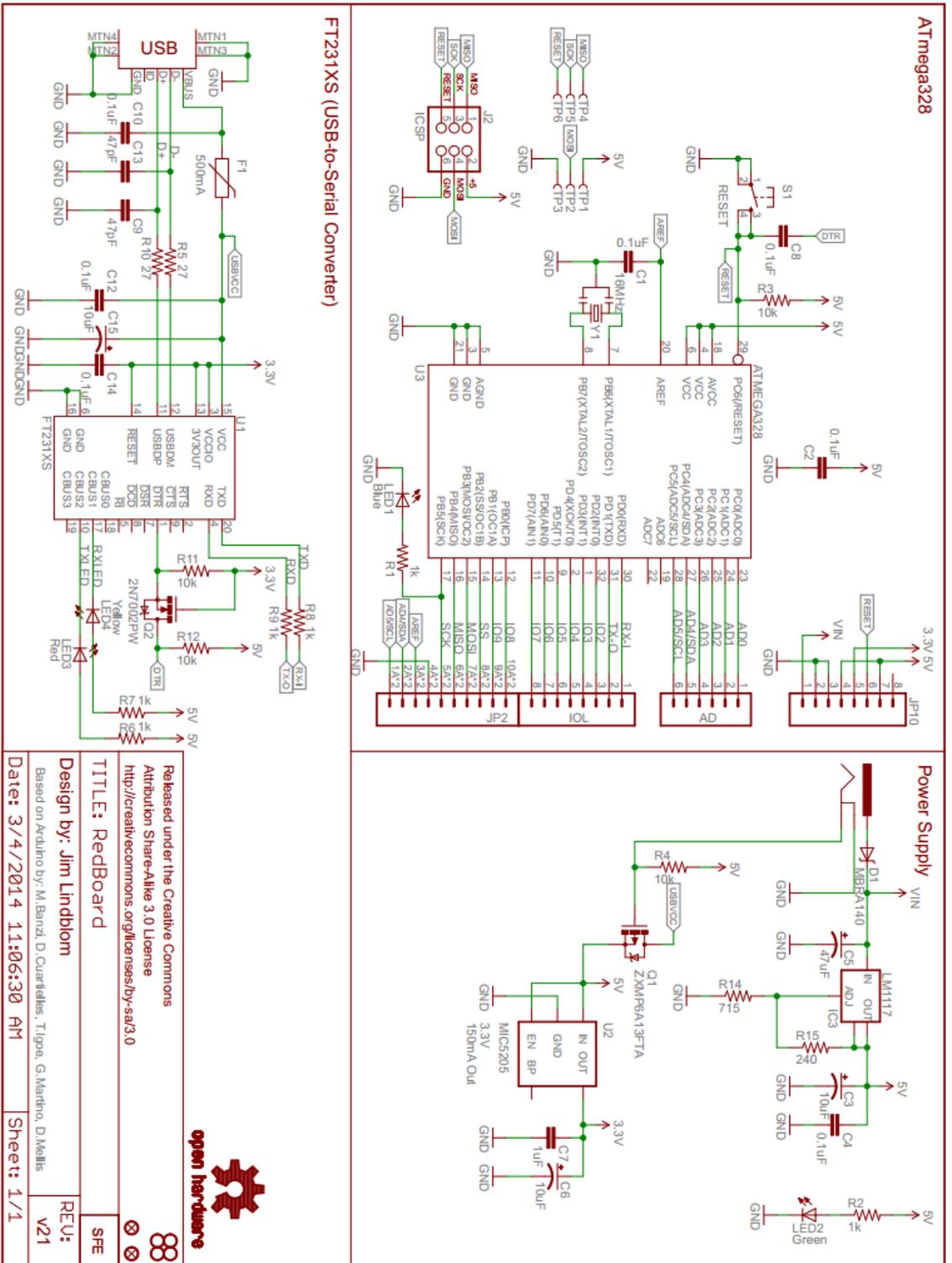
ELECTRICAL CHARACTERISTICS

V_{CC} = ±15V, T_{amb} = 25°C (unless otherwise specified)

| Symbol | Parameter | TL082I,M,AC,AI, AM,BC,BI,BM | | | TL082C | | | Unit |
|----------------------------------|---|--------------------------------|----------------------|------------------------------|----------------------|------------------|--------------|------------------------|
| | | Min. | Typ. | Max. | Min. | Typ. | Max. | |
| V _{io} | Input Offset Voltage (R _S = 50Ω) T _{amb} = 25°C T _{min.} ≤ T _{amb} ≤ T _{max.} | | 3 3 1 | 10 6 3 13 7 5 | | 3 | 10 13 | mV |
| DV _{io} | Input Offset Voltage Drift | | 10 | | | 10 | | μV/°C |
| I _{io} | Input Offset Current * T _{amb} = 25°C T _{min.} ≤ T _{amb} ≤ T _{max.} | | 5 | 100 4 | | 5 | 100 4 | pA nA |
| I _{ib} | Input Bias Current * T _{amb} = 25°C T _{min.} ≤ T _{amb} ≤ T _{max.} | | 20 | 200 20 | | 20 | 400 20 | pA nA |
| A _{vd} | Large Signal Voltage Gain (R _L = 2kΩ, V _O = ±10V) T _{amb} = 25°C T _{min.} ≤ T _{amb} ≤ T _{max.} | 50 25 | 200 | | 25 15 | 200 | | V/mV |
| SVR | Supply Voltage Rejection Ratio (R _S = 50Ω) T _{amb} = 25°C T _{min.} ≤ T _{amb} ≤ T _{max.} | 80 80 | 86 | | 70 70 | 86 | | dB |
| I _{cc} | Supply Current, per Amp, no Load T _{amb} = 25°C T _{min.} ≤ T _{amb} ≤ T _{max.} | | 1.4 | 2.5 2.5 | | 1.4 | 2.5 2.5 | mA |
| V _{icm} | Input Common Mode Voltage Range | ±11 | +15 -12 | | ±11 | +15 -12 | | V |
| CMR | Common Mode Rejection Ratio (R _S = 50Ω) T _{amb} = 25°C T _{min.} ≤ T _{amb} ≤ T _{max.} | 80 80 | 86 | | 70 70 | 86 | | dB |
| I _{os} | Output Short-circuit Current T _{amb} = 25°C T _{min.} ≤ T _{amb} ≤ T _{max.} | 10 10 | 40 | 60 60 | 10 10 | 40 | 60 60 | mA |
| ±V _{Opp} | Output Voltage Swing T _{amb} = 25°C T _{min.} ≤ T _{amb} ≤ T _{max.} | | 10 12 10 12 | 12 13.5 | 10 12 10 12 | 12 13.5 | | V |
| SR | Slew Rate (V _{in} = 10V, R _L = 2kΩ, C _L = 100pF, T _{amb} = 25°C, unity gain) | 8 | 16 | | 8 | 16 | | V/μs |
| t _r | Rise Time (V _{in} = 20mV, R _L = 2kΩ, C _L = 100pF, T _{amb} = 25°C, unity gain) | | 0.1 | | | 0.1 | | μs |
| K _{OV} | Overshoot (V _{in} = 20mV, R _L = 2kΩ, C _L = 100pF, T _{amb} = 25°C, unity gain) | | 10 | | | 10 | | % |
| GBP | Gain Bandwidth Product (f = 100kHz, T _{amb} = 25°C, V _{in} = 10mV, R _L = 2kΩ, C _L = 100pF) | 2.5 | 4 | | 2.5 | 4 | | MHz |
| R _i | Input Resistance | | 10 ¹² | | | 10 ¹² | | Ω |
| THD | Total Harmonic Distortion (f = 1kHz, A _v = 20dB, R _L = 2kΩ, C _L = 100pF, T _{amb} = 25°C, V _O = 2V _{pp}) | | 0.01 | | | 0.01 | | % |
| e _n | Equivalent Input Noise Voltage (f = 1kHz, R _S = 100Ω) | | 15 | | | 15 | | $\frac{nV}{\sqrt{Hz}}$ |
| ∅ _m | Phase Margin | | 45 | | | 45 | | Degrees |
| V _{O1} /V _{O2} | Channel Separation (A _v = 100) | | 120 | | | 120 | | dB |

* The input bias currents are junction leakage currents which approximately double for every 10°C increase in the junction temperature.

7.7 Sparkfun REDBOARD

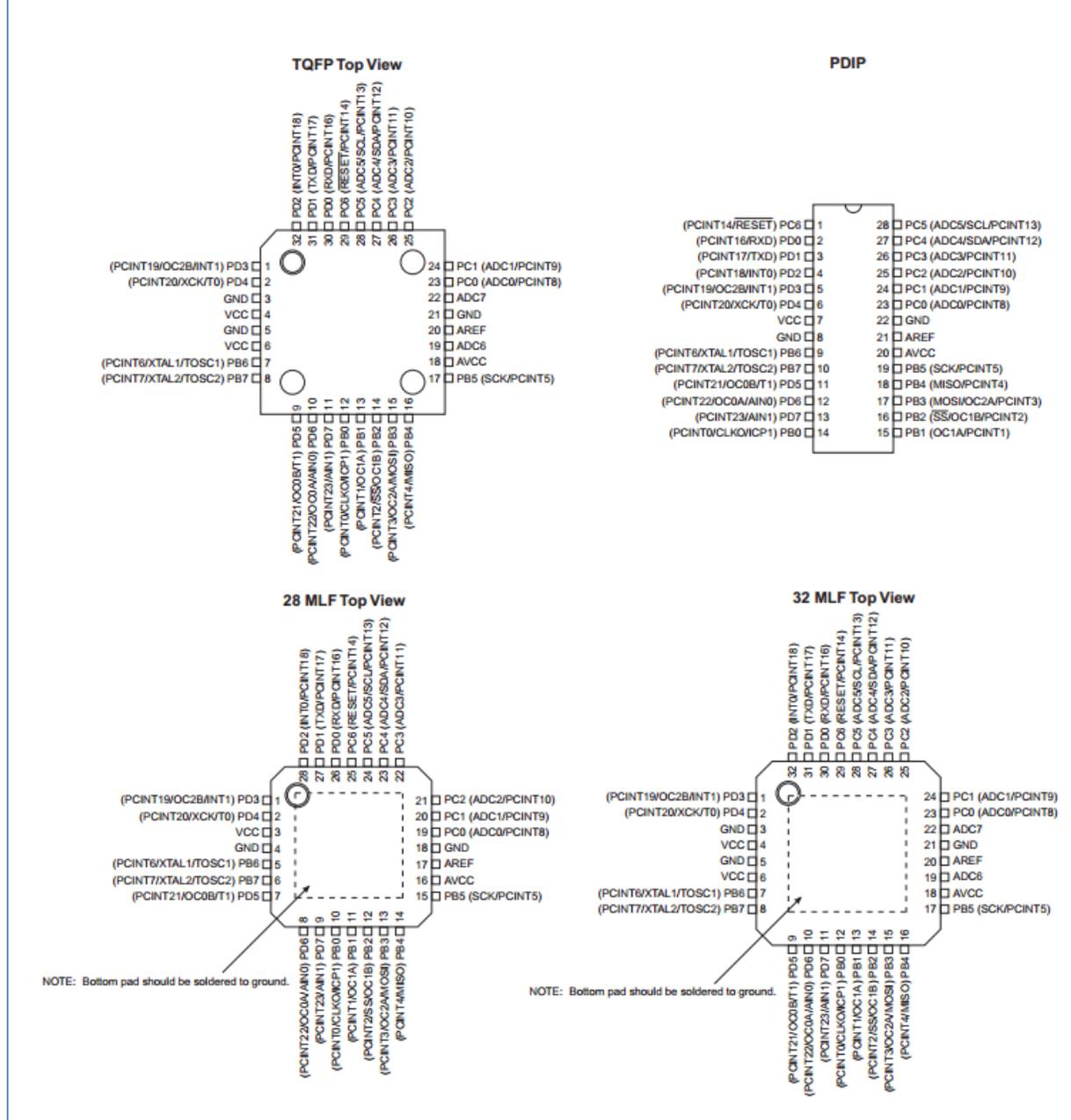


7.8 Datasheet ATmega328

ATmega48PA/88PA/168PA/328P

1. Pin Configurations

Figure 1-1. Pinout ATmega48PA/88PA/168PA/328P



1.1 Pin Descriptions

1.1.1 VCC

Digital supply voltage.

1.1.2 GND

Ground.

1.1.3 Port B (PB7:0) XTAL1/XTAL2/TOSC1/TOSC2

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Depending on the clock selection fuse settings, PB6 can be used as input to the inverting Oscillator amplifier and input to the internal clock operating circuit.

Depending on the clock selection fuse settings, PB7 can be used as output from the inverting Oscillator amplifier.

If the Internal Calibrated RC Oscillator is used as chip clock source, PB7..6 is used as TOSC2..1 input for the Asynchronous Timer/Counter2 if the AS2 bit in ASSR is set.

The various special features of Port B are elaborated in ["Alternate Functions of Port B" on page 82](#) and ["System Clock and Clock Options" on page 26](#).

1.1.4 Port C (PC5:0)

Port C is a 7-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The PC5..0 output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running.

1.1.5 PC6/RESET

If the RSTDISBL Fuse is programmed, PC6 is used as an I/O pin. Note that the electrical characteristics of PC6 differ from those of the other pins of Port C.

If the RSTDISBL Fuse is unprogrammed, PC6 is used as a Reset input. A low level on this pin for longer than the minimum pulse length will generate a Reset, even if the clock is not running. The minimum pulse length is given in [Table 28-3 on page 318](#). Shorter pulses are not guaranteed to generate a Reset.

The various special features of Port C are elaborated in ["Alternate Functions of Port C" on page 85](#).

1.1.6 Port D (PD7:0)

Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running.

The various special features of Port D are elaborated in ["Alternate Functions of Port D" on page 88](#).

1.1.7 **AV_{CC}**

AV_{CC} is the supply voltage pin for the A/D Converter, PC3:0, and ADC7:6. It should be externally connected to V_{CC}, even if the ADC is not used. If the ADC is used, it should be connected to V_{CC} through a low-pass filter. Note that PC6..4 use digital supply voltage, V_{CC}.

1.1.8 **AREF**

AREF is the analog reference pin for the A/D Converter.

1.1.9 **ADC7:6 (TQFP and QFN/MLF Package Only)**

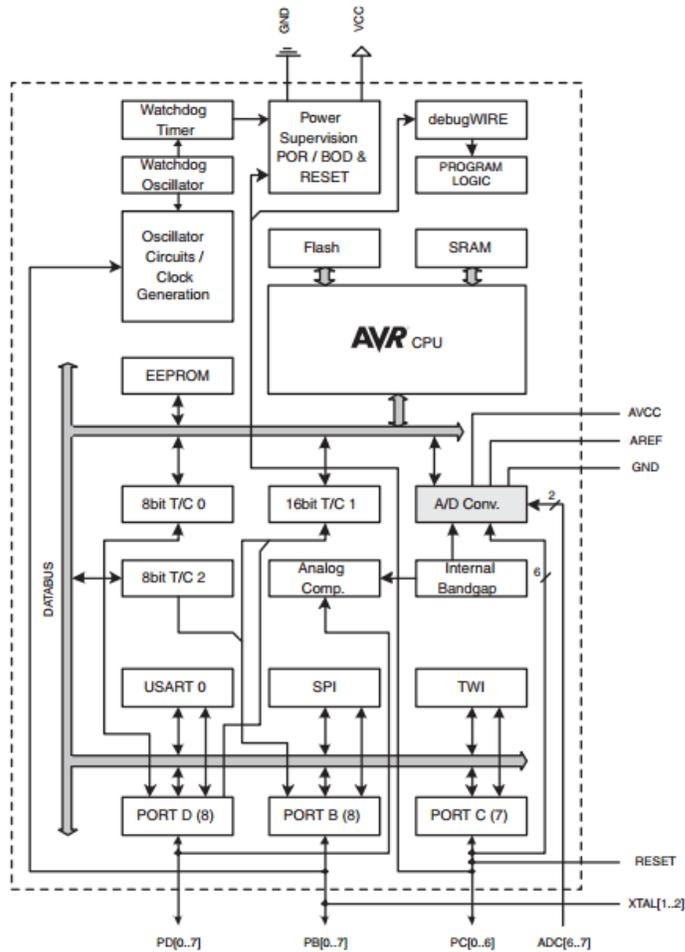
In the TQFP and QFN/MLF package, ADC7:6 serve as analog inputs to the A/D converter. These pins are powered from the analog supply and serve as 10-bit ADC channels.

2. Overview

The ATmega48PA/88PA/168PA/328P is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega48PA/88PA/168PA/328P achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

2.1 Block Diagram

Figure 2-1. Block Diagram



The AVR core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting

architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.

The ATmega48PA/88PA/168PA/328P provides the following features: 4K/8K bytes of In-System Programmable Flash with Read-While-Write capabilities, 256/512/512/1K bytes EEPROM, 512/1K/1K/2K bytes SRAM, 23 general purpose I/O lines, 32 general purpose working registers, three flexible Timer/Counters with compare modes, internal and external interrupts, a serial programmable USART, a byte-oriented 2-wire Serial Interface, an SPI serial port, a 6-channel 10-bit ADC (8 channels in TQFP and QFN/MLF packages), a programmable Watchdog Timer with internal Oscillator, and five software selectable power saving modes. The Idle mode stops the CPU while allowing the SRAM, Timer/Counters, USART, 2-wire Serial Interface, SPI port, and interrupt system to continue functioning. The Power-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next interrupt or hardware reset. In Power-save mode, the asynchronous timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC Noise Reduction mode stops the CPU and all I/O modules except asynchronous timer and ADC, to minimize switching noise during ADC conversions. In Standby mode, the crystal/resonator Oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low power consumption.

The device is manufactured using Atmel's high density non-volatile memory technology. The On-chip ISP Flash allows the program memory to be reprogrammed In-System through an SPI serial interface, by a conventional non-volatile memory programmer, or by an On-chip Boot program running on the AVR core. The Boot program can use any interface to download the application program in the Application Flash memory. Software in the Boot Flash section will continue to run while the Application Flash section is updated, providing true Read-While-Write operation. By combining an 8-bit RISC CPU with In-System Self-Programmable Flash on a monolithic chip, the Atmel ATmega48PA/88PA/168PA/328P is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications.

The ATmega48PA/88PA/168PA/328P AVR is supported with a full suite of program and system development tools including: C Compilers, Macro Assemblers, Program Debugger/Simulators, In-Circuit Emulators, and Evaluation kits.

2.2 Comparison Between ATmega48PA, ATmega88PA, ATmega168PA and ATmega328P

The ATmega48PA, ATmega88PA, ATmega168PA and ATmega328P differ only in memory sizes, boot loader support, and interrupt vector sizes. [Table 2-1](#) summarizes the different memory and interrupt vector sizes for the three devices.

Table 2-1. Memory Size Summary

| Device | Flash | EEPROM | RAM | Interrupt Vector Size |
|-------------|-----------|-----------|-----------|----------------------------|
| ATmega48PA | 4K Bytes | 256 Bytes | 512 Bytes | 1 instruction word/vector |
| ATmega88PA | 8K Bytes | 512 Bytes | 1K Bytes | 1 instruction word/vector |
| ATmega168PA | 16K Bytes | 512 Bytes | 1K Bytes | 2 instruction words/vector |
| ATmega328P | 32K Bytes | 1K Bytes | 2K Bytes | 2 instruction words/vector |

ATmega88PA, ATmega168PA and ATmega328P support a real Read-While-Write Self-Programming mechanism. There is a separate Boot Loader Section, and the SPM instruction can only execute from there. In ATmega48PA, there is no Read-While-Write support and no separate Boot Loader Section. The SPM instruction can execute from the entire Flash.

3. Resources

A comprehensive set of development tools, application notes and datasheets are available for download on <http://www.atmel.com/avr>.

4. Data Retention

Reliability Qualification results show that the projected data retention failure rate is much less than 1 PPM over 20 years at 85°C or 100 years at 25°C.

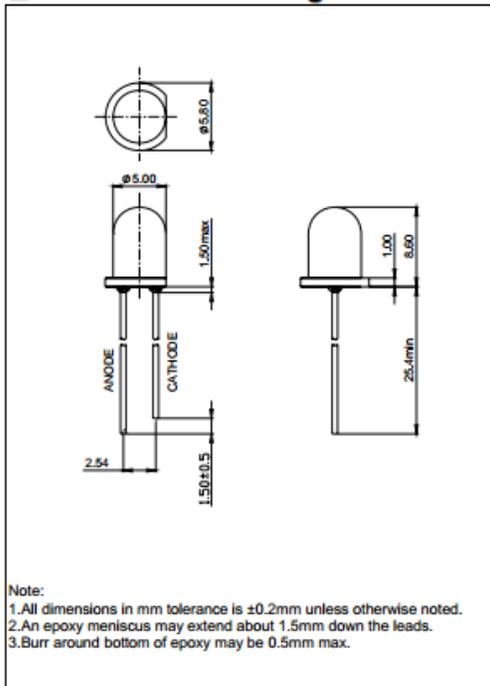
5. About Code Examples

This documentation contains simple code examples that briefly show how to use various parts of the device. These code examples assume that the part specific header file is included before compilation. Be aware that not all C compiler vendors include bit definitions in the header files and interrupt handling in C is compiler dependent. Please confirm with the C compiler documentation for more details.

For I/O Registers located in extended I/O map, "IN", "OUT", "SBIS", "SBIC", "CBI", and "SBI" instructions must be replaced with instructions that allow access to extended I/O. Typically "LDS" and "STS" combined with "SBR", "SBRC", "SBR", and "CBR".

7.9 Datasheet LED luminoso

Dimension Drawing



Applications:

- Toys
- Lighting
- Traffic light
- Automotive
- Commercial Outdoor Advertising
- Front Panel Indicator

Absolute Maximum Ratings ($T_a = 25^\circ\text{C}$)

| Items | Symbol | Absolute maximum Rating | Unit |
|----------------------------|-----------|--|------------------|
| Forward Current(DC) | I_F | 50 | mA |
| Peak Forward Current* | I_{FP} | 100 | mA |
| Reverse Voltage | V_R | 5 | V |
| Power Dissipation | P_D | 150 | mW |
| Operation Temperature | T_{opr} | -20 ~ +95 | $^\circ\text{C}$ |
| Storage Temperature | T_{stg} | -40 ~ +100 | $^\circ\text{C}$ |
| Lead Soldering Temperature | T_{sol} | Max.260 $^\circ\text{C}$ for 5 sec Max. (3mm from the base of the epoxy bulb) | |

*pulse width ≤ 0.1 msec duty $\leq 1/10$

Typical Electrical & Optical Characteristics ($T_a = 25^\circ\text{C}$)

| Items | Symbol | Condition | Min. | Typ. | Max. | Unit |
|---------------------|-----------------|---------------------|------|------|------|---------------|
| Forward Voltage | V_F | $I_F = 20\text{mA}$ | 1.8 | --- | 2.4 | V |
| Reverse Current | I_R | $V_R = 5\text{V}$ | --- | --- | 10 | μA |
| Dominant Wavelength | λ_D | $I_F = 20\text{mA}$ | 618 | --- | 630 | nm |
| Luminous Intensity | I_V | $I_F = 20\text{mA}$ | 1700 | --- | 3500 | mcd |
| 50% Power Angle | $2\theta_{1/2}$ | $I_F = 20\text{mA}$ | --- | 30 | --- | deg |

Ranks Combination ($I_F = 20\text{mA}$)

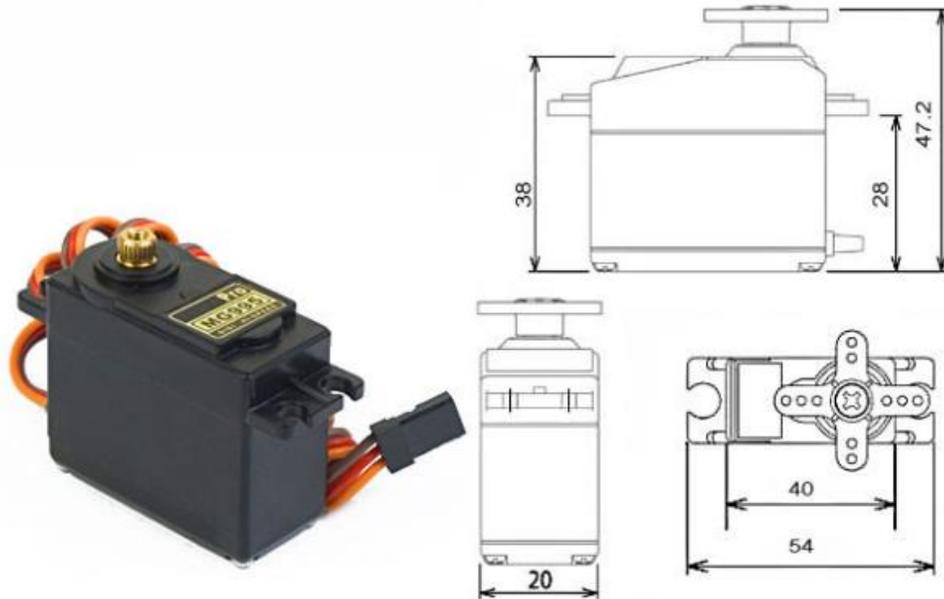
| | | | | | | |
|--------------------------|-----------|-----------|-----------|-----------|---------|---------|
| Rank | 0D | 0E | 0F | 0G | --- | |
| Dominant Wavelength (nm) | 618-621 | 621-624 | 624-627 | 627-630 | --- | |
| Rank | 0Q | 1R | 2R | 0S | --- | |
| Luminous Intensity (mcd) | 1700-2000 | 2000-2500 | 2500-3000 | 3000-3500 | --- | |
| Rank | 0G | 0H | 0J | 0K | 0L | 0M |
| Forward Voltage(V) | 1.8-1.9 | 1.9-2.0 | 2.0-2.1 | 2.1-2.2 | 2.2-2.3 | 2.3-2.4 |

Important Notes:

- 1) All ranks will be included per delivery.
- 2) Tolerance of measurement of luminous intensity is $\pm 15\%$.
- 3) Tolerance of measurement of dominant wavelength is $\pm 1\text{nm}$.
- 4) Tolerance of measurement of forward voltage is $\pm 0.05\text{V}$.
- 5) Pb content < 1000PPM.

7.10 Motor SERVO MG995

MG995 High Speed Metal Gear Dual Ball Bearing Servo

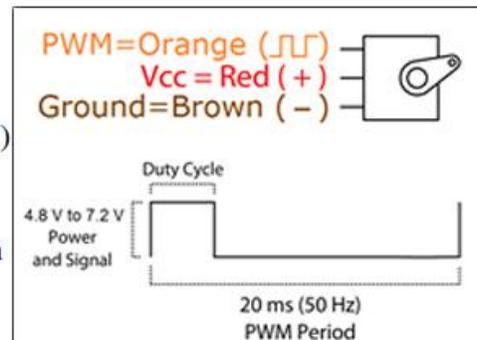


The unit comes complete with 30cm wire and 3 pin 'S' type female header connector that fits most receivers, including Futaba, JR, GWS, Cirrus, Blue Bird, Blue Arrow, Corona, Berg, Spektrum and Hitec.

This high-speed standard servo can rotate approximately 120 degrees (60 in each direction). You can use any servo code, hardware or library to control these servos, so it's great for beginners who want to make stuff move without building a motor controller with feedback & gear box, especially since it will fit in small places. The MG995 Metal Gear Servo also comes with a selection of arms and hardware to get you set up nice and fast!

Specifications

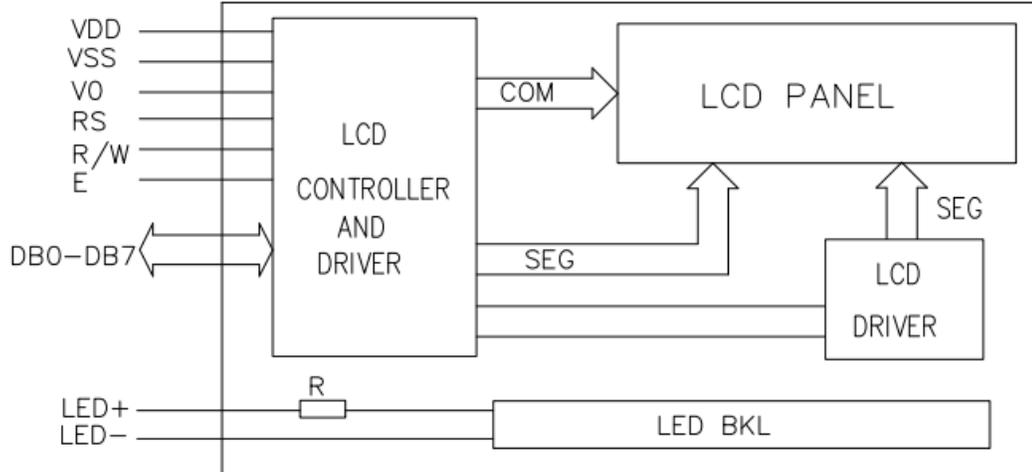
- Weight: 55 g
- Dimension: 40.7 x 19.7 x 42.9 mm approx.
- Stall torque: 8.5 kgf·cm (4.8 V), 10 kgf·cm (6 V)
- Operating speed: 0.2 s/60° (4.8 V), 0.16 s/60° (6 V)
- Operating voltage: 4.8 V a 7.2 V
- Dead band width: 5 μ s
- Stable and shock proof double ball bearing design
- Temperature range: 0 °C – 55 °C



4. Absolute maximum ratings

| Item | Symbol | Standard | | | Unit |
|-----------------------------|--------------|----------|---|-----|------|
| Power voltage | V_{DD-VSS} | 0 | - | 7.0 | V |
| Input voltage | V_{IN} | VSS | - | VDD | |
| Operating temperature range | V_{OP} | 0 | - | +50 | °C |
| Storage temperature range | V_{ST} | -10 | - | +60 | |

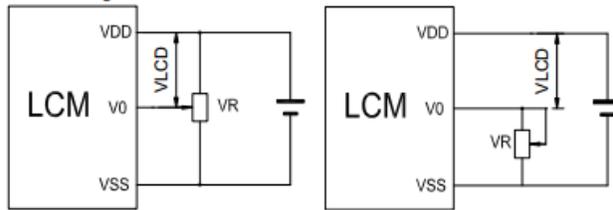
5. Block diagram



6. Interface pin description

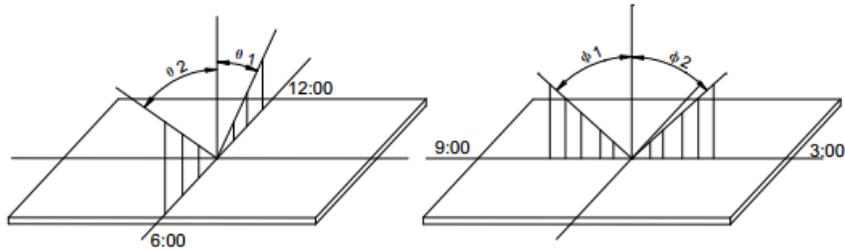
| Pin no. | Symbol | External connection | Function |
|---------|---------|----------------------|---|
| 1 | VSS | Power supply | Signal ground for LCM |
| 2 | VDD | | Power supply for logic for LCM |
| 3 | V0 | | Contrast adjust |
| 4 | RS | MPU | Register select signal |
| 5 | R/W | MPU | Read/write select signal |
| 6 | E | MPU | Operation (data read/write) enable signal |
| 7~10 | DB0~DB3 | MPU | Four low order bi-directional three-state data bus lines. Used for data transfer between the MPU and the LCM. These four are not used during 4-bit operation. |
| 11~14 | DB4~DB7 | MPU | Four high order bi-directional three-state data bus lines. Used for data transfer between the MPU |
| 15 | LED+ | LED BKL power supply | Power supply for BKL |
| 16 | LED- | | Power supply for BKL |

7. Contrast adjust



$V_{DD}-V_0$: LCD Driving voltage VR: 10k~20k

8. Optical characteristics



STN type display module ($T_a=25^\circ\text{C}$, $V_{DD}=3.3\text{V}$)

| Item | Symbol | Condition | Min. | Typ. | Max. | Unit |
|----------------------|------------|--------------|------|------|------|------|
| Viewing angle | $\theta 1$ | $C_r \geq 3$ | | 20 | | deg |
| | $\theta 2$ | | | 40 | | |
| | $\Phi 1$ | | | 35 | | |
| | $\Phi 2$ | | | 35 | | |
| Contrast ratio | C_r | | - | 10 | - | - |
| Response time (rise) | T_r | - | - | 200 | 250 | ms |
| Response time (fall) | T_f | - | - | 300 | 350 | |

9. Electrical characteristics

DC characteristics

| Parameter | Symbol | Conditions | Min. | Typ. | Max. | Unit |
|--------------------------|--------------|---|------|------|----------|---------------|
| Supply voltage for LCD | $V_{DD}-V_0$ | $T_a=25^\circ\text{C}$ | - | 3.0 | - | V |
| Input voltage | V_{DD} | | 3.1 | 3.3 | 3.5 | |
| Supply current | I_{DD} | $T_a=25^\circ\text{C}$, $V_{DD}=3.3\text{V}$ | - | 1.5 | 2.5 | mA |
| Input leakage current | I_{LKG} | | - | - | 1.0 | μA |
| "H" level input voltage | V_{IH} | | 2.2 | - | V_{DD} | V |
| "L" level input voltage | V_{IL} | Twice initial value or less | 0 | - | 0.6 | |
| "H" level output voltage | V_{OH} | LOH=-0.25mA | 2.4 | - | - | |
| "L" level output voltage | V_{OL} | LOH=1.6mA | - | - | 0.4 | |
| Backlight supply voltage | V_F | | - | 1.9 | | mA |
| Backlight supply current | I_{LED} | $V_{LED}=3.3\text{V}$ $R=100\ \Omega$ | | | 16 | |

7.12 Datasheet SMS/PSM

SCHURTER GmbH
D – 79346 Endingen
www.schurter.com

 **SCHURTER**
ELECTRONIC COMPONENTS

1 Description

1.1 SMS/PMS Base module

Miniature push button switches with a low height of 4,55 - 4,95 mm for surface mounting (SMS) and PCB mounting (PMS).

The SMS has large flat surfaces on the top side as well as on the other sides, which are also parallel to each other. This makes the SMS a perfect switch for automatic mounting.

The SMS switch is suitable for the SMD soldering process "IR-Reflow".

The switch comes with the SMD-leads "Gullwing and J". With J-leads the switch can be lined up with a spacing of 1/2" in one coordinate direction, and with > 13,5 mm in the other coordinate direction. With Gullwing-leads, the switch can be arranged with a spacing of 1/2" in one coordinate direction, and in the other coordinate direction with > 17,5 mm.

A minimum spacing of 1/2" to 15 mm is necessary for the PCB version.

Basically, the SMS and PMS come in two basic versions concerning the degree of protection. Available are IP 40 and IP 67. According to the degree of protection the IP 40 version is not proof against fluxing and washing, whereas the IP 67 version is. Consequently, the IP 67 version can be exposed to the specified soldering and cleaning processes.

The miniature push button switches feature a very good tactile response with an actuation force of about 2N. SMS and PMS are also available with an elongated actuator. These variants serve as base modules for the SMS/PMS variable height version.



1.2 SMS/PMS Variable Height

The variable height SMS/PMS consists of the SMS/PMS base module with elongated actuator and a slip-on button with eight variable heights.

The PMS will be supplied with a mounted button. The button for the SMS has to be ordered separately. After soldering, the button must be put on the base module with elongated actuator.

Heights between 8,5 mm and 13,75 mm for the SMS and 8,35 mm and 13,60 mm for the PMS are available. Depending on the base module being used, degree of protection for the variable height SMS/PMS is IP 40 or IP 67.



Changes that contribute to technical improvement are subject to alternations

| Page | Production date: | Produced by: | Modification date: | Modified by: | Modification No. | Data sheet No. | Index |
|--------|------------------|--------------|--------------------|--------------|------------------|----------------|-------|
| 2 of 9 | 07.07.2005 | Lickert | 15.06.2006 | M.Fischer | 9235 | 105.9513 | - |

Print date: 6/15/2007 2:39:00 PM

2 Data and dimensional drawings

2.1 Technical Data SMS/PMS Base module/Variable Height

| Electrical data: | IP40 | IP67 |
|---|---------------------------------------|---------------------------|
| Contact material | Gold ; Gold/Silver ⁽¹⁾ | Gold |
| Switching voltage max. | 30V AC/ 42V DC | 30V AC/ 42V DC |
| Switching current max. | 50 mA | 50 mA |
| Rated breaking capacity | 12 V/10 mA | 12 V/10 mA |
| Lifetime (at 12V/10mA) | >1 x 10 ⁶ cycles | >1x10 ⁶ cycles |
| Lifetime (at 24V/80mA) | - ; >1x10 ⁵ ⁽¹⁾ | - |
| Initial contact resistance new (IEC 512-2 mV-method) | <50 mOhm | <50 mOhm |
| Initial contact resistance after 1 x 10 ⁶ cycles | <150 mOhm | <150 mOhm |
| Insulation resistance (IEC 512-2) | > 1x 10 ⁸ Ohm | > 1x 10 ⁸ Ohm |
| Contact bounce time | typ. 0,15 ms | typ. 0,15 ms |

| Mechanical data: | IP40 | IP67 |
|--|----------------------------|---------------------|
| Actuating force | 1,8±0,4 N | 2,2±0,4 N |
| Actuating travel | 0,35±0,1 mm | 0,35±0,1 mm |
| Mechanical strength (force axial, load 1 min.) | max. 100 N | max. 100 N |
| Lifetime (IEC 512-5. Test 9a. Actuating force 5N) | >1x 10 ⁶ cycles | >1x 10 ⁶ |

| Soldering data: | SMS IP40/IP67 | PMS IP40/IP67 |
|---------------------------|--------------------------|--------------------------|
| Soldering method | IR Reflow | Wave soldering |
| Soldering heat resistance | 245 °C/5sec. | 248,5 °C/1sec |

⁽¹⁾ PMS Typ 1241.1652

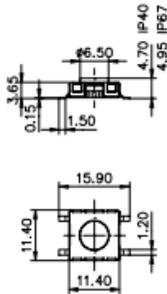
Changes that contribute to technical improvement are subject to alternations

| Page | Production date: | Produced by: | Modification date: | Modified by: | Modification No. | Data sheet No. | Index |
|--------|------------------|--------------|--------------------|--------------|------------------|----------------|-------|
| 3 of 9 | 07.07.2005 | Lickert | 15.06.2006 | M.Fischer | 9235 | 105.9513 | - |

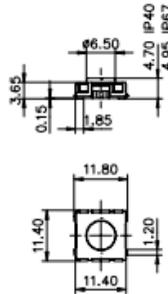
Print date: 6/15/2007 2:39:00 PM

2.2 Dimensions SMS/PMS Base module/variable height

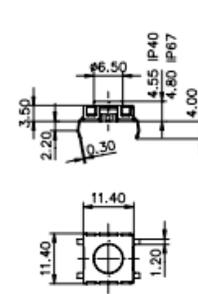
SMS Gullwing Base module



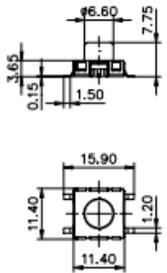
SMS J-lead Base module



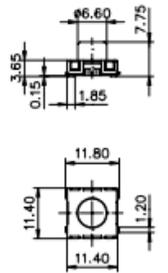
PMS PCB Base module



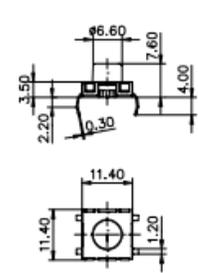
SMS Gullwing Variable height



SMS J-lead Variable height

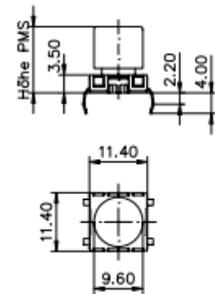
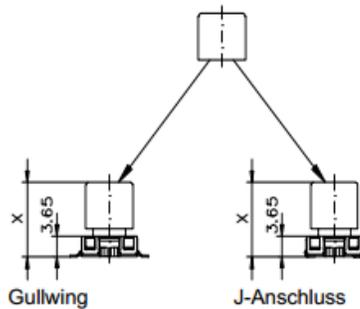


PMS PCB Variable height



SMS Gullwing and J lead with elongated button

PMS PCB with elongated button



Total height information: **See point 3.1.** Part numbers SMS und PMS Variable height, SMS elongated button must be ordered separately.

Changes that contribute to technical improvement are subject to alternations

| Page | Production date: | Produced by: | Modification date: | Modified by: | Modification No. | Data sheet No. | Index |
|--------|------------------|--------------|--------------------|--------------|------------------|----------------|-------|
| 6 of 9 | 07.07.2005 | Lickert | 15.06.2006 | M.Fischer | 9235 | 105.9513 | - |

Print date: 6/15/2007 2:39:00 PM

7.13 Encordado para guitarra D'ADDARIO EJ47 80/20.

Other Images



Tension Chart

| Item # | Note | Diameter | | Tension | |
|--------|------|----------|--------|---------|-------|
| | | Inches | mm | lbs | kg |
| J4501 | E | 0.0280 | 0.7112 | 16.230 | 7.360 |
| J4502 | B | 0.0322 | 0.8180 | 12.010 | 5.460 |
| J4503 | G | 0.0403 | 1.0236 | 11.880 | 5.390 |
| J4704 | D | 0.0290 | 0.7366 | 14.690 | 6.660 |
| J4705 | A | 0.0350 | 0.8890 | 13.910 | 6.310 |
| J4706 | E | 0.0430 | 1.0922 | 13.270 | 6.020 |

[Family Tension Chart](#)

Description

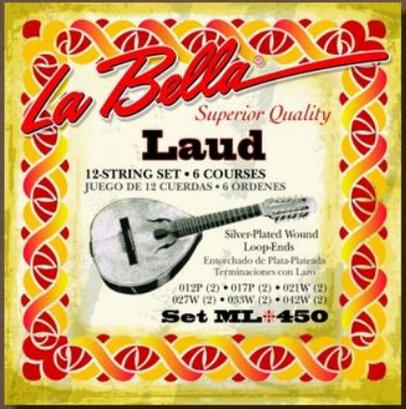
EJ47, normal tension, features 80/20 bronze wound basses for bright acoustic tone, comfortable feel and dynamic projection.

Pro-Arte, the world's most popular classical strings are D'Addario's premium classical guitar string sets. All Pro-Arte treble strings are sorted by a sophisticated computer-controlled laser machine which performs diameter and tension measurements, ensuring true-sounding, precise trebles in every set. Pro-Arte basses are wound using silver-plated copper on a multi-filament nylon core for consistent and warm, yet projecting tone.

- Normal tension offers balance of volume and comfortable resistance
- Laser sorted clear nylon treble strings for consistent, precise intonation
- Precision wound basses for warm, projecting tone and the utmost consistency
- Made in the U.S.A. for the highest quality and performance
- String Gauges: Trebles .0280, .0322, .0403 Basses .029, .35, .043
- Environmentally friendly, corrosion resistant packaging for strings that are always fresh

7.14 Encordado para laúd español LA BELLA ML450

La Bella
NEWS HISTORY PRODUCTS ARTISTS LUTHIERS DEALERS GUIDES BUY NOW



ML450 LAUD

The laud is a tenor-pitched member of the Spanish cittern family and is tuned one octave lower than the Bandurria. It is a large flat-backed pear-shaped instrument with origins tracing back to the 12th century when the Arabic influence left its musical mark on the Iberian Peninsula.

Tuning: G#2 G#2, C#3 C#3, F#3 F#3, B3 B3, E4 E4, A4 A4

- 12-STRING SET * 6 COURSES
- Plain Steel / Silver-Plated Wound
- Loop-Ends

1. 012P - 012P
2. 017P - 017P
3. 021W - 021W
4. 027W - 027W
5. 033W - 033W
6. 042W - 042W

7.15 Encordado para bandurria LA BELLA MB550

La Bella NEWS HISTORY PRODUCTS ARTISTS LUTHIERS DEALERS GUIDES BUY NOW



MB550 BANDURRIA

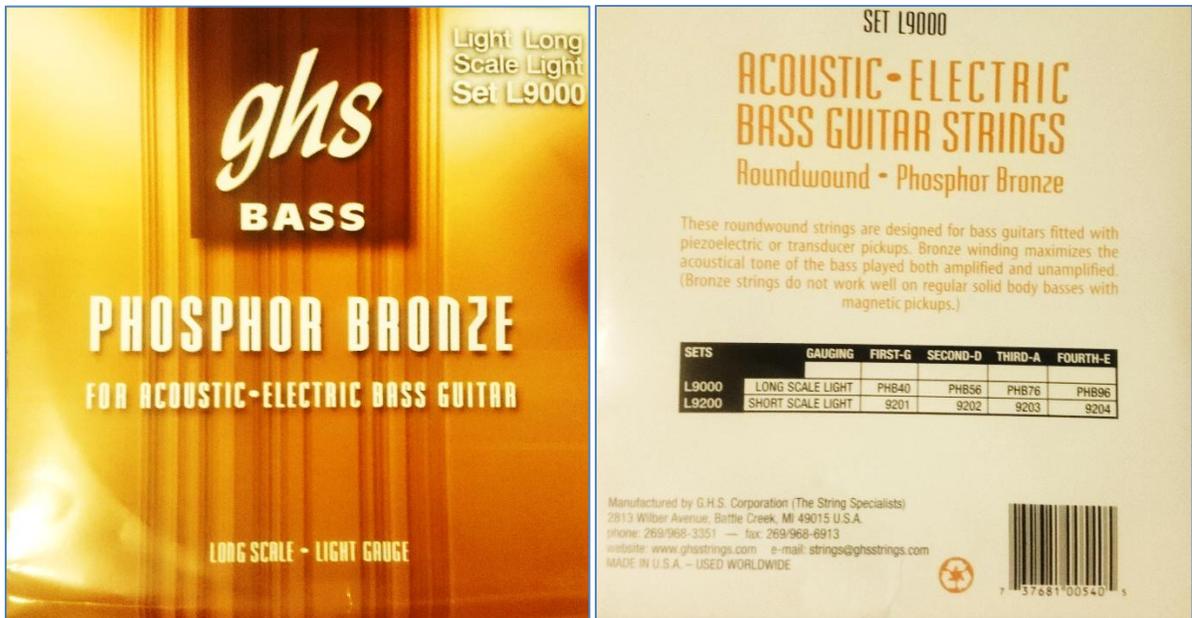
The Bandurria is the smallest and highest pitched member in the laud family. Mandolin-like, this tear-dropped-shaped cittern has a very short and wide neck. Over the past 200 years, the Bandurria's tunings and construction have evolved in the Spanish colonized regions of the world. Today, it has twelve steel strings in six unison courses and is popular in Spain, South America and the Philippines.

Tuning: G#3 G#3, C#4 C#4, F#4 F#4, B4 B4, E5 E5, A5 A5

- 12-STRING SET * 6 COURSES
- Plain Steel / Silver-Plated Wound
- Loop-Ends

1. 010P - 010P
2. 014P - 014P
3. 019W - 019W
4. 023W - 023W
5. 030W - 030W
6. 035W - 035W

7.16 Encordado para bajo eléctrico GHS L9000



ghs
BASS

Light Long Scale Light Set L9000

PHOSPHOR BRONZE
FOR ACOUSTIC-ELECTRIC BASS GUITAR

LONG SCALE - LIGHT GAUGE

SET L9000
ACOUSTIC-ELECTRIC BASS GUITAR STRINGS
Roundwound - Phosphor Bronze

These roundwound strings are designed for bass guitars fitted with piezoelectric or transducer pickups. Bronze winding maximizes the acoustical tone of the bass played both amplified and unamplified. (Bronze strings do not work well on regular solid body basses with magnetic pickups.)

| SETS | GAUGING | FIRST-G | SECOND-D | THIRD-A | FOURTH-E |
|-------|-------------------|---------|----------|---------|----------|
| L9000 | LONG SCALE LIGHT | PHB40 | PHB56 | PHB76 | PHB96 |
| L9200 | SHORT SCALE LIGHT | 9201 | 9202 | 9203 | 9204 |

Manufactured by G.H.S. Corporation (The String Specialists)
2813 Wilber Avenue, Battle Creek, MI 49015 U.S.A.
phone: 269/968-3351 — fax: 269/968-6913
website: www.ghstrings.com e-mail: strings@ghstrings.com
MADE IN U.S.A. — USED WORLDWIDE

7 37681 00540 5

7.17 Código final completo.

```
Memoria | Arduino 1.0.6
Archivo Editar Sketch Herramientas Ayuda
Memoria
1 #define LIN_OUT 1 //Activa la salida de la densidad espectral.
2 #define FHT_N 256 //Uso de 256 muestras para la Transformada.
3 //define WINDOW 0
4 #include <FHT.h> //Usar biblioteca FHT.
5 #include <LiquidCrystal.h> //Usar biblioteca LCD
6 #include <Servo.h>
7 Servo MOTOR; // Full rotational
8 int muestra=0; //Variable para almacenar lectura de entrafra analógica.
9 int ind=0; //Indice auxiliar para llenar el vector fht_input[]
10 boolean umbral=0; //Condicion para comenzar a llenar el vector fht_input[]
11 boolean lleno=0; //Variable que indica si ya se completó el llenado del vector fht_input[]
12 int Fs; //Variable que indica la tasa de muestreo
13 double df; //Variable que indica resolucion de los bins.
14 double F; //Variable que indica la frecuencia deseada.
15 LiquidCrystal lcd(12,11,5,4,3,2); //Inicializar pins para pantalla
16 int octava=3, nota=1;
17 int B1Actual, B1Anterior=HIGH, B2Actual, B2Anterior=HIGH, B1PIN, B2PIN; // variables para controlar botones
18 void setup()
19 {
20   pinMode(13, OUTPUT);
21   Serial.begin(9600);
22   lcd.begin(16, 2); //Pantalla de dos lineas de 16 caracteres
23   lcd.clear(); //limpiar pantalla
24   DisplayNota();
25   // Set up the pushbutton pins to be an input:
26   pinMode(B1PIN, INPUT);
27   pinMode(B2PIN, INPUT);
28 }
29 void loop(){
30   B1Actual = digitalRead(7); //Leer el estado del boton1. LOW es presionado, HIGH es sin presionar.
31   if(B1Actual!=B1Anterior){ //Si cambió su estado desde la ultima lectura.
32     if(B1Actual==HIGH){ //El cambio fue que se soltó el botón.
33       nota=nota+1; //Aumenta el valor de la nota deseada.
34       if(nota>12){ //Si supero un SI, entonces regresa al D0.
35         nota=1;
36         DisplayNota(); //Muestra la nota deseada en pantalla.
37       }
38       B1Anterior=B1Actual; //Actualiza el ultimo estado del boton1 registrado.
39     }
40     B2Actual = digitalRead(8); //Leer el estado del boton2.
41     if(B2Actual!=B2Anterior){ //Si cambió su estado desde la ultima lectura.
42       if(B2Actual==HIGH){ //El cambio fue que se soltó el botón.
43         octava=octava+1; //Aumenta el valor de la octava de nota deseada.
44         if(octava>5){ //Si se superó la quinta octava, volver a la primera.
45           octava=1;
```

```

Memoria | Arduino 1.0.6
Archivo Editar Sketch Herramientas Ayuda
Memoria
45     octava=1;
46     DisplayNota();//Muestra la nota desada en pantalla.
47 }
48 B2Anterior=B2Actual;//Actualiza el ultimo estado del boton2 registrado.
49 }
50 if(lleno){//Si el vector fht_input[] está listo para ser tratado.
51     int maxindex=FHT_N/4;//Indice en que se encuentra la mayor densidad espectral.
52     fht_reorder();//Ordena los datos para la FHT.
53     fht_run();//Ejecuta la FHT.
54     fht_mag_lin();//Calcula la densidad espectral
55     for(int i=maxindex;i<FHT_N/2;i++){//Recorre sólo la segunda mitad del arreglo. Ahí está la frecuencia que se busca.
56         if(fht_lin_out[i]>fht_lin_out[maxindex]){//La tasa de muestreo está ajustada de esa forma.
57             maxindex=i;//Si la densidad espectral es mayor, se guarda el nuevo índice.
58         }
59     }
60     double lectura = 1.0*(maxindex+0.5)*df;//Frecuencia obtenida. df es la resolución de cada bin.
61     double inferior = freq(nota-3,octava);//Frecuencia de 3 semitonos más grave que la nota deseada.
62     double superior=freq(nota+3,octava);//Frecuencia de 3 semitonos más agudo que la nota deseada.
63     double lec = log(lectura)/log(2);//Logaritmo en base2 de las últimas 3 variables calculadas
64     double inf = log(inferior)/log(2);//Esto se usa para linealizar la distancia en frecuencia entre cada nota.
65     double sup=log(superior)/log(2);
66     lleno=0;//Ya se usó el vector fht_input y está listo para ser completado nuevamente.
67     lcd.setCursor(9,0);//Se define la posición para escribir en la pantalla.
68     lcd.print(lectura);//Se muestra en pantalla la frecuencia obtenida.
69     if(lectura > inferior && lectura < superior){// Si la frecuencia detectada está dentro del rango
70         MOTOR.attach(9); //El motor recibe ordenes dese el PIN 9.
71         int giro = Escala(lec,inf,sup);//Transforma linealmente la diferencia.
72         Serial.print("Lectura = ");
73         Serial.print(lec);
74         Serial.print("  (");
75         Serial.print(inf);
76         Serial.print("  ");
77         Serial.print(sup);
78         Serial.print(")  Giro = ");
79         Serial.println(giro);
80         if(giro<90 && giro>80)
81             giro=80;
82         //if(giro>90 && giro <100)
83         //giro=105;
84         MOTOR.write(giro);//Da la orden de girar.
85     }
86     else{//Si la lectura está fuera de los límites
87         MOTOR.detach();//Se detiene el motor.
88     }
89 }
ISR(TIMERO_COMPA_vect){

```

```
Memoria | Arduino 1.0.6
Archivo Editar Sketch Herramientas Ayuda
Memoria
89 ISR(TIMERO_COMPA_vect){
90   muestra = analogRead(A0); //Se almacena muestra medida desde la entrada analógica.
91   if(muestra==0 || muestra==1023){ //Detector de saturación.
92     digitalWrite(13,HIGH);
93   } else{
94     digitalWrite(13,LOW);
95   } if(muestra<=420 || muestra>=1023-420){ //Umbral para llenar vector fht_input[]
96     umbral=1; //La cuerda ha sido pulsada.
97   } if(umbral && !lleno){ // Si la cuerda fue pulsada y fht_input[] no ha sido completado.
98     int margen = (int)0.3*Fs; //Margen para no capturar la pulsación de la cuerda.
99     if(ind>=margen){ //Dejar pasar 0.3[s] después de la pulsación de la cuerda.
100      fht_input[ind-margen]=muestra; //Se guarda la muestra.
101      ind++; //Aumenta el índice auxiliar de llenado de fht_input[]
102      if(ind-margen>=FHT_N) //Si ya se llenó el vector fht_input[]
103      {
104        ind=0; //Restauramos desde cero el índice auxiliar de llenado de fht_input[]
105        umbral=0; // Se espera la próxima pulsación de la cuerda.
106        lleno=1; //Boolean que indica que el vector fht_input[] ya ha sido completado.
107      }
108    }
109  }
110  float freq(int n, int o){
111    return 440.0*pow(2.0, (1.0*(o-4)*12+(n-10))/12.0);
112  }
113  void DisplayNota(){
114    if(octava==1 && nota<5){
115      nota=5;
116    } if(octava==5 && nota>10){
117      nota=1;
118    } String str;
119    switch(nota){
120      case 1:
121        str="D0 ";
122        break;
123      case 2:
124        str="D0# ";
125        break;
126      case 3:
127        str="RE ";
128        break;
129      case 4:
130        str="RE# ";
131        break;
132      case 5:
133        str="MI ";
134    }
```

```
Memoria | Arduino 1.0.6
Archivo Editar Sketch Herramientas Ayuda
Memoria
133     str="MI ";
134     break;
135 case 6:
136     str="FA ";
137     break;
138 case 7:
139     str="FA# ";
140     break;
141 case 8:
142     str="SOL ";
143     break;
144 case 9:
145     str="SOL#";
146     break;
147 case 10:
148     str="LA ";
149     break;
150 case 11:
151     str="LA# ";
152     break;
153 case 12:
154     str="SI ";
155     break;
156 }
157 lcd.setCursor(0,0);
158 lcd.print(str+octava);
159 F=freq(nota, octava);
160 lcd.setCursor(0,1);
161 lcd.print(" ");
162 lcd.setCursor(0,1);
163 lcd.print(F);
164 lcd.print("Hz");
165 SetSampleRate();
166 }
167 void SetSampleRate(){ //Ajusta el reloj timer0
168 cli(); //Detener los interrupts
169 TCCR0A = 0; // set entire TCCR0A register to 0
170 TCCR0B = 0; // same for TCCR0B
171 TCNT0 = 0; //Inicializar contador desde 0
172 int tasas[19] = {100, 200, 300, 400, 500, 600, 702, 801, 905, 1000, 1101, 1201, 1302, 1404, 1506, 1602, 1700, 1811, 1908};
173 for(int i=0; i<18; i++){ //Recorre el vector con los valores de las posibles tasas de muestreo.
174     if(F*2<tasas[i]){ //Busca el primer valor que sea mayor a dos veces la frecuencia deseada.
175         Fs=tasas[i]; //Se establece ese valor como tasa de muestreo.
176         break;
177     }

```

```
Memoria | Arduino 1.0.6
Archivo Editar Sketch Herramientas Ayuda
Memoria
177     }
178   }
179   switch(Fs){//Se ajusta el reloj del timer0 a la nueva tasa de muestreo.
180     case 100:
181       OCR0A = 155;// = (16*10^6) / (100*16025) - 1 (debes ser <256)
182       TCCR0B |= (1 << CS02) | (1 << CS00);//Activar bit CS00 y CS02 para prescaler=1024
183       break;
184     case 200:
185       OCR0A = 77;// = (16*10^6) / (200.320512*1024) - 1 (debes ser <256)
186       TCCR0B |= (1 << CS02) | (1 << CS00);//Activar bit CS00 y CS02 para prescaler=1024
187       break;
188     case 300:
189       OCR0A = 207;// = (16*10^6) / (300.480769*256) - 1 (debes ser <256)
190       TCCR0B |= (1 << CS02);//Activar bit S02 para prescaler=256
191       break;
192     case 400:
193       OCR0A = 155;// = (16*10^6) / (400.6410256*256) - 1 (debes ser <256)
194       TCCR0B |= (1 << CS02);//Activar bit S02 para prescaler=256
195       break;
196     case 500:
197       OCR0A = 124;// = (16*10^6) / (500*256) - 1 (debes ser <256)
198       TCCR0B |= (1 << CS02);//Activar bit S02 para prescaler=256
199       break;
200     case 600:
201       OCR0A = 103;// = (16*10^6) / (600.9615384*256) - 1 (debes ser <256)
202       TCCR0B |= (1 << CS02);//Activar bit S02 para prescaler=256
203       break;
204     case 702:
205       OCR0A = 88;// = (16*10^6) / (702.247191*256) - 1 (debes ser <256)
206       TCCR0B |= (1 << CS02);//Activar bit S02 para prescaler=256
207       break;
208     case 801:
209       OCR0A = 77;// = (16*10^6) / (801.282051*256) - 1 (debes ser <256)
210       TCCR0B |= (1 << CS02);//Activar bit S02 para prescaler=256
211       break;
212     case 905:
213       OCR0A = 68;// = (16*10^6) / (905.79710144*256) - 1 (debes ser <256)
214       TCCR0B |= (1 << CS02);//Activar bit S02 para prescaler=256
215       break;
216     case 1000:
217       OCR0A = 249;// = (16*10^6) / (1000*64) - 1 (debes ser <256)
218       TCCR0B |= (1 << CS01) | (1 << CS00);//Activar bit CS01 y CS00 para prescaler=64
219       break;
220     case 1101:
221       OCR0A = 226;// = (16*10^6) / (1101.32158590*64) - 1 (debes ser <256)
```

```

Memoria | Arduino 1.0.6
Archivo Editar Sketch Herramientas Ayuda
Memoria §
221     OCR0A = 226; // = (16*10^6) / (1101.32158590*64) - 1 (debes ser <256)
222     TCCR0B |= (1 << CS01) | (1 << CS00); //Activar bit CS01 y CS00 para prescaler=64
223     break;
224     case 1201:
225     OCR0A = 207; // = (16*10^6) / (1201.923076923*64) - 1 (debes ser <256)
226     TCCR0B |= (1 << CS01) | (1 << CS00); //Activar bit CS01 y CS00 para prescaler=64
227     break;
228     case 1302:
229     OCR0A = 191; // = (16*10^6) / (1302.08333*64) - 1 (debes ser <256)
230     TCCR0B |= (1 << CS01) | (1 << CS00); //Activar bit CS01 y CS00 para prescaler=64
231     break;
232     case 1404:
233     OCR0A = 177; // = (16*10^6) / (1404.494382022*64) - 1 (debes ser <256)
234     TCCR0B |= (1 << CS01) | (1 << CS00); //Activar bit CS01 y CS00 para prescaler=64
235     break;
236     case 1506:
237     OCR0A = 165; // = (16*10^6) / (1506.02409638*64) - 1 (debes ser <256)
238     TCCR0B |= (1 << CS01) | (1 << CS00); //Activar bit CS01 y CS00 para prescaler=64
239     break;
240     case 1602:
241     OCR0A = 155; // = (16*10^6) / (1602.56410256*64) - 1 (debes ser <256)
242     TCCR0B |= (1 << CS01) | (1 << CS00); //Activar bit CS01 y CS00 para prescaler=64
243     break;
244     case 1700:
245     OCR0A = 146; // = (16*10^6) / (1700.680272108*64) - 1 (debes ser <256)
246     TCCR0B |= (1 << CS01) | (1 << CS00); //Activar bit CS01 y CS00 para prescaler=64
247     break;
248     case 1811:
249     OCR0A = 137; // = (16*10^6) / (1811.5942*64) - 1 (debes ser <256)
250     TCCR0B |= (1 << CS01) | (1 << CS00); //Activar bit CS01 y CS00 para prescaler=64
251     break;
252     case 1908:
253     OCR0A = 130; // = (16*10^6) / (1908.396946*64) - 1 (debes ser <256)
254     TCCR0B |= (1 << CS01) | (1 << CS00); //Activar bit CS01 y CS00 para prescaler=64
255     break;
256     }
257     df=1.0*Fs/FHT_N;
258     TCCR0A |= (1 << WGM01); //Activar on CTC mode
259     TIMSK0 |= (1 << OCIE0A); //Activar compare interrupt
260     sei(); //allow interrupts
261     lcd.setCursor(12,1);
262     lcd.print("    ");
263     lcd.setCursor(12,1);
264     lcd.print(Fs);
265 }

266 int Escala(double x,double a, double b){
267     return (int) ((x-a)*180/(b-a));
268 }

```