UNIVERSIDAD NACIONAL DE CHIMBORAZO

FACULTAD DE INGENIERÍA

ESCUELA DE INGENIERÍA ELECTRÓNICA Y TELECOMUNICACIONES

"TRABAJO DE GRADO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES"

TRABAJO DE GRADUACIÓN

"DISEÑO Y CONSTRUCCIÓN DE UN RECONOCEDOR DE FRECUENCIAS DE AUDIO PARA UN ROBOT BAILARÍN"

AUTOR:

DARWIN LUIS OROZCO PILCO

DIRECTOR:

ING. ALFONSO GUNSHA

RIOBAMBA - ECUADOR

AÑO 2016

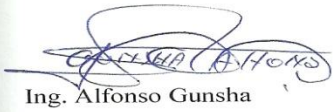
INFORME DEL TUTOR

INFORME DEL TUTOR

Yo Ing. ALFONSO JAVIER GUNSHA MORALES, en mi calidad de Tutor, del trabajo investigativo titulado: DISEÑO Y CONSTRUCCIÓN DE UN RECONOCEDOR DE FRECUENCIAS DE AUDIO PARA UN ROBOT BAILARÍN, tengo a bien informar que el mencionado trabajo cumple con los requisitos exigidos para ser expuestos al público, luego de ser evaluado por el tribunal designado.

Riobamba, Mayo del 2016

Atentamente



Ing. Alfonso Gunsha

TUTOR

i

Los miembros del Tribunal de graduación del proyecto de investigación: DISEÑO Y CONSTRUCCIÓN DE UN RECONOCEDOR DE FRECUENCIAS DE AUDIO PARA UN ROBOT BAILARÍN, presentado por Darwin Luis Orozco Pilco y dirigida por el Ing. Alfonso Gunsha.

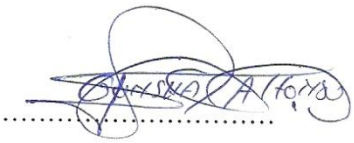
Una vez escuchada la defensa oral y revisado el informe final del proyecto de investigación con fines de graduación escrito en la cual se ha constado el cumplimiento de las observaciones realizadas, remite la presente por uso y custodia en la biblioteca de la Facultad de Ingeniería de la Universidad Nacional De Chimborazo.

Para constancia de lo expuesto firman:

Ing. Paulina Vélez
Presidenta del Tribunal

Firma

Ing. Alfonso Gunsha
Director del Proyecto



Firma

Ing. Geovanny Cuzco
Miembro del Tribunal

Firma

AUTORÍA DE LA INVESTIGACIÓN

AUTORÍA DE LA INVESTIGACIÓN

"La responsabilidad del contenido de este Proyecto de Graduación, corresponde exclusivamente a Darwin Luis Orozco Pilco; y el patrimonio de la misma a la Universidad Nacional de Chimborazo."

Autor:

.....

Darwin Luis Orozco Pilco

C.I 060409504-2

iii

AGRADECIMIENTO

Agradezco de la manera más sincera a mis padres que siempre me apoyaron en la buenas y malas, a la Universidad Nacional de Chimborazo, Tutor de tesis, amigos, familiares, a mi hermano Paul que hicieron posible la culminación del tema de investigación, a los profesores de la Universidad Nacional de Chimborazo por bridarme sus conocimientos, su guía para mi formación académica.

A Ximena Macas por sus consejos, su apoyo incondicional por sus recomendaciones por su tiempo y apoyo en la realización del proyecto.

Darwin Orozco

DEDICATORIA

El presente trabajo de investigación lo dedico a Dios por darme bendiciones a mi vida, a mis padres Gloria y Luis por su apoyo y consejos han sabido guiar mi vida; a mis hermanos: Paul, Katherine, Henrry.

Una dedicatoria muy especial a mi padre Luis que se encuentra en el cielo el cual siempre fue mi apoyo en las buenas y en las malas, a Ximena que fue mi inspiración para la culminación de mi proyecto de investigación.

Darwin Orozco

ÍNDICE

CAPITULO II

METODOLOGÍA

CAPITULO III

RESULTADOS

CAPITULO IV

CONCLUSIONES Y RECOMENDACIONES

ÍNDICE DE GRÁFICOS

ÍNDICE DE TABLAS

RESUMEN

El presente proyecto de investigación detalla el diseño y construcción de un reconocedor de frecuencias de audio para un robot bailarín demostrando diferentes aplicaciones en el área de la robótica y electrónica.

El PIC 16F877A es un micro controlador de la familia microchip su consumo de potencia es muy bajo es completamente estático, los datos de la memoria no se pierden puede ser programado para trabajar en la recepción y transmisión de datos; los comandos de programación se describen en el presente documento.

Con los estudiantes que pertenecen al grupo de investigación de la carrera de Electrónica y Telecomunicaciones se experimentó el control del robot bailarín al son de la música, demostrando así la eficiencia en el control de los diferentes movimientos realizados por el robot.

En el micro controlador se realizó la programación de un algoritmo en Mikro C el cual me permite analizar la señal de audio de entrada, detecta los diferentes tonos de audio (graves, medios, agudos) los datos obtenidos del algoritmo son transmitidos al robot mediante la comunicación UART el cual me permite activar o desactivar los movimientos del robot al son de la música.

SUMMARY

UNIVERSIDAD NACIONAL DE CHIMBORAZO

CENTRO DE IDIOMAS

Lic. Lorena Gallegos 20 de Mayo de 2016

SUMMARY

This research explains in detail the design and construction of a audio frequency recognizer for a dancer robot demonstrating different applications in the robotics and electronics area.

The PIC 16F877A is a micro controller family of microchip, its power consumption is very low is completely static, the memory data is not lost can be programmed to work in the reception and transmission of data; programming commands are described in the present document.

The students who belong to the research group of Electronics and Telecommunications major, control robot dancer music was qualified, demonstrating efficiency in controlling the various movements made by the robot.

In the micro controller programming an algorithm in Mikro C was developed which allows to analyze the audio signal input, detects the different audio tones (bass, middle, treble) the obtained data in the algorithm are transmitted to the robot through communication UART which allows to enable or disable the robot's movements to the sound of music.

CENTRO DE IDIOMAS

xiii

INTRODUCCIÓN

La robótica está determinada como una ciencia aplicada que nace de la mezcla de la tecnología con las maquinas- herramientas que estudia el diseño y construcción de máquinas capaces de desempeñar distintas tareas esto permite que un programa informático controle las operaciones que antes realizaba un obrero, básicamente la robótica se ocupa de todo en lo referente a los robots, lo cual contiene el control de motores, mecanismos automáticos, sensores etc.

La robótica ha creado cursos estudiantiles de robótica donde haciendo gala de la electrónica pueden seguir una línea, evadir obstáculos por medio de visión artificial, bailar, luchar entre ellos etc. Actualmente la ciencia ha puesto toda la atención en desarrollar robots capaces de emular movimientos de los seres vivos.

Los robots bípedos han cautivado la mayor atención de la sociedad científica y tecnológica por los retos que conlleva es fundamental analizar los movimientos del robot.

El presente trabajo de investigación se ejecuta para que el robot realice movimientos de forma autónoma se ha investigado y desarrollado el presente trabajo de manera práctica y sencilla guiándonos así en el concepto y características de un algoritmo, el robot tiene la capacidad de emular los movimientos de un bailarín y moverse con la autonomía al son de un ritmo musical, en la investigación dejo información muy detallada del programa realizado que servirá como complemento de estudio aplicado a la robótica.

CAPITULO I

1 FUNDAMENTACIÓN TEÓRICA

1.1 FILTRO

El filtro es un elemento que diferencia una determinada frecuencia o gama de frecuencias de una determinada señal que cruza por ella consiguiendo cambiar su fase y su amplitud.

Las características que concretan un filtro vienen determinadas por los siguientes parámetros.

1.1.1 FUNCIÓN DE TRANSFERENCIA

La forma de comportamiento de un filtro (analógico, digital, mecánico) se representa por su función de transferencia, se establece la forma en que la señal aplicada cambia en fase y amplitud pasar por el filtro.

Filtro Butterworth: Posee un corte agudo y un paso de banda suave.

Filtro Chevyshev: Paso de banda con ondulaciones y un corte agudo.

Filtro Bessel: En caso de ser analógico asegura una diferenciación de fase firme o constante.

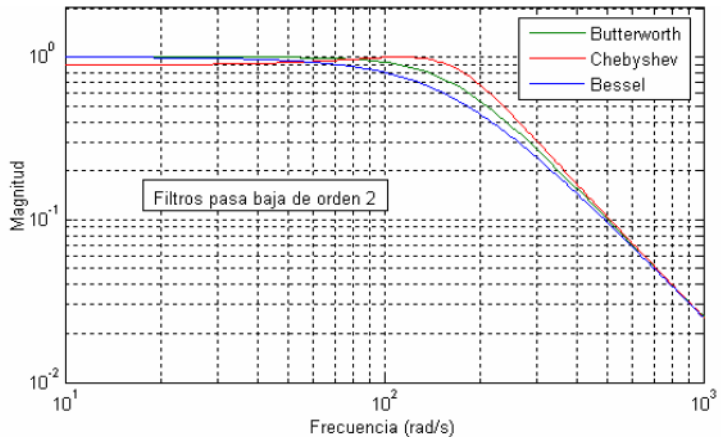


Figura 1: Filtros
Fuente:http://www2.uca.es/grupinvest/instrument_electro/ppjjgdr/Cir_An_Apl/Cir_An_Apl_arch/temas/T4_caa.pdf

1.1.2 ORDEN

Un filtro se representa de acuerdo a su orden describe el grado de aceptación o de rechazo de frecuencias por arriba o debajo de su respectiva frecuencia de corte. El filtro de primer orden se determina por su frecuencia de corte (fc) tiene que ser igual a (F). El filtro de segundo orden tiene el doble de pendiente (ilustrado en la escala algorítmica). (Moreno Velazco, 2012)

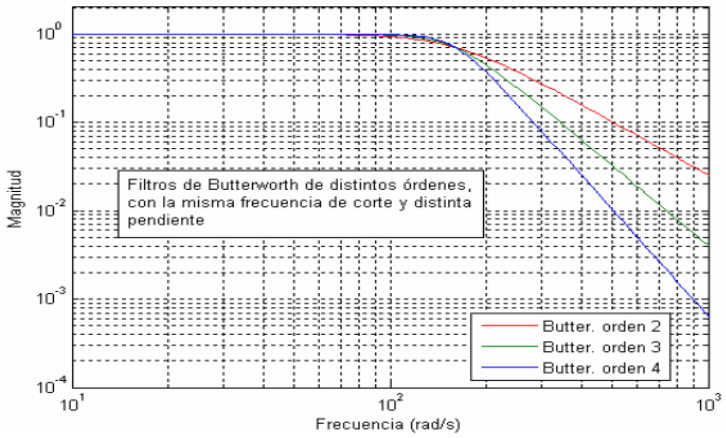


Figura 2: Orden de filtros
Fuente:http://ingenieria.udea.edu.co/~jfvargas/bk_old/Ctos2/documen/Capitulo5.pdf

1.2 FILTROS ACTIVOS

Un filtro activo es un convertidor basado en semiconductores y su principal objetivo es eliminar los armónicos de tensión y corriente; puede también realizar diversas funciones como son compensar el factor de potencia, compensar las corrientes de secuencia cero en sistemas balanceados, también compensa desbalances de tensión en varias aplicaciones, inclusive ayuda a proveer energía a cargas críticas durante intervalos corto de tiempo.

Los filtros activos pueden ser del tipo serie, para así obstaculizar el paso de las corrientes armónicas como se ilustra en la figura (a). En la figura (b) se ilustra el tipo de derivación, para así comprimir el contenido de armónicos de la red. (Miraya, 2004)

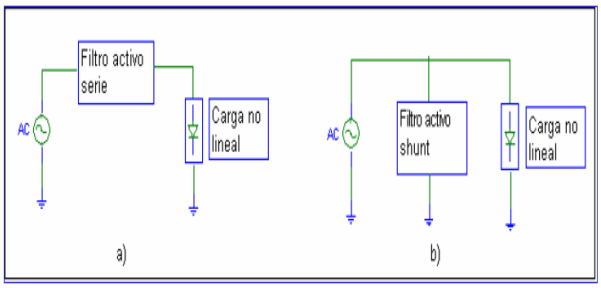


Figura 3: Filtros activos
Fuente: Autor

1.2.1 CLASES DE FILTROS ACTIVOS

- **Filtro pasa bajas**

  Permite solo el paso de frecuencias inferiores a una determinada frecuencia de corte (fc) atenuando las frecuencias superiores.

- **Filtro pasa altas**

  Permite pasar las frecuencias que se encuentran por arriba de una determinada frecuencia de corte (fc) resultando atenuadas las frecuencias inferiores.

- **Filtro pasa banda**

  Deja pasar solo las frecuencias situadas dentro de una banda delimitada por una (fc1) frecuencia de corte inferior y otra frecuencia de corte superior (fc2). Son atenuadas las frecuencias que se encuentran fuera de esta banda.

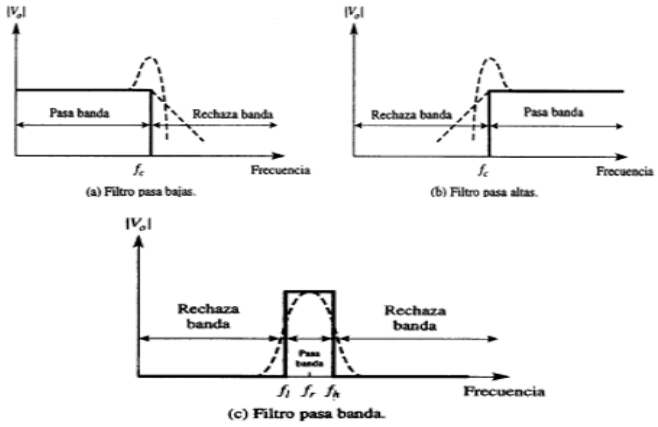


Figura 4: Tipos de filtros activos
Fuente: http://www2.ulpgc.es/hege/almacen/download/29/29861/filtros.pdf

1.3 FILTROS DIGITALES

Los filtros digitales son componentes básicos de todo sistema de procesamiento de señales, son sistemas discretos encargados de atenuar algunos componentes de frecuencia operan sobre una señal de entrada digital y efectúa un proceso de filtrado modificando el contenido espectral de una señal. (Gomez Gutièrrez, 2009-2010)

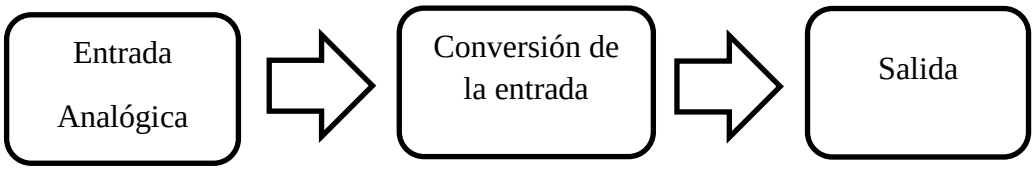


Figura 5: Diagrama de bloques
Fuente Autor

1.3.1 CLASES DE FILTROS DIGITALES

Existen varios tipos de filtros digitales y se clasifican de acuerdo a sus frecuencias.

- **Filtro pasa bajo**

  El filtro pasa bajo se caracteriza por dejar pasar frecuencias bajas en su banda de paso y rechazar frecuencias altas en la banda de rechazo. Las bandas quedan establecidas por:

  Banda de paso: Desde 0 hasta Wp (frecuencia de paso)

  Banda de rechazo: Desde Ws (frecuencia de corte) hasta ∞

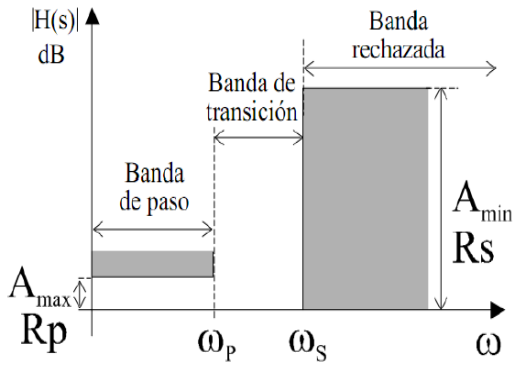


Figura 6: Características del filtro pasa bajos
Fuente: http://www.dtic.upf.edu/~jlozano/audio/edicion2.html

- **Filtro pasa alto**

  Permite el paso de señales arriba de su frecuencia de corte deja pasar frecuencias altas en su banda de paso y rechaza frecuencias bajas. Las bandas de frecuencia que describen al filtro pasa alto se representan por:

  Banda de paso: Desde Wp (frecuencia de paso)

  Banda de rechazo: Desde 0 hasta Ws (frecuencia de corte)

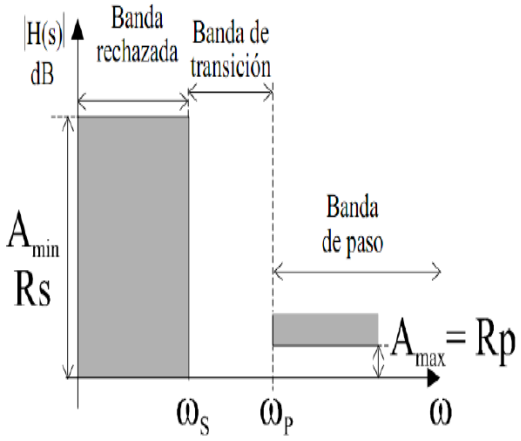


Figura 7: Características del filtro pasa altos
Fuente: http://www.dtic.upf.edu/~jlozano/audio/edicion2.html

- **Filtro pasa banda**

  Permite el paso de la señal si se encuentra arriba de la primera frecuencia de corte y por debajo de la segunda frecuencia de corte, hay presente dos bandas de rechazo una superior y otra inferior las cuales definen las bandas de paso solo deja pasar frecuencias que se encuentran dentro del rango. (Ojeda Guadamud, 2014)

  Banda de paso: Desde Wp1 (frecuencia de paso inferior) hasta Wp2 (frecuencia de paso superior).

  Banda rechazo inferior: Desde 0 hasta Ws1 (frecuencia de corte inferior)

  Banda de rechazo superior: Desde Ws2 (frecuencia de corte superior) hasta ∞.

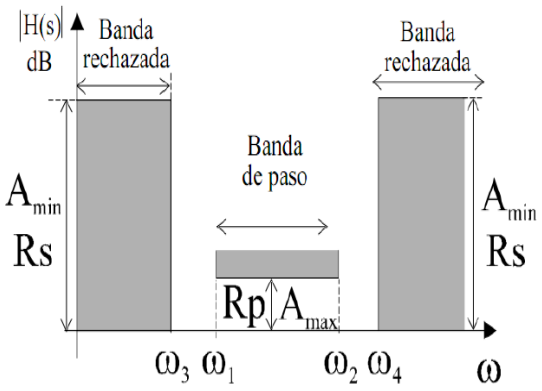


Figura 8: Características del filtro pasa bandas
Fuente: http://www.dtic.upf.edu/~jlozano/audio/edicion2.html

1.4 POR SU RESPUESTA IMPULSIONAL

1.4.1 FILTROS FIR (RESPUESTA IMPULSIONAL FINITA)

FIR (Respuesta finita al impulso) es un filtro digital también llamado no recursivo, si su entrada es un impulso su salida será un número limitado de términos no nulos. Para lograr la salida se emplean valores de la entrada actual y anterior. Su expresión en el dominio n se muestra en la **Ecuación 1**.

$$yn = \sum_{K=0}^{N-1} b_k x(n-k)$$

Ec. (1)

El orden del filtro está determinado por N (el número de coeficientes), la salida puede ser expresada como la construcción de una señal x(n) con un filtro h(n).

Características de un filtro FIR:

- El filtro FIR puede ser diseñado para tener fase lineal.
- Son estables porque son hechos únicamente con ceros.
- Tiene memoria finita.

- Es fácil de comprender e implementar.

Estructura de un filtro FIR.

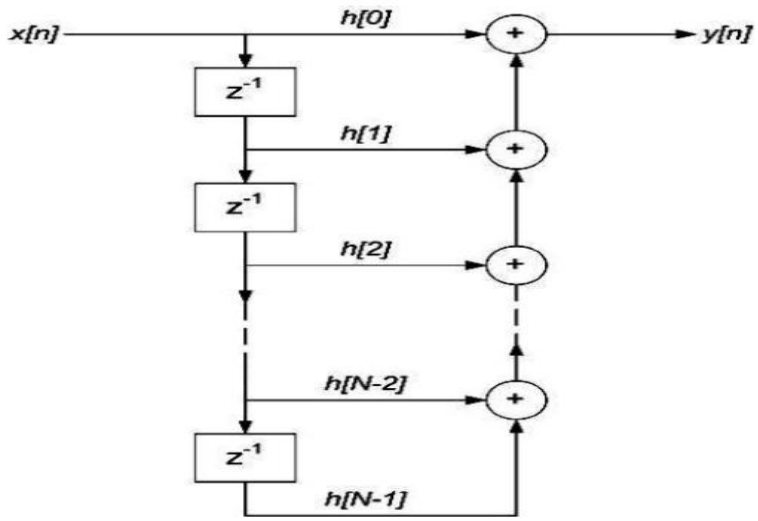


Figura 9: Estructura de un filtro FIR
Fuente: http://www.efn.uncor.edu/investigacion/DSP/documentos/Filtro_FIR.pdf

En la **Ecuación 2**, se refleja la función de transferencia de un filtro FIR.

$$H\ Z\ = \sum_{k=0}^{N-1} h_k z^{-k} = h_0 + h_1 z^{-1} + \cdots + h_{N-1} Z^{-(N-1)}$$

Ec. (2)

Los filtros FIR son estables ya que solo tienen elementos en el numerador en su función de transferencia, no introducen desfases en la señal. (Lara, Tapia, Gamboa, Narváez, & Parreño, 2013)

1.4.2 FILTROS IIR (RESPUESTA IMPULSIONAL INFINITA)

Los filtros IIR son llamados de respuesta infinita se caracteriza por tener una retroalimentación de la señal de salida, como la ecuación de diferencias depende de las salidas anteriores del filtro, existe una dependencia de los infinitos estados anteriores de la

variable de salida a la variable de salida actual, por tal razón son llamados de Respuesta al Impulso Infinita en la **Ecuación 3** observamos su representación. (M, 2010)

$$y\ n\ = \sum_{k=0}^{L-1} a\ k\ x\ n-k\ + \sum_{k=1}^{L-1} b\ k\ y[n-k]$$

Ec. (3)

La función de transferencia de un filtro IIR podemos apreciar en la **Ecuación 4**:

$$H\ z\ = \frac{\sum_{k=0}^{N-1} b_k z^{-k}}{1 + \sum_{k=0}^{M-1} a_k z^{-k}}$$

Ec. (4)

La implementación directa de los filtros IIR se obtiene a partir de su ecuación.

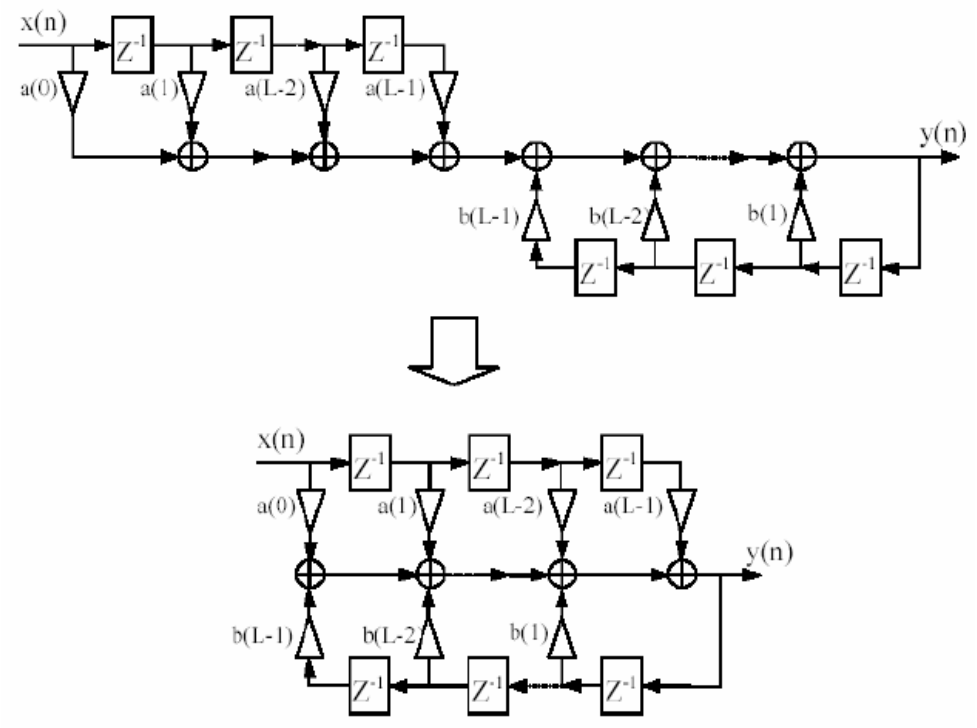


Figura 10: Estructura de un filtro IIR
Fuente:http://bibing.us.es/proyectos/abreproy/11375/fichero/MEMORIA%252FEstudio+filtro+iir.pdf

Observar que el filtro IIR puede considerarse como una cascada de dos subsistemas, numerador y denominador.

Características de los filtros IIR:

- Requieren menos memoria para implementar su función de transferencia.
- Este tipo de filtro no son estables.
- El uso de polos confiere a un filtro IIR la capacidad de implementar funciones de transferencia.
- Un filtro IIR produce distorsión en la fase.

1.5 FRECUENCIA DE AUDIO

La frecuencia de sonido se mide en (HZ), hace referencia a la cantidad de veces que vibra el aire que se transmite el sonido en un segundo la medición de onda puede comenzar en cualquier punto.

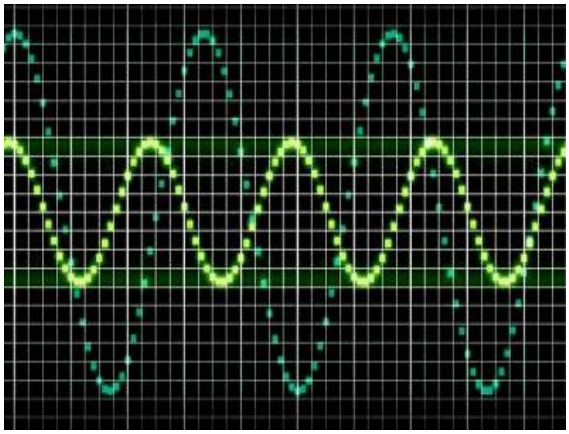


Figura 11: Ultrasonidos e infrasonidos
Fuente: http://www.fotonostra.com/digital/frecuenciaudio.htm

Se denominan ultrasónicos a las vibraciones de aire que oscilan un número de veces superior a 20.000 Hz.

Los infrasonidos son aquellos cuya frecuencia sonora está por debajo de los 20 Hz.

1.5.1 TONOS GRAVES, AGUDOS Y MEDIOS

La frecuencia de sonido está relacionada con la altura de la oscilación de la onda sonora, la altura del sonido es perceptible solo si la frecuencia de oscilación es la misma de un intervalo mínimo de tiempo. (Perez Vega, 2010)

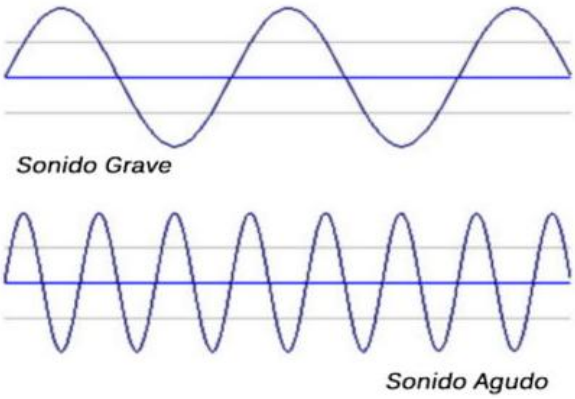


Figura 12: Tonos de audio
Fuente:http://www.insht.es/InshtWeb/Contenidos/Documentacion/Textos/GuiasMonitor/Ergonomia/IX/Ficheros/EIX02.pdf

- **Tonos agudos**

  Más conocidos como frecuencias altas que oscilan entre los 4k y 2k Hz.

- **Tonos medios**

  Son las frecuencias medias que oscilan entre los 500 y 2k Hz.

- **Tonos graves**

  Frecuencias bajas que oscilan entre 16 y 500 Hz.

1.6 COMUNICACIÓN UART

Asynchronous Receiver Transmitter más conocido como módulo UART, permite comunicación asíncrona full-dúplex con otros dispositivos o componentes tales como: computadoras, convertidores, módulos inalámbricos, etc. En la **figura 13** se muestra el

diagrama simplificado de UART, el cual consiste de los siguientes elementos principales. (Terven Salinas, 2013)

- Generador de Baudios
- Transmisor Asíncrono
- Receptor Asíncrono

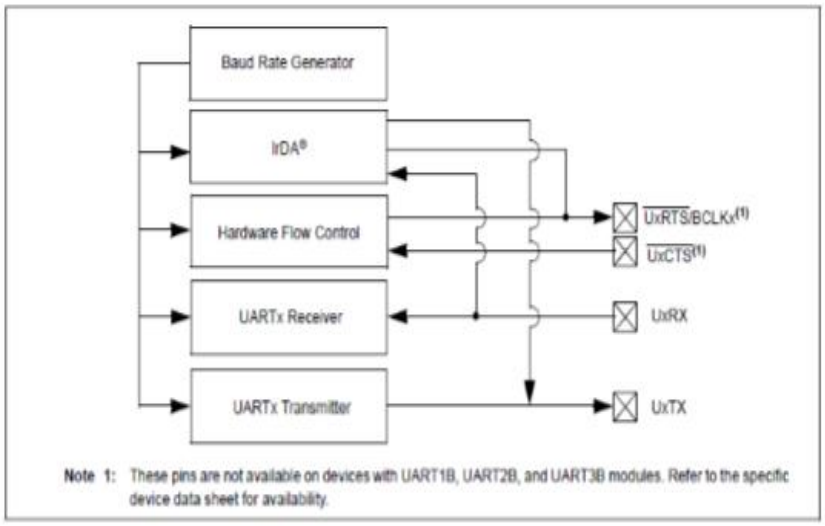


Figura 13: Diagrama de bloque
Fuente: http://www.tervenet.com/itmaz/micros2/PIC32_11_UART.pdf

Las características principales del módulo UART son:

- Interrupciones de transmisión y recepción.
- Transmisión Full-dúplex de 8- o 9-bits a través de los pines TX y RX.
- Opciones de paridad Par, Impar o Sin paridad para datos de 8 bits.
- Uno o dos bits de parada (Stop bits).
- Detección automática de la tasa de baudios.
- Tasas de transferencia de desde 76 bps hasta 20 Mbps a 80 MHz.
- Soporte para modo de 9 bits con detección de dirección (bit 9 en 1).

1.7 ALGORITMO

Conjunto de acciones o secuencias de operaciones ejecutadas en un determinado orden para resolver un problema. El algoritmo es independiente de los lenguajes de programación puede escribirse y luego ejecutarse en un lenguaje diferente de programación es una infraestructura que radica en desarrollar el razonamiento lógico, el algoritmo debe ser preciso y determinístico. (Joyanes Aguilar, 7 agosto 2015)

- **Preciso:** El algoritmo debe ejecutar la tarea para la cual fue diseñada
- **Determinístico:** El resultado debe depender estrictamente de los datos administrados.

1.7.1 CARACTERÍSTICAS DE UN ALGORITMO

Las principales características de un algoritmo son las siguientes:

- Debe ser preciso e indicar el orden de realización de cada paso.
- Debe ser definido, se debe obtener el mismo resultado cada vez.
- Se debe terminar en algún momento, debe tener un número finito de pasos.

1.7.2 ESTRUCTURA DE UN ALGORITMO

Un algoritmo se constituye de tres secciones principales.

Figura 14: Estructura de un algoritmo
Fuente: Autor

Entrada: Es la introducción de datos para ser transformados.

Proceso: Es el conjunto de operaciones para dar solución a un problema.

Salida: Son los resultados obtenidos a través del proceso.

1.8 MICROCONTROLADOR

El micro controlador es un circuito integrado sirve para controlar sistemas automáticos o procesos, mediante instrucciones especiales las cuales tienen la capacidad de ser programadas y ejecutar las instrucciones que son grabadas en su memoria con el mínimo error. (Canto Q., 2013)

Partes básicas de un micro controlador

- Líneas de entrada / salida (I/O)
- Lógica de control (Coordina la interacción entre los demás bloques)
- Memoria ROM (Memoria solo de lectura)
- Memoria RAM (Memoria de acceso aleatorio)

Componentes de un micro controlador

PWM	Modulación por ancho de pulso.
Comunicación Serial	Permite establecer comunicación con otro micro controlador o computadora a través de la interfaz serial RS- 232.
Interrupciones	Cuando se requiere atender eventos en tiempo real o se tiene procesos que no esperan.
Conversor Analógico Digital (A/D)	Permite medir señales no digitales o sensores que emitan señales analógicas.
Memoria EEPROM	Los datos no se alteran cuando se retira la alimentación, en un tipo de memoria ROM que se puede programar o borrar eléctricamente.

Tabla 1: Componentes de un micro controlador
Fuente: (Garre del Olmo, 2008)

En la **figura 15** podemos visualizar los componentes que forman parte de un micro controlador.

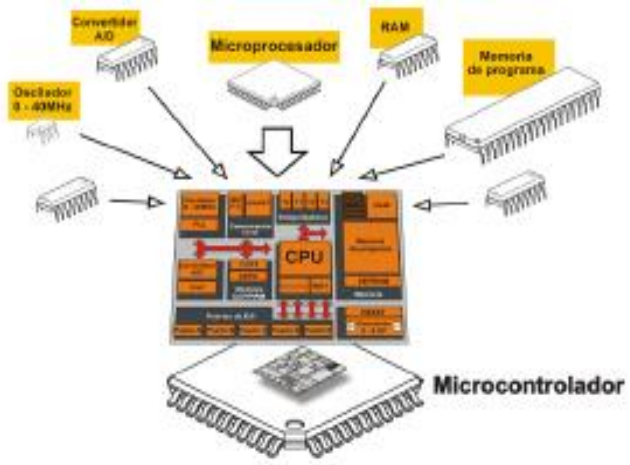


Figura 15: Micro controlador
Fuente: http://www.mikroe.com/chapters/view/79/capitulo-1-el-mundo-de-los-microcontroladores/

1.8.1 MICROCONTROLADOR 16F877A

Es un micro controlador de la familia microchip, dispone de todos sus componentes más importantes que la mayoría de micro controladores, en la **figura 16** podemos visualizar los pines del micro controlador.

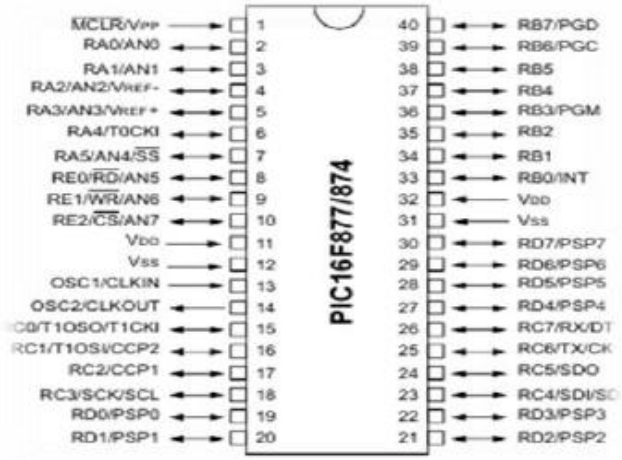


Figura 16: PIC 16F877A
Fuente. http://ww1.microchip.com/downloads/en/DeviceDoc/39582C.pdf

1.8.2 RECURSOS PERIFÉRICOS

TIMER0	Temporizador (Contador de 8 bits con prescaler de 8 bits).
TIMER1	Temporizador(contador de 8 bits con registro de periodo prescaler y post prescaler)
USART/SCI	Universal Synchronous Receiver Transmitter) con 9 bits.
PWM	Dos módulos de captura, comparación.
SSP,SPI,I2C	Puerto Serie Síncrono (SSP) CON SPI (Modo maestro) e I2C (Master/Slave).

Tabla 2: Recursos periféricos
Fuente. (Microchip, 2001-2013)

1.9 MIKRO C

Mikro C "Es un paquete de software con una amplia variedad de ayudas y herramientas que facilita la creación de proyectos aplicativos para micro controladores de la familia 12F, 16F, 18F es un lenguaje compilador se basa en programar un código de máquina que contiene una a una las instrucciones del programa". (Clavijo Mendosa, 2011)

En la **figura 17** podemos apreciar la apariencia visual de Mikro C.

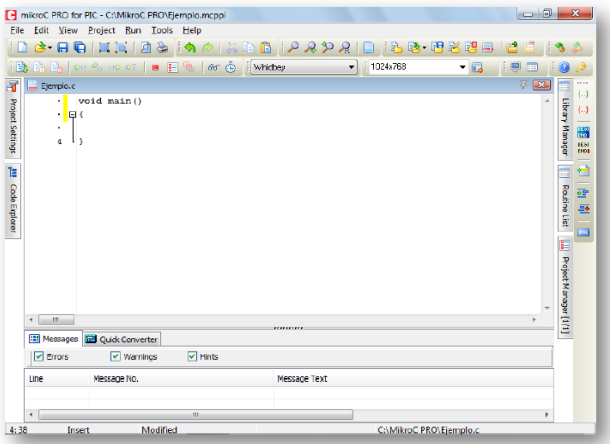


Figura 17: Mikro C
Fuente: Autor

El compilador de alto nivel en lenguaje C utiliza estructuras que facilitan la programación, optimiza las operaciones matemáticas y los procesos por medio del uso de funciones definidas, así como también el uso de un conjunto de variables, de diferentes tipos como: carácter, entero, punto decimal.

"El compilador crea automáticamente el código ensamblador y a su vez un código similar consignado a un archivo con extensión *.hex, este archivo programa eléctricamente al micro controlador, también se puede realizar una simulación computacional". (Clavijo Mendosa, 2011)

En la tabla 3 se puede observar las características de las variables en Mikro C.

Tipo de variable	**Tamaño en Bytes**	**Valores que soporta**
bit	1	0 ó 1
char	1	-127 a 127
short	1	-127 a 127
int	2	-32767 a 32767
long	4	-2147483647 a 2147483647
float	4	-1.5x10^45 a 3.4x10^38
double	4	-1.5x10^45 a 3.4x10
unsigned char	1	0 a 255
unsigned short	1	0 a 255
unsigned int	2	0 a 65535
unsigned long	4	0 a 4294967295

Tabla 3: Variables en Mikro C
Fuente: (Clavijo Mendosa, 2011)

CAPITULO II

2 METODOLOGÍA

2.1 TIPO DE ESTUDIO

El presente estudio es orientado a la línea de investigación del detector de frecuencias de audio, usados en el ámbito de la electrónica y robótica.

Los tipos de estudio empleados en el presente proyecto de investigación son:

Cuantitativo: En la investigación del proyecto se realizó el armado del robot con los estudiantes sin el uso del detector y posteriormente usando el detector de frecuencias de audio para determinar si el robot cubre las expectativas de los estudiantes.

Método analítico: El método del estudio se fundamenta en el análisis de los dispositivos que conforman los circuitos existentes en el detector de frecuencias de audio para realizar los diferentes movimientos de robot.

Experimental: Se relaciona a los experimentos y aplicaciones desarrolladas con el enfoque científico para comprobar la comunicación entre el micro controlador y el robot, usados para las prácticas y aplicaciones con el detector de frecuencias de audio.

Observación: Aplicado a la práctica y utilización de los diferentes circuitos utilizados en el diseño, para caracterizar el modelo del detector con sus distintos dispositivos.

2.2 POBLACIÓN Y MUESTRA

El proyecto se implementará para los grupos de investigación de Electrónica y Telecomunicaciones de la Universidad Nacional de Chimborazo estará disponible para los estudiantes de la Escuela y para aquellas personas que trabajan en robótica.

2.3 OPERACIONALIZACIÓN DE VARIABLES

En la **tabla 4** se muestra la operacionalización de variables.

VARIABLE	CONCEPTO	CATEGORÍA	INDICADOR	TÉCNICAS DE INSTRUMENTO
VARIABLE INDEPENDIENTE: El diseño y construcción de un reconocedor de frecuencias de audio	**Frecuencias de audio** El reconocedor de frecuencias permite implementar fácilmente prácticas con la robótica. Es fácilmente transportable, está construido en una placa electrónica la cual conecta todos los dispositivos electrónicos.	Desarrollo de prácticas con robots.	1. Aplicación de la práctica. 2. Investigación y desarrollo con la robótica. 3.Experimentación practica	Manual de prácticas.
VARIABLE DEPENDIENTE: Robot bailarín	**Robótica** Ciencia y técnica que está involucrada en el diseño, fabricación de robots. Aprendizaje practico, mediante el cual los estudiantes aplican lo aprendido en clases en el lugar de trabajo, en entornos de investigación.	-Comunicación UART entre el PIC y el robot. -Aprendizaje significativo en la práctica.	4. Aplicaciones de comunicación UART. 5.Compresion y entendimiento	-Observaciones -Análisis del robot.

Tabla 4: Operacionalización de variables
Elaborado: Autor

2.4 PROCEDIMIENTOS

2.4.1 REALIZACIÓN DE UN FILTRO DIGITAL FIR EN MATLAB

Filtro FIR

Es un filtro digital que, si su entrada es un impulso la salida será un número limitado de términos no nulos. Para obtener la salida solo se emplean valores de las entradas actuales y anteriores.

El filtro está realizado en Matlab para extraer los tonos altos, medios y bajos de audio. Como primer paso se realizó el procesamiento de la señal se puede observar en la **figura 18**.

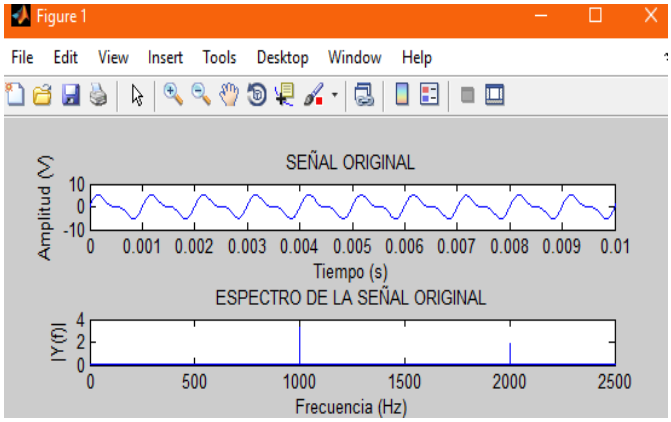


Figura 18: Procesamiento de la señal
Fuente: Autor

Una vez procesada la señal como segundo paso se procedió a extraer el audio se lo puede apreciar en la **figura 19**.

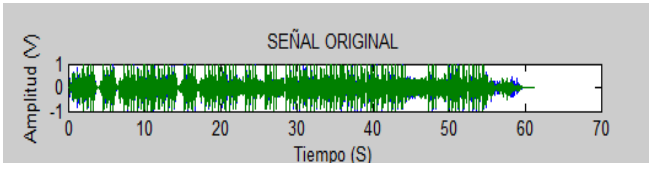


Figura 19: Procesamiento de audio
Fuente: Autor

En la **figura 20** se puede observar el filtrado de la señal de audio ingresada.

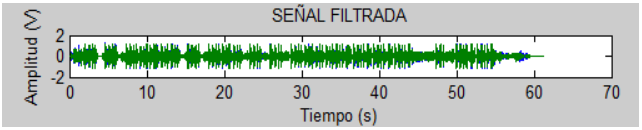


Figura 20: Señal filtrada
Fuente: Autor

Una vez filtrada la señal de audio procedemos a aplicar el filtro para extraer los tonos bajos, medios y altos como se ilustra la f**igura 21**.

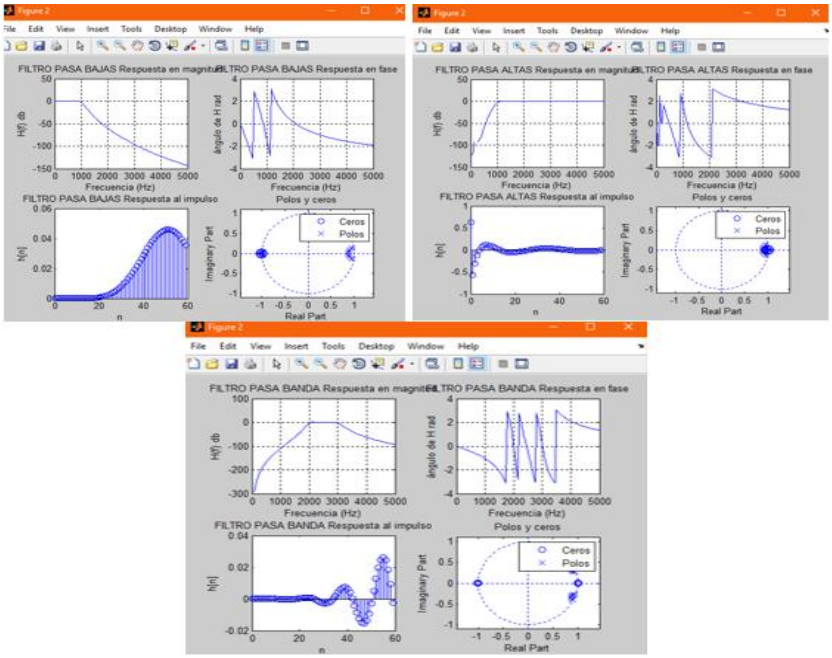


Figura 21: Extracción de tonos
Fuente: Autor

El código completo del filtrado de la señal de audio para extraer los tonos altos, medios y bajos se lo puede ver en el **ANEXO A.**

2.4.2 DISEÑO Y PROGRAMACIÓN DEL ALGORITMO

ALGORITMO

Conjunto de acciones o secuencias de operaciones ejecutadas en un determinado orden para resolver un problema.

2.4.2.1 DISEÑO DEL ALGORITMO

En la **figura 22** podemos apreciar el algoritmo diseñado para la extracción de los diferentes tonos de audio.

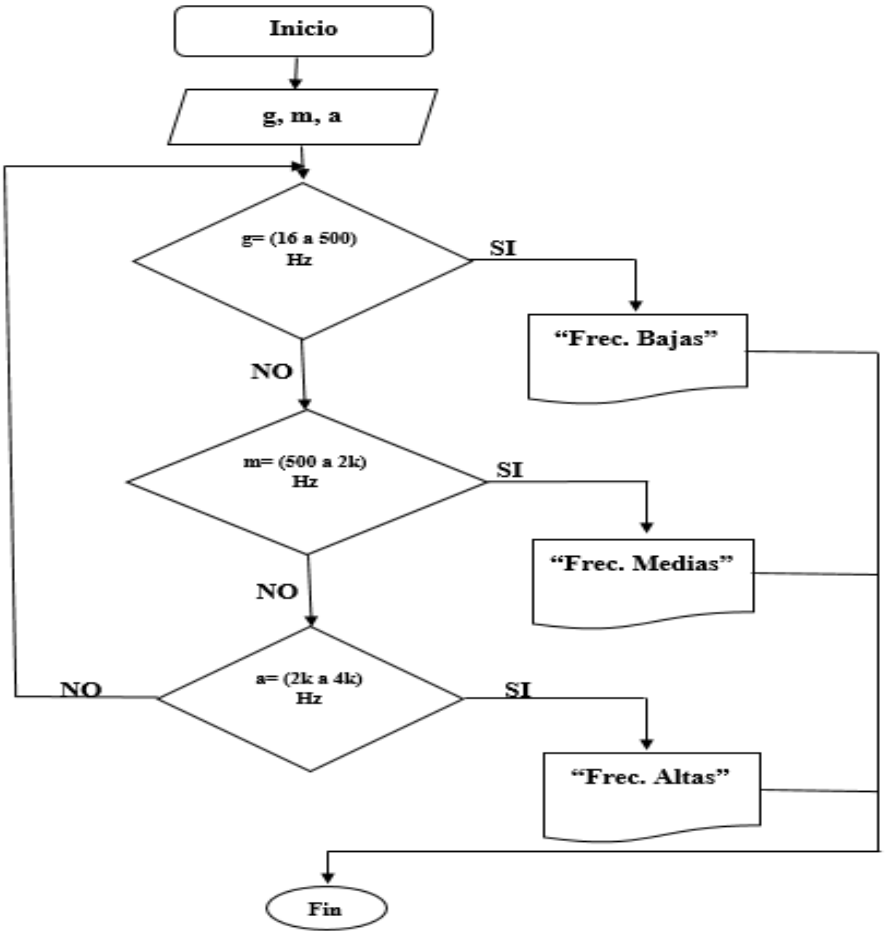


Figura 22: Diseño del algoritmo
Fuente: Autor

En **la tabla 5** podemos observar los diferentes tonos de audio con su respectiva frecuencia.

TONOS DE AUDIO	**FRECUENCIAS**
Graves	(16 a 500) Hz
Medios	(500 a 2k) Hz
Agudos	(2k a 4k) Hz

Tabla 5: Frecuencias de audio
Elaborado: Autor

2.4.2.2 CÁLCULOS DEL ALGORITMO

Aquí realizáramos todos los cálculos necesarios para ingresar los datos del algoritmo en Mikro c, en la **tabla 5** visualizamos las frecuencias máximas y mínimas.

4k Hz = 5V (Picos altos más significativos)

16 Hz = 0V (Picos bajos menos significativos)

Una vez obtenido las frecuencias máximas y mínimas procedemos a realizar el cálculo de los voltajes dependiendo de los diferentes tonos de audio como se observa en la **tabla 6**.

Rango Hz = (4,000 - 16) Hz

Rango Hz = 3,981 Hz

3,981Hz = 5V

X (1V) = ?

$$X\ 1V\ = \frac{3{,}981\ Hz * 1V}{5V}$$

X (1V) = 749.6 Hz

TONOS DE AUDIO	**RANGOS DE FRECUENCIA**	**RANGOS DE VOLTAJE**
Graves	(16 a 500) Hz	(0 a 0.8) V
Medios	(500 a 2k) Hz	(0.8 a 2.3) V
Agudos	(2k a 4k) Hz	(2.3 a 5) V

Tabla 6: Rangos de voltaje
Elaborado: Autor

Una vez obtenido los rangos de voltaje de acuerdo a cada tono musical procedemos a calcular los rangos del algoritmo para poder realizar el programa en Mikro C como se puede visualizar en la **tabla 7.**

5V = 660 (Frecuencia máxima)

0V = 60 (Frecuencia mínima)

Rango = (660 – 60)

Rango = 600

$$X\ \ 0.8V\ \ = \frac{600 * 0.8\ V}{5V}$$

X (0.8V) = 96 + 60

X (0.8V) = 156

TONOS DE AUDIO	**RANGOS DE FRECUENCIA Y VOLTAJE**	**BIT**	**RANGOS DE LECTURA DEL ADC**
Graves	(16 a 500) Hz (0 a 0.8) V	Primer bit Segundo bit	**(**60 a 108) (108 a 156)
Medios	(500 a 2k) Hz (0.8 a 2.3) V	Tercer bit Cuarto bit Quinto bit	(156 a 207) (207 a 259) (259 a 310)
Agudos	(2k a 4k) Hz (2.3 a 5) V	Sexto bit Séptimo bit Octavo bit	(310 a 423) (423 a 536) (536 a 660)

Tabla 7: Rangos de lectura del ADC
Elaborado: Autor

2.4.2.3 PROGRAMACIÓN DEL ALGORITMO EN MIKRO C

El algoritmo esta realizado en el lenguaje de programación llamado Mikro C es un lenguaje que facilita la creación de proyectos aplicativos para micro controladores de la familia 12F, 16F, 18F.

Procedemos a crear el programa en Mikro C como se observa en la **figura 23**.

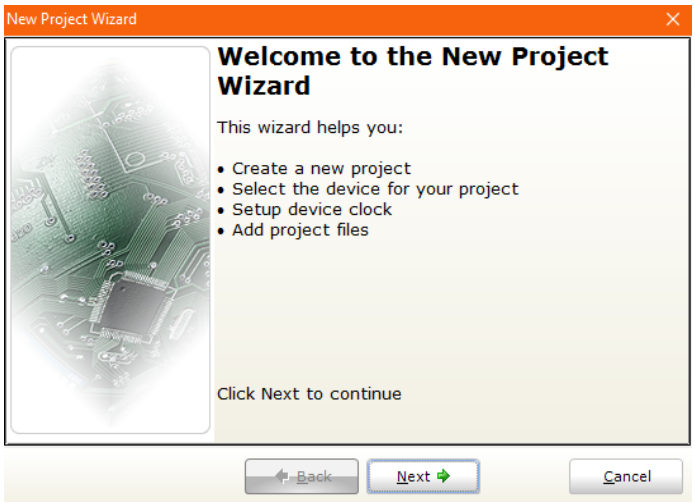


Figura 23: Creación del new Project
Fuente: Autor

En la **figura 24** visualizamos la configuración del Pic a utilizar.

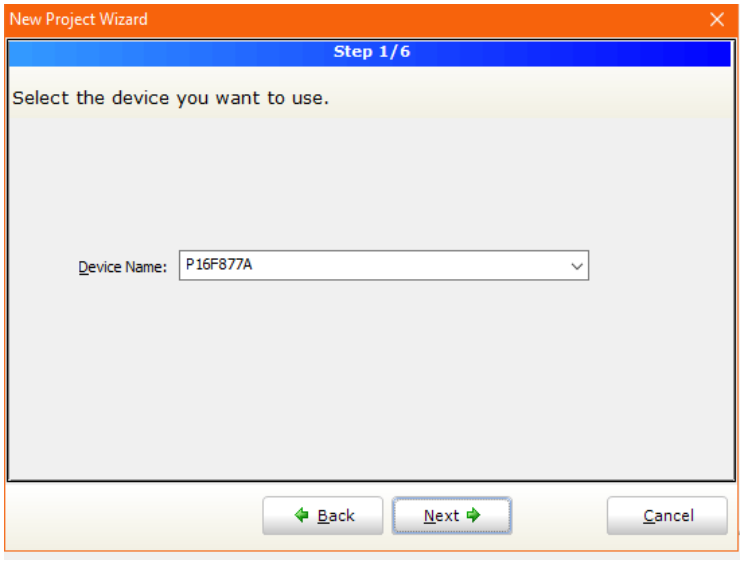


Figura 24: Pic a utilizar
Fuente: Autor

En la **figura 25** apreciamos la configuración del cristal que se va a utilizar.

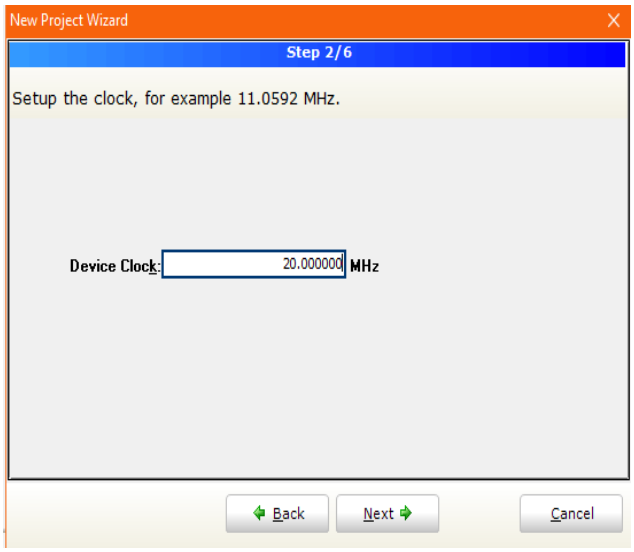


Figura 25: Configuración del cristal
Fuente: Autor

A continuación, procedemos a realizar la programación del algoritmo, efectuamos la configuración para la lectura del ADC el cual me lee la señal de audio de ingreso como se observa en la **figura 26**.

La configuración del ADC realizada se lo puede visualizar en el **Anexo B**.

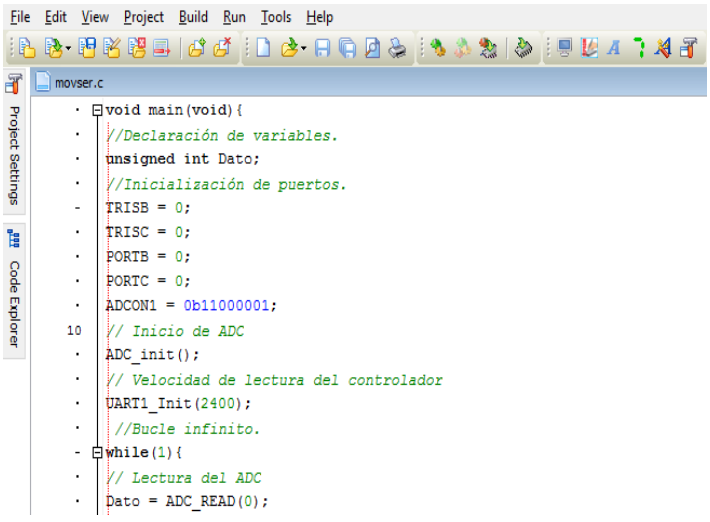


Figura 26: Lectura del ADC
Fuente: Autor

En la **figura 27** podemos visualizar como realiza el proceso el algoritmo para detectar los diferentes tonos de audio en la **tabla 7** se puede observar los datos que utilizamos.

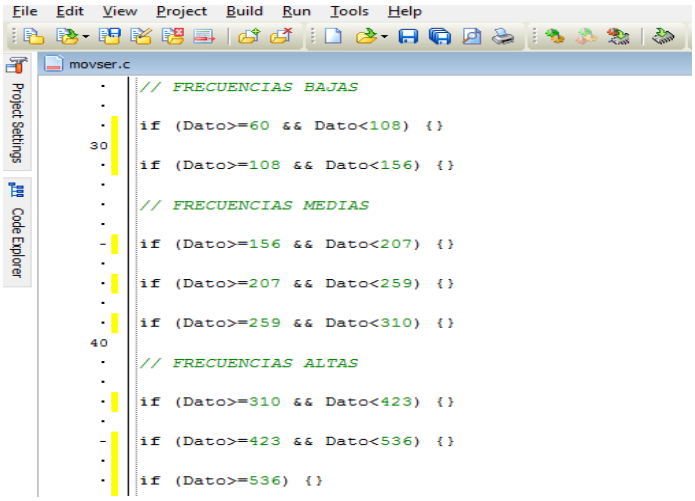


Figura 27: Programación del algoritmo
Fuente: Autor

Una vez ingresado los datos del algoritmo procedemos a configurar la comunicación UART para así poder obtener una comunicación entre el PIC y el robot se lo puede observar en la **figura 28**, la configuración de la comunicación UART se lo puede observar en el **Anexo B**.

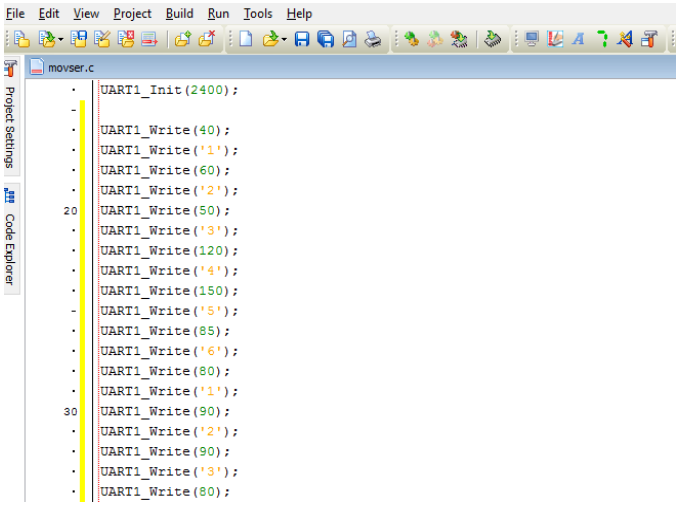


Figura 28: Programación de la Comunicación UART
Fuente: Autor

Por último Compilamos el programa, este tipo de lenguaje de programación me genera un código con extensión *.hex, este código se lo grava eléctricamente al micro controlador para

realizar la grabación del micro controlador se utiliza un dispositivo llamado quemador de PIC en la **figura 29** se puede observar cómo se grava el programa en el micro controlador.

Figura 29: Quemador de PIC
Fuente: Autor

El código del algoritmo está realizado en el lenguaje de programación Mikro C se lo puede apreciar en el **ANEXO B.**

2.4.3 LECTURA DE LAS SEÑALES DE AUDIO Y COMUNICACIÓN UART CON EL PIC 16F877A

PIC 16F877A

Es un micro controlador de la familia microchip su consumo de potencia es muy bajo y además es completamente estático, esto quiere decir que el reloj puede detenerse y los datos de la memoria no se pierden.

2.4.3.1 LECTURA ADC

En la **figura 30** observamos la lectura de audio que me realiza el Pic 16f 877A por la entrada del pin AN0.

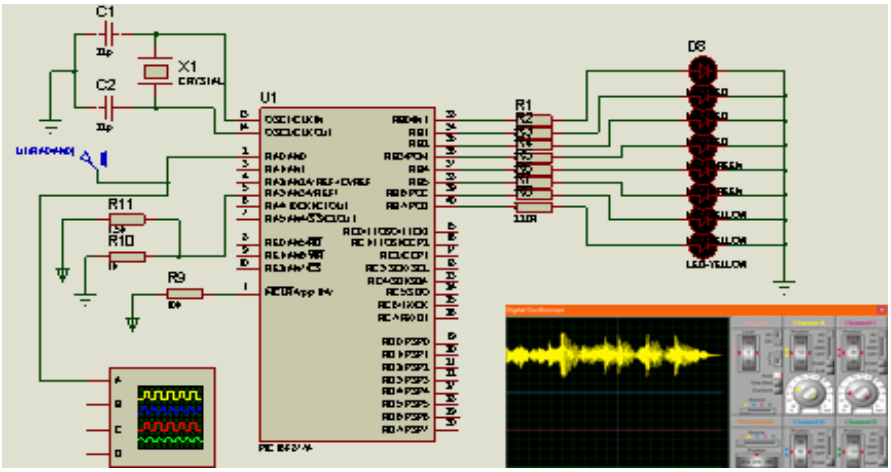


Figura 30: Lectura de la señal de audio
Fuente: Auto

2.4.3.2 TRANSMISIÓN DE DATOS MEDIANTE LA COMUNICACIÓN UART

En la **figura 31** visualizamos la comunicación que me realiza el Pic 16f877A hacia el robot, el Pic se encarga de enviar datos obtenidos del algoritmo dependiendo del tono que se detecte.

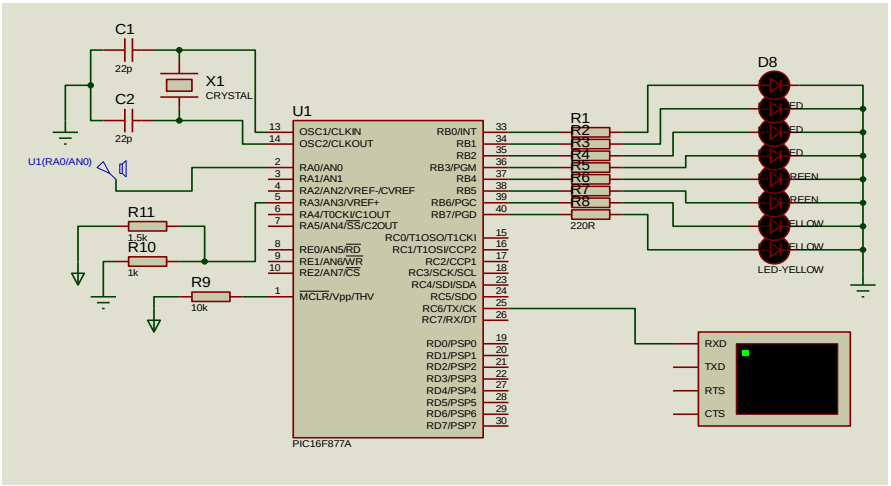


Figura 31: Comunicación UART
Fuente: Autor

2.4.3.3 DISEÑO DEL RECONOCEDOR DE FRECUENCIAS DE AUDIO

En la **figura 32** se puede visualizar la placa electrónica que está diseñada para realizar el procesamiento de los tonos de audio, el PIC 16f877A realiza la lectura de las diferentes

frecuencias de audio y lo transmite al robot dependiendo de la frecuencia de cada tono de audio adquirido.

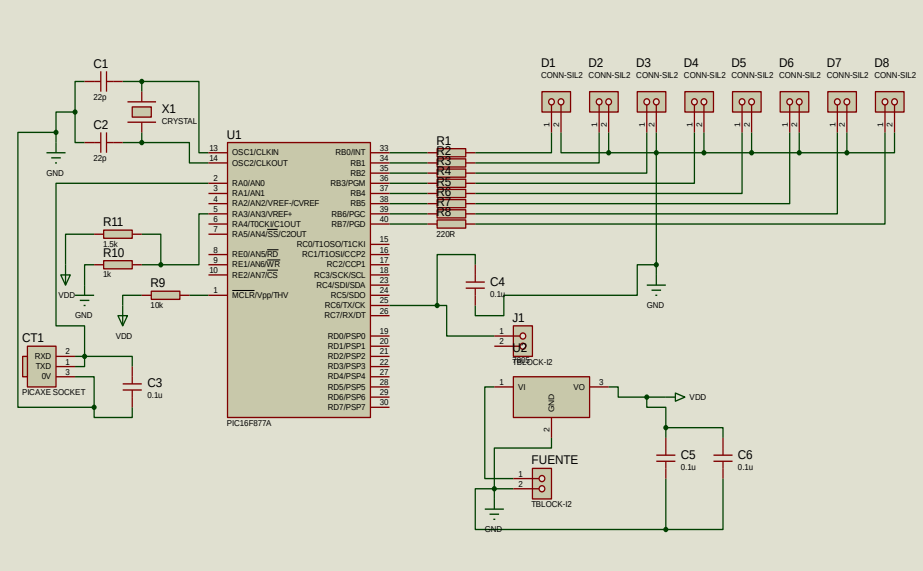


Figura 32: Diseño de la placa
Fuente: Autor

2.4.3.4 DISEÑO DE LA PLACA DEL RECONOCEDOR DE FRECUENCIAS DE AUDIO EN ARES

En la **figura 33** se muestra el diseño realizado en ares para el funcionamiento del reconocedor de frecuencias de audio, el ruteo de las pistas tiene un ancho de pista de 4 mm tomando en cuenta el cálculo realizado.

$$\text{Ancho} = \{[I / (k1 * \Delta T^{k2})]^{1/k3}\} / (L * 1{,}378)$$

I= Corriente máxima 1A

K1= Constante estándar que vale 0,0150

K2= Constante estándar que vale 0,5453

K3=Constante que vale 0,7349

L= 2 onzas de cobre

ΔT= Variación de temperatura

$$Ancho = \frac{\left(\frac{1}{0.0150 * 10^{0.5458}}\right)^{\frac{1}{0.7849}}}{2 * 1.378}$$

Ancho = 3.875 mm ≈ 4 mm

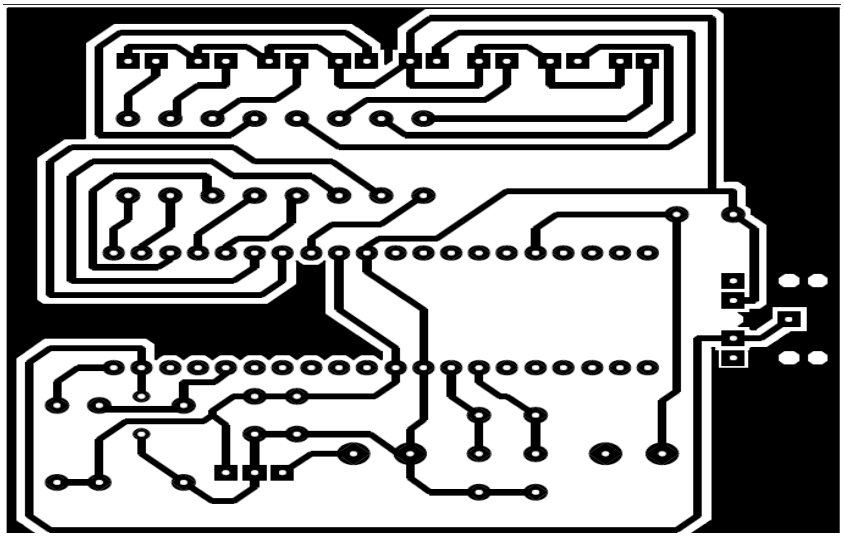


Figura 33: Diseño en el ares del reconocedor de frecuencias de audio
Fuente: Autor

2.4.3.5 DIAGRAMA EN 3D DE LA PLACA DEL RECONOCEDOR DE FRECUENCIAS DE AUDIO

En la **figura 34** se muestra la placa en 3D como queda físicamente el reconocedor de frecuencias.

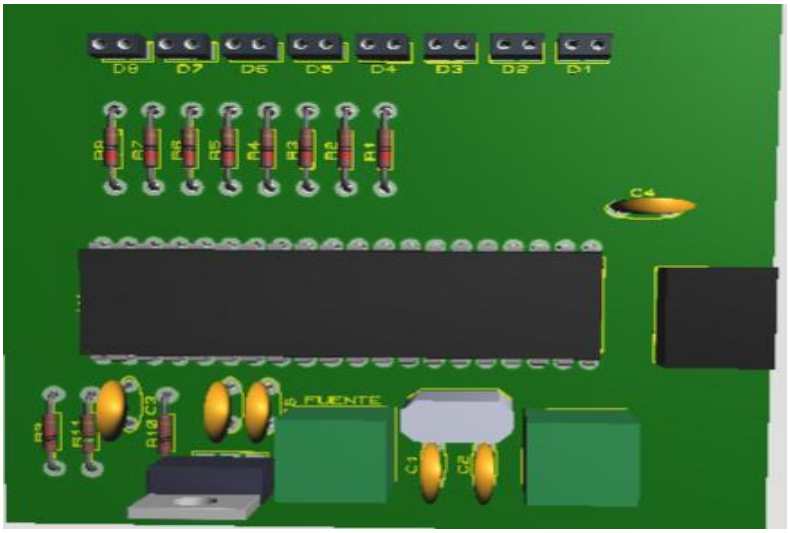


Figura 34: Placa del reconocedor de frecuencias de audio en 3D
Fuente: Autor

La simulación y diseño de la placa del reconocedor de frecuencias de audio se lo puede visualizar en el **ANEXO C.**

2.5 PROCESAMIENTO Y ANÁLISIS

En el siguiente trabajo de investigación se obtiene conocimientos adecuados y profundos en el área de electrónica y robótica, utilizando diferentes tipos de protocolos aplicados al micro controlador PIC 16F877A.

Con el protocolo de comunicación UART podemos controlar los diferentes movimientos realizados por el robot.

Las frecuencias de audio serán obtenidas mediante una lectura ADC con el PIC16F877A, el cual se encarga de procesar los diferentes tonos de audio como son: agudos, medios y graves dependiendo del tipo de tono el PIC16F877A mediante la comunicación UART envía al robot diferentes datos los cuales activan o desactivan los diferentes movimientos del robot.

2.5.1 ANÁLISIS DEL PIC 16F877A

El presente trabajo de investigación es el micro controlador 16F877A pertenece a la gama alta de micro controladores de arquitectura abierta pudiéndose expansionar en el exterior al poder sacar los buses de datos, direcciones y control. Así se pueden configurar sistemas similares a los que utilizan los microprocesadores convencionales, siendo capaces de ampliar la configuración interna del PIC añadiendo nuevos dispositivos de memoria y de E/S externas, disponen de reset automático al conectar la alimentación, reset ante el fallo de la alimentación, Perro guardián, Código de Protección opcional. Disponen de comparadores analógicos, convertidores Analógicos/Digitales (CAD) de 8 bits, circuito de Captura/Comparación, Interfaz serie síncrono SP/I2 de 2, 3 o 4 hilos, Interfaz serie asíncrono SCI de 8 o 9 bits, PWM para manipular impulsos de 8 o 10 bits con modulación de anchura y PSP, que consiste en un Puerto paralelo esclavo que proporciona la conexión con otros

microprocesadores, Comunicación UART permite transmitir datos a una determinada velocidad. Además, estos recursos se distribuyen eficazmente entre los diversos modelos para que el usuario disponga de un gran abanico de posibilidades para optimizar su elección.

2.5.2 ANÁLISIS PRÁCTICO DE LA COMUNICAIÓN UART

En la práctica de comunicación se realizó usando estas configuracionesUART1_Init (2400); es usado para comenzar la Tx de los datos a la velocidad determinada, UART1_Write (Dato); UART1_Write ('# servo'); para comprobar la comunicación del PIC hacia el robot; Uart (explicación del bit de Start/Stop). Tx (emisor) con Rx (receptor), es un tipo de comunicación asíncrona.

En la práctica los datos transmitidos a través de la comunicación UART es muy confiable, una característica de la transmisión UART a tener en cuenta es la velocidad en la que se va a transmitir los datos en la **figura 35** se puede observar la Tx de datos que está determinada a una velocidad de 2400 baudios.

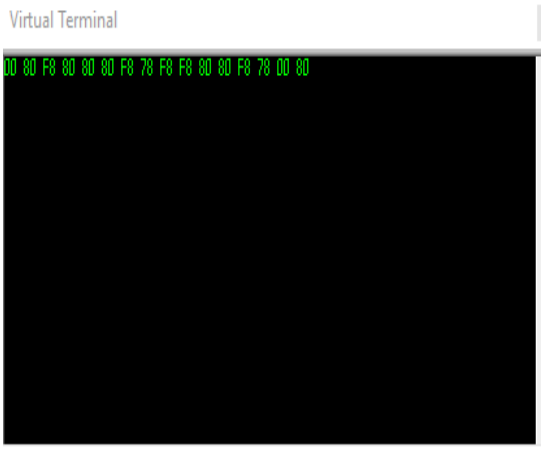


Figura 35: Transmisión de datos
Fuente: Autor

2.5.3 TÉCNICAS DE PROCEDIMIENTO PARA EL ANÁLISIS

El análisis estadístico es cualitativo y cuantitativo. Para la parte cualitativa es comprobar la estabilidad y confiabilidad de la comunicación UART en la transmisión de datos de los

movimientos del robot, para la parte cuantitativa se comprobará la eficiencia en el desarrollo de prácticas usando el reconocedor de frecuencias de audio por parte del grupo de investigación del área de Electrónica y Telecomunicaciones.

2.6 COMPROBACIÓN DE LA HIPÓTESIS

2.6.1 PLANTEAMIENTO DE LA HIPÓTESIS

Se plantea la hipótesis nula y la hipótesis afirmativa:

Ho Con el reconocedor de frecuencias de audio no mejora la eficiencia en el control del robot.

Hl Con el reconocedor de frecuencias de audio mejora la eficiencia en el control del robot.

2.6.2 ESTABLECIMIENTO DEL NIVEL DE SIGNIFICANCIA

El nivel de significancia es, α= 0,05 % que representa el 5% y con un 95% de confianza.

2.6.3 ELECCIÓN DE LA PRUEBA DE HIPÓTESIS

Es una prueba de muestra relacionadas a un mismo grupo se aplica dos medidas en un tiempo distinto, es un estudio longitudinal, la variable fija que crea los grupos son dos medidas una antes y una media después, la variable aleatoria de comparación es la variable tiempo es una variable numérica, por lo tanto, se usa la prueba t student con muestras relacionadas para comprobar la hipótesis.

2.6.4 MUESTRA

La muestra realizada para la comprobación de la hipótesis es de 7 estudiantes que pertenecen al grupo de investigación de Ingeniería Electrónica y Telecomunicaciones de la Universidad Nacional de Chimborazo.

2.6.5 MEDIA

Estudiantes	Tiempo sin reconocedor de frecuencias	Tiempo con reconocedor de frecuencias
1	55.3	40.1
2	59.1	43.9
3	60	45
4	48.6	35.2
5	45.8	30.5
6	46	32.1
7	50.7	37.2

Tabla 8: Tiempo tomados de estudiantes para comprobación de la hipótesis.
Elaborado: Autor

Media calculada

M1= (365.5) /7=52.21 Media (Tiempo sin reconocedor de frecuencias)

M2= (264) /7=37.71 Media (Tiempo con reconocedor de frecuencias)

2.6.6 PRUEBA DE NORMALIDAD

Para calcular la normalidad se usa el método de Shapiro-Wilk un método usado para muestras menores a 30 se usa el software spss para encontrar el **p valor** (nivel de significación más pequeño posible que puede escogerse), para ver si la variable de tiempo se comporta con normalidad.

Pruebas de normalidad

	Kolmogorov-Smirnov			Shapiro-Wilk		
	Estadístico	gl	Sig.	Estadístico	gl	Sig.
Tsrf	,172	7	,200*	,891	7	,278
Tcrf	,152	7	,200*	,945	7	,687

Tabla 9: Pruebas de normalidad en spss
Fuente: Autor

Tsrf=Tiempo sin reconocedor, Tcrf=Tiempo con reconocedor, gl=cantidad de estudiantes.

Normalidad

Pvalor (t-antes) = ,278	>	α= 0.05
Pvalor (t-después) = ,687	>	α= 0.05

Tabla 10: Tabla de resultados de normalidad
Fuente: Autor

El valor de datos de p valor (nivel de significancia) son mayor que el nivel de α (error aceptado).

Los datos de tiempo provienen de una distribución normal, la variable tiempo se comporta normalmente.

2.6.7 PRUEBA T PARA MUESTRAS RELACIONADAS

Estadísticos de muestras relacionadas

		Media	N	Desviación estándar	Media de error estándar
Par 1	Tsrf	52,2143	7	5,95355	2,25023
	Tcrf	37,7143	7	5,58553	2,11113

Tabla 11: Estadísticas de muestras relacionadas
Fuente: Autor

Correlaciones de muestras emparejadas

		N	Correlación	Sig.
Par 1	Tsrf & Tcrf	7	,991	,000

Tabla 12: Correlación de muestras relacionadas
Fuente: Autor

Prueba de muestras relacionadas

		Diferencias emparejadas					t	gl	Sig. (bilateral)
		Media	Desviación estándar	Media de error estándar	95% de intervalo de confianza de la diferencia				
					Inferior	Superior			
Par 1	Tsrf - Tcrf	14,50000	,86023	,32514	13,70442	15,29558	44,597	6	,000

Tabla 13: Prueba de muestras relacionadas
Fuente: Autor

El criterio a tomar en cuenta en la prueba de muestras relacionadas:

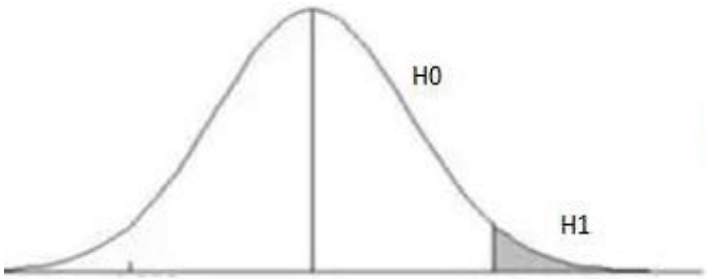
- Si la probabilidad obtenida de P-valor <= α se rechaza Ho y se acepta H1
- Si la probabilidad obtenida de P-valor >= α se rechaza H1 y se acepta Ho.

La variable tiempo antes y después de la prueba disminuye significativamente, comparando las medias en la tabla.

En la **tabla 13** se muestra que gl es 6, comprobando con la tabla de distribución t el valor es 1.943 mayor que α= 0.05% y tomando en cuenta el criterio de decisión de la prueba t student se llega a la conclusión que se rechaza Ho y se acepta H1.

Hl Con el reconocedor de frecuencias de audio mejora la eficiencia en el control del robot.

En la **figura 36** se puede apreciar la curva de normalidad y las zonas de H0 y H1.



α / r	0,25	0,2	0,15	0,1	0,05	0,025	0,01	0,005	0,0005
1	1,000	1,376	1,963	3,078	6,314	12,706	31,821	63,656	636,578
2	0,816	1,061	1,386	1,886	2,920	4,303	6,965	9,925	31,600
3	0,765	0,978	1,250	1,638	2,353	3,182	4,541	5,841	12,924
4	0,741	0,941	1,190	1,533	2,132	2,776	3,747	4,604	8,610
5	0,727	0,920	1,156	1,476	2,015	2,571	3,365	4,032	6,869
6	0,718	0,906	1,134	1,440	1,943	2,447	3,143	3,707	5,959
7	0,711	0,896	1,119	1,415	1,895	2,365	2,998	3,499	5,408
8	0,706	0,889	1,108	1,397	1,860	2,306	2,896	3,355	5,041
9	0,703	0,883	1,100	1,383	1,833	2,262	2,821	3,250	4,781
10	0,700	0,879	1,093	1,372	1,812	2,228	2,764	3,169	4,587
11	0,697	0,876	1,088	1,363	1,796	2,201	2,718	3,106	4,437
12	0,695	0,873	1,083	1,356	1,782	2,179	2,681	3,055	4,318
13	0,694	0,870	1,079	1,350	1,771	2,160	2,650	3,012	4,221
14	0,692	0,868	1,076	1,345	1,761	2,145	2,624	2,977	4,140
15	0,691	0,866	1,074	1,341	1,753	2,131	2,602	2,947	4,073

Figura 36: Aceptación de H1 y rechazo de H0
Fuente: Autor

CAPITULO III

3 RESULTADOS

A continuación, se detalla las pruebas realizadas para la comunicación entre el PIC 16F877A y el robot.

3.1 PRUEBA REALIZADA CON EL FILTRO

Realizamos las respectivas pruebas con el filtro realizado en Matlab, ingresamos la señal de audio original y la salida nos da la señal de audio filtrada como se puede observar en la **figura 37**.

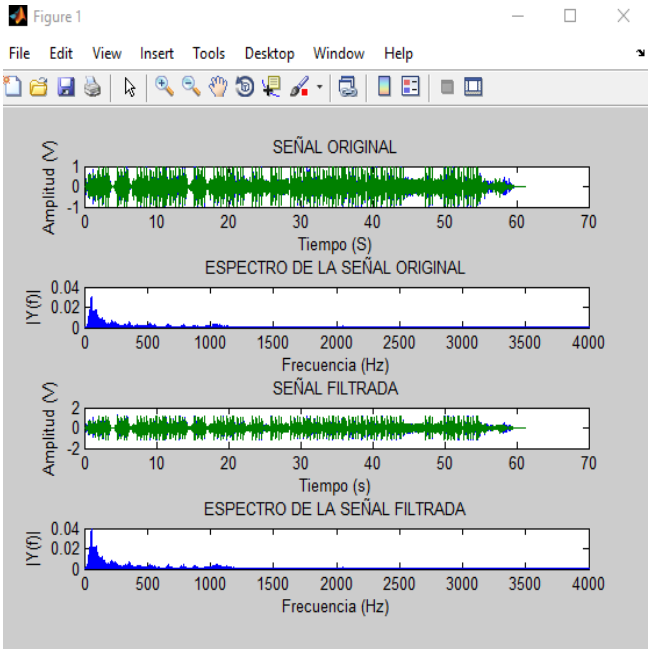


Figura 37: Prueba con el filtro realizado en Matlab
Fuente: Autor

El resultado de esta prueba fue que al realizar el proceso del filtrado del audio la salida de la señal me genera una pérdida de datos de un 60%.

3.2 PRUEBAS DE COMUNICACIÓN UART

Se procede a comprobar él envió de datos mediante la comunicación UART como se muestra en la **figura 38**.

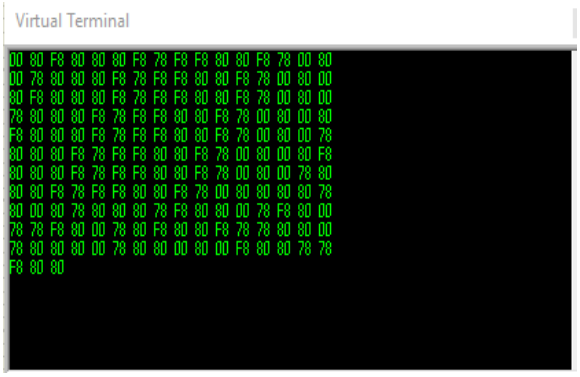


Figura 38: Comunicación UART
Fuente: Autor

El resultado de la comunicación UART fue exitoso.

3.2.1 PRUEBAS DE ENVIÓ DE DATOS DE LOS DIFERENTES TONOS DE AUDIO

Se probó la comunicación entre el PIC y el robot con los diferentes tonos de audio, para ver que la comunicación entre dispositivos sea exitosa se procede a realizar la extracción de las diferentes frecuencias de audio.

Procedemos a extraer las frecuencias bajas los leds de color verde me detectan los tonos graves, como se ilustra en la **figura 39**.

En la **figura 40** se puede observar los movimientos realizados por el robot con los tonos graves.

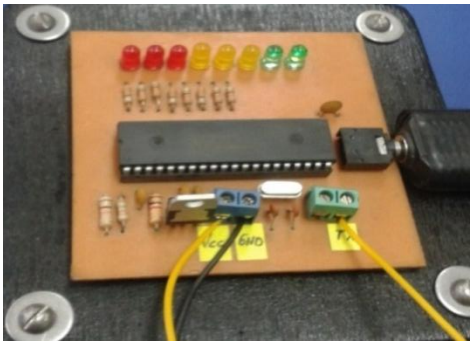


Figura 39: Extracción de tonos graves
Fuente: Autor

Figura 40: Robot bailando tonos graves
Fuente: Autor

Después se realiza la extracción de las frecuencias medias los leds de color amarillo me detectan los tonos medios, como se observa en la **figura 41**.

En la **figura 42** se puede apreciar los movimientos realizados por el robot con los tonos medios.

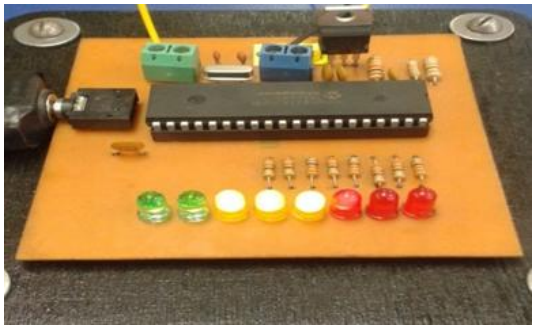


Figura 41: Extracción de tonos medios
Fuente: Autor

Figura 42: Robot bailando tonos medios
Fuente: Autor

Luego se procede a extraer las frecuencias altas los leds de color rojo me detectan los tonos agudos como se observa en la **figura 43**.

Los movimientos realizados por el robot con los tonos agudos se pueden visualizar en la **figura 44**.

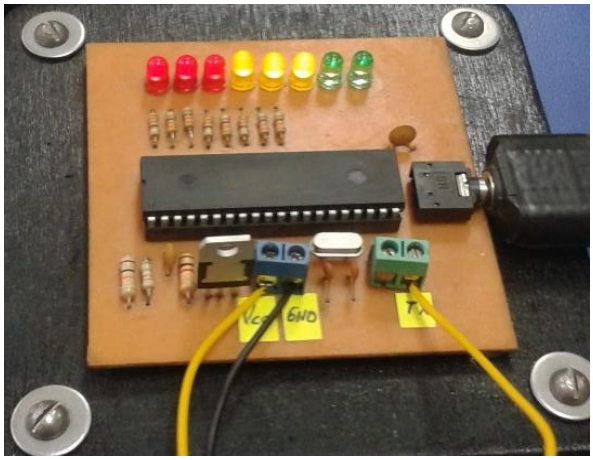


Figura 43: Extracción de tonos agudos
Fuente: Autor

Figura 44: Robot bailando tonos agudos
Fuente: Autor

Los resultados de estas pruebas fueron exitosos ya que al detectar las diferentes frecuencias de audio el robot realiza diferentes movimientos de baile al son de la música.

3.3 ANÁLISIS FINANCIERO

Costos	Descripción	Valor Usd.
Hardware	Laptop	1200,00
	Micro controlador	20,00
	Protoboard	35,00
	Resistencias	2,00
	Condensadores	1,00
	Cristal de cuarzo	2,00
	Leds	1,00
	Cables	6,00
	Borneras	1,00
	Trabajos en plaqueta de cobre	80,00
Software	Mikro C	0,00
	Proteus Profesional	0,00
	Matlab	0,00
Proyecto	Transporte	150,00
Varios	Cables, estaño, pasta, cautín.	50,00
Subtotal		1548,00
Imprevistos 10% subtotal		154,80
TOTAL		**1702,80**

Tabla 14: Análisis financiero
Fuente: Autor

CAPITULO IV

4 CONCLUSIONES Y RECOMENDACIONES

4.1 CONCLUSIONES

- El filtro digital al extraer los diferentes tonos de audio para realizar el proceso del filtrado de la señal de audio me genera una pérdida de datos del 60% a la salida de la señal filtrada.
- El algoritmo programado en Mikro C analiza la señal de entrada y clasifica los diferentes tonos de audio dándonos así a si una mayor efectividad de lectura de los datos.
- La comunicación UART transmite los diferentes movimientos que debe realizar el robot de acuerdo al tono de audio detectado por el algoritmo.
- El reconocedor de frecuencias de audio me permite obtener un control eficiente de los movimientos que realiza el robot al son de la música.

4.2 RECOMENDACIONES

- En el micro controlador se debe configurar la lectura ADC por un puerto determinado del PIC 16F877A.
- Evitar cualquier tipo de ruido en el sistema al momento de realizar la comunicación entre el PIC y el robot, es recomendable utilizar fuentes independientes de 5V.
- En la configuración de la transmisión de datos se recomienda determinar la velocidad de trasmisión en baudios.

CAPITULO V

5 PROPUESTA

5.1 TÍTULO DELA PROPUESTA

Diseño y construcción de un reconocedor de frecuencias de audio para un robot bailarín.

5.2 INTRODUCCIÓN

La construcción del reconocedor de frecuencias de audio me permite mejorar el control del robot para realizar distintos ritmos de baile al son de la música, también se puede utilizar para la realización de prácticas relacionadas a la robótica,

El algoritmo está configurado mediante comandos de programación a través de un micro controlador, el cual me recepta los diferentes tonos de audio.

La comunicación UART se encarga de Transmitir los datos obtenidos del algoritmo el cual activa y desactiva los diferentes movimientos realizados por el robot.

5.3 DISCUSIÓN

La investigación se enfoca en extraer los tonos graves, medios y agudos utilizando el reconocer las frecuencias de audio para un robot bailarín.

Basándose en investigación teórica sobre el rango de frecuencias a las que trabajan los diferentes tonos de audio, usando el micro controlador 16F877A se realizó la programación de un algoritmo en Mikro C el cual detecta las diferentes frecuencias de audio. La comunicación UART realiza el control del robot comprobando así la estabilidad de la transmisión de datos.

Se realizaron pruebas con diferentes géneros musicales comprobando a si la extracción de los diferentes tonos de audio, con el uso del reconocedor de frecuencias de audio el control del robot al realizar los movimientos del baile al son de la música son eficientes y estables.

CAPITULO VI

6 OBJETIVOS

6.1 OBJETIVO GENERAL

- Diseñar y construir un reconocedor de frecuencias de audio para un robot bailarín.

6.2 OBJETIVOS ESPECÍFICOS

- Conocer el procesamiento de filtros digitales para extraer los tonos de audio.
- Comprender los conceptos y las características de un algoritmo.
- Desarrollar un algoritmo que me permita realizar los movimientos del robot.
- Diseñar el control del robot que funcione con tonos musicales.

6.3 FUNDAMENTACIÓN CIENTÍFICO – TÉCNICA

Contendrá una versión resumida y actualizada del estado del conocimiento que se encuentra en el tema específico de la propuesta.

También conocidos como Frecuencias de sonido: Las frecuencias de sonido son un número de oscilaciones o variaciones de la presión por segundo. La frecuencia es una magnitud objetiva y mensurable referida a formas de onda periódicas. Tiene que ver con cuántos ciclos por segundo tiene que dar la onda, indicando la idea de rapidez con que se producen. (Foto-Nostra, 2008)

6.4 DESCRIPCIÓN DE LA PROPUESTA

El reconocedor de frecuencias de audio para un robot bailarín utilizando un algoritmo para detectar los diferentes tonos de audio, mediante la comunicación UART transmite los datos hacia el robot de acuerdo a los parámetros de programación.

6.5 DISEÑO ORGANIZACIONAL

Se establecerá la estructura orgánica y funcional de la unidad administrativa que ejecutará la propuesta como se muestra en la **figura 45**.

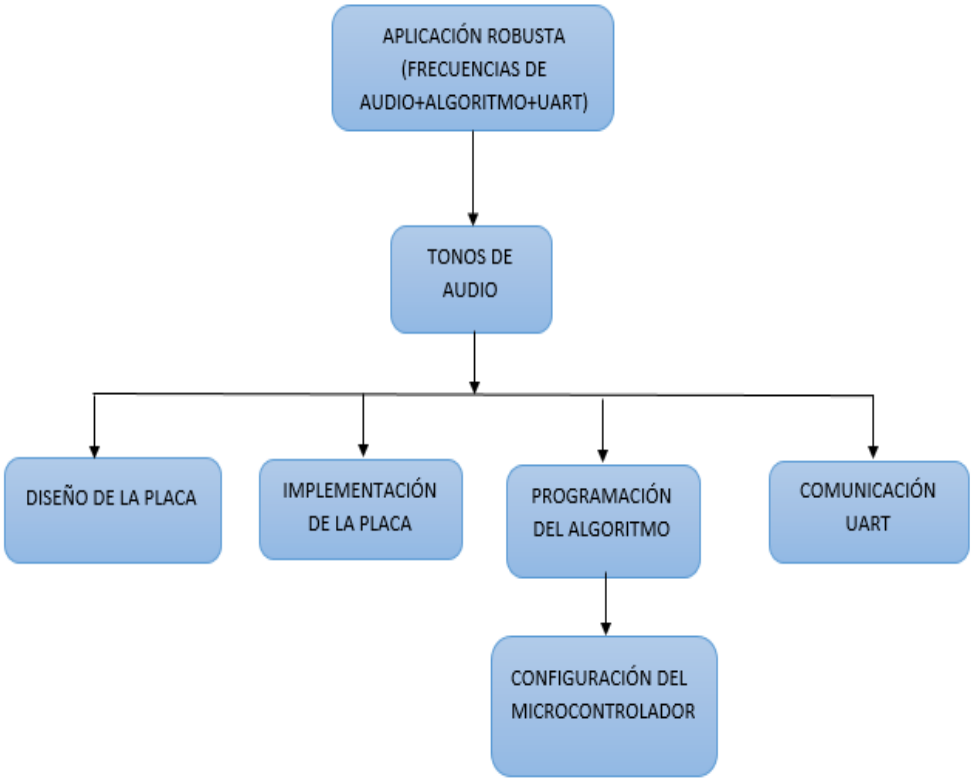


Figura 45: Diseño organizacional del proyecto
Fuente: Autor

CAPITULO VII

7 BIBLIOGRAFÍA

Canto Q., C. (2013). *Microcontroladores*. Obtenido de http://galia.fc.uaslp.mx/~cantocar/microcontroladores/SLIDES_8051_PDF/2_INTROD.PDF

Clavijo Mendosa, J. R. (2011). *Diseño y Simulaciòn de Sistemas Microcontrolados en Lenguaje C.* Colombia .

Foto-Nostra. (2008). *Frecuencia de audio*. Obtenido de http://www.fotonostra.com/digital/frecuenciaudio.htm

Garre del Olmo, C. (2008). *Microcontroladores*. Obtenido de http://dac.escet.urjc.es/docencia/Micros/MP06a.pdf

Gomez Gutièrrez, E. (2009-2010). *Introduccion al filtro digitral.* Catalunya.

Joyanes Aguilar, L. (7 agosto 2015). *Fundamentos de programacion.* 4ta Edicion.

Lara, A., Tapia, V., Gamboa, D., Narváez, J., & Parreño, D. (2013). *Scribd*. Obtenido de https://es.scribd.com/doc/46470741/Los-Filtros-FIR

M, M. (2010). *Open course Ware*. Obtenido de http://ocw.uv.es/ingenieria-y-arquitectura/filtros-digitales/tema_4_diseno_de_filtros_iir.pdf

Microchip. (2001-2013). *Microchip*. Obtenido de http://ww1.microchip.com/downloads/en/DeviceDoc/39582C.pdf

Miraya, F. (2004). *Electrònica III*. Obtenido de http://www.fceia.unr.edu.ar/enica3/filtros-t.pdf

Moreno Velazco, I. (2012). *Tecnologia electronica*. Obtenido de http://www.unet.edu.ve/~ielectro/6-Filtrado.pdf

Ojeda Guadamud, J. (20 de Febrero de 2014). *Repositorio*. Obtenido de http://repositorio.ucsg.edu.ec:8080/bitstream/123456789/1723/1/T-UCSG-PRE-TEC-ITEL-42.pdf

Perez Vega, C. (2010). *Ingenieria en comunicaciones*. Obtenido de http://personales.unican.es/perezvr/pdf/Sonido%20y%20Audicion.pdf

Terven Salinas, J. (2013). *Micros*. Obtenido de http://www.tervenet.com/itmaz/micros2/PIC32_11_UART.pdf

8 ANEXOS

ANEXO A

GLOSARIO DE TÉRMINOS

RX	Receptor
TX	Transmisión
ADC	Lectura analógica
Tsrf	Tiempo sin reconocedor
Tcrf	Tiempo con reconocedor
UART	Comunicación de datos
Mikro C	Lenguaje de programación
gl	Número de estudiantes

FILTRO DIGITAL FIR EN MATLAB

A continuación, se muestra el filtro que se utilizó para extraer los diferentes tonos de audio el programa se lo realizado en Matlab.

CÓDIGO EN MATLAB DE UN FILTRO DIGITAL

```
%% Procesamiento de una señal de audio usando MATLAB

%% Selección del tipo de filtrado
% 1 -> Pasa bajo
% 2 -> Pasa alto
% 3 -> Pasa banda
tipo=1;
%% Crear señal de audio
```

```
% Frecuencia fundamental
f0=8e3; % 8KHz
% Amplitud
a=3; % V=4
% Frecuencia de muestreo
fs=44.1e3; % Frecuencia de una señal de audio CD
% Tiempo de duración en segundos
T=1.5;
% Vector de tiempo
t=linspace(0,T,T*fs);
% Creación de la señal
% Primer señal (tono 1)
s1=a*sin(2*pi*f0*t);
% Segunda señal (tono 2)
s2=0.75*a*sin(2*pi*(1.5*f0)*t);
% Tercera señal (tono 3)
s3=0.5*a*sin(2*pi*(2*f0)*t);
% Señal compuesta (suma de dos tonos)
y = s1 + s2 + s3;
% Graficar la señal original
subplot(411)
plot(t,y)
title('SEÑAL ORIGINAL')% Título
xlabel('Tiempo (s)')        % Etiqueta del eje X
ylabel('Amplitud (V)')      % Etiqueta del eje Y
xlim([0 20/f0])           % Límite de la señal
%% Grabar y reproducir la señal de audio
% wavwrite(0.1*y,fs,'audio')
% wavplay(0.1*y,fs)
```

```
%% FFT de la señal
subplot(412)
% Llamado a la función que calcula la FFT
fft_signal(y,fs);title('ESPECTRO DE LA SEÑAL ORIGINAL')
xlim([0 f0*3])
%% Filtrado de la señal
switch tipo
   case 1
      % Cálculo de los coeficientes del filtro (filtro pasa bajas)
      % Este filtrado deja solo la señal de 1000 Hz
      % Frecuencia normalizada
      titutlo='FILTRO PASA BAJAS';
      fNorm = 8.5e3 / (fs/2);
      [b,a] = fir1(10, fNorm, 'low');
   case 2
      %- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
      % Cálculo de los coeficientes del filtro (filtro pasa bajas)
      % Este filtrado deja solo la señal de 2000 Hz
      % Frecuencia normalizada
      titutlo='FILTRO PASA ALTAS';
      fNorm = 15e3 / (fs/2);
      [b,a] = fir1(10, fNorm, 'high');
   otherwise
      %- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
      % Cálculo de los coeficientes del filtro (filtro pasa banda)
      % Este filtrado deja solo la señal de 1500 Hz
      % Frecuencias normalizadas
      titutlo='FILTRO PASA BANDA';
      Wp = [11.5e3 12.5e3]/(fs/2); Ws = [11e3 13e3]/(fs/2);
```

```
        Rp = 3; Rs = 40;
        [n,Wn] = buttord(Wp,Ws,Rp,Rs)
        [b,a] = fir1(n,Wn);
        %- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
end
% Filtrado de la señal
y_Low = filtfilt(b, a, y);
% Graficación de la señal en el tiempo
subplot(413)
plot(t,y_Low)
title('SEÑAL FILTRADA')
xlabel('Tiempo (s)')
ylabel('Amplitud (V)')
xlim([0 20/f0])
% Graficación de la señal en frecuencia
subplot(414)
% Llamado a la función que calcula la FFT
fft_signal(y_Low,fs);title('ESPECTRO DE LA SEÑAL FILTRADA')
xlim([0 3*f0])
%% Gráficas del filtro
% Respuesta en frecuencia del filtro
[H,w]=freqz(b,a,512,fs);
figure(2)
%Trazado de la respuesta en Magnitud
subplot(221)
plot(w,20*log10(abs(H)));
grid on;
title ([titutlo, 'Respuesta en magnitud']);
xlabel('Frecuencia (Hz)');
```

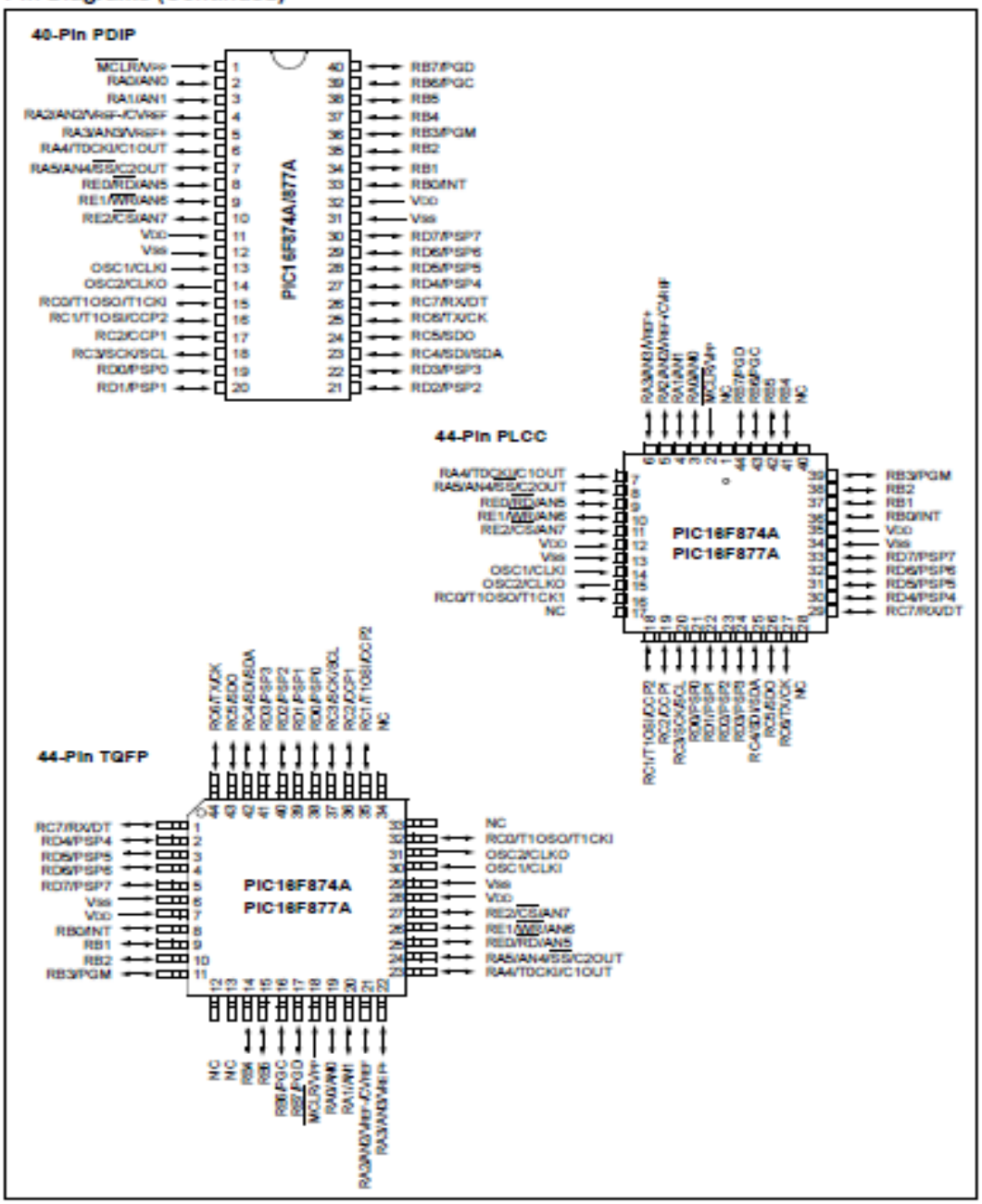
```
ylabel('H(f) db')
% xlim([0 0.4])
% Respuesta en fase
subplot(222)
plot(w,angle(H));
grid on;
title ([titutlo,'Respuesta en fase']);
xlabel('Frecuencia (Hz)')
ylabel('ángulo de H rad')
% xlim([0 0.4])
%Respuesta al impulso
subplot(223)
[y_eje,tt]= impz(b,a,60);
stem(tt,y_eje);
title ([titutlo,'Respuesta al impulso']);
 %Ploteo de los polos y ceros
z= roots(b);   % Ceros
p = roots(a);  % Polos
subplot(224)
zplane(z,p)
title('Polos y ceros')
legend('Ceros','Polos')
%% Reproducción de audio de entrada y salida
disp('Audio de entrada (señal original)')
% Se multiplica por 0.2 para atenuar la salida del tono por la bocina
wavplay(0.2*y,fs)
disp('Audio de salida (señal filtrada)')
% Se multiplica por 0.2 para atenuar la salida del tono por la bocina
wavplay(0.2*y_Low,fs)
```

ANEXO B

PROGRAMACIÓN DEL ALGORITMO EN LENGUAJE MIKRO C

DATA SHEET DEL PIC 16F877A

Pin Diagrams (Continued)



CONFIGURACIÓN PARA LA LECTURA ADC

11.0 ANALOG-TO-DIGITAL CONVERTER (A/D) MODULE

The Analog-to-Digital (A/D) Converter module has five inputs for the 28-pin devices and eight for the 40/44-pin devices.

The conversion of an analog input signal results in a corresponding 10-bit digital number. The A/D module has high and low-voltage reference input that is software selectable to some combination of VDD, VSS, RA2 or RA3.

The A/D converter has a unique feature of being able to operate while the device is in Sleep mode. To operate in Sleep, the A/D clock must be derived from the A/D's internal RC oscillator.

The A/D module has four registers. These registers are:

- A/D Result High Register (ADRESH)
- A/D Result Low Register (ADRESL)
- A/D Control Register 0 (ADCON0)
- A/D Control Register 1 (ADCON1)

The ADCON0 register, shown in Register 11-1, controls the operation of the A/D module. The ADCON1 register, shown in Register 11-2, configures the functions of the port pins. The port pins can be configured as analog inputs (RA3 can also be the voltage reference) or as digital I/O.

Additional information on using the A/D module can be found in the PIC® Mid-Range MCU Family Reference Manual (DS33023).

REGISTER 11-1: ADCON0 REGISTER (ADDRESS 1Fh)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0
ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/DONE	—	ADON
bit 7							bit 0

bit 7-6 **ADCS1:ADCS0:** A/D Conversion Clock Select bits (ADCON0 bits in **bold**)

ADCON1 <ADCS2>	ADCON0 <ADCS1:ADCS0>	Clock Conversion
0	00	Fosc/2
0	01	Fosc/8
0	10	Fosc/32
0	11	FRC (clock derived from the internal A/D RC oscillator)
1	00	Fosc/4
1	01	Fosc/16
1	10	Fosc/64
1	11	FRC (clock derived from the internal A/D RC oscillator)

bit 5-3 **CHS2:CHS0:** Analog Channel Select bits

000 = Channel 0 (AN0)
001 = Channel 1 (AN1)
010 = Channel 2 (AN2)
011 = Channel 3 (AN3)
100 = Channel 4 (AN4)
101 = Channel 5 (AN5)
110 = Channel 6 (AN6)
111 = Channel 7 (AN7)

Note: The PIC16F873A/876A devices only implement A/D channels 0 through 4; the unimplemented selections are reserved. Do not select any unimplemented channels with these devices.

bit 2 **GO/DONE:** A/D Conversion Status bit

When ADON = 1:

1 = A/D conversion in progress (setting this bit starts the A/D conversion which is automatically cleared by hardware when the A/D conversion is complete)
0 = A/D conversion not in progress

bit 1 **Unimplemented:** Read as '0'

bit 0 **ADON:** A/D On bit

1 = A/D converter module is powered up
0 = A/D converter module is shut-off and consumes no operating current

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 11-2: ADCON1 REGISTER (ADDRESS 9Fh)

R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	ADCS2	—	—	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

bit 7 **ADFM:** A/D Result Format Select bit

1 = Right justified. Six (6) Most Significant bits of ADRESH are read as '0'.
0 = Left justified. Six (6) Least Significant bits of ADRESL are read as '0'.

bit 6 **ADCS2:** A/D Conversion Clock Select bit (ADCON1 bits in shaded area and in bold)

ADCON1 <ADCS2>	ADCON0 <ADCS1:ADCS0>	Clock Conversion
0	00	FOSC/2
0	01	FOSC/8
0	10	FOSC/32
0	11	FRC (clock derived from the internal A/D RC oscillator)
1	00	FOSC/4
1	01	FOSC/16
1	10	FOSC/64
1	11	FRC (clock derived from the internal A/D RC oscillator)

bit 5-4 **Unimplemented:** Read as '0'

bit 3-0 **PCFG3:PCFG0:** A/D Port Configuration Control bits

PCFG <3:0>	AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0	VREF+	VREF-	C/R
0000	A	A	A	A	A	A	A	A	VDD	VSS	8/0
0001	A	A	A	A	VREF+	A	A	A	AN3	VSS	7/1
0010	D	D	D	A	A	A	A	A	VDD	VSS	5/0
0011	D	D	D	A	VREF+	A	A	A	AN3	VSS	4/1
0100	D	D	D	D	A	D	A	A	VDD	VSS	3/0
0101	D	D	D	D	VREF+	D	A	A	AN3	VSS	2/1
011x	D	D	D	D	D	D	D	D	—	—	0/0
1000	A	A	A	A	VREF+	VREF-	A	A	AN3	AN2	6/2
1001	D	D	A	A	A	A	A	A	VDD	VSS	6/0
1010	D	D	A	A	VREF+	A	A	A	AN3	VSS	5/1
1011	D	D	A	A	VREF+	VREF-	A	A	AN3	AN2	4/2
1100	D	D	D	A	VREF+	VREF-	A	A	AN3	AN2	3/2
1101	D	D	D	D	VREF+	VREF-	A	A	AN3	AN2	2/2
1110	D	D	D	D	D	D	D	A	VDD	VSS	1/0
1111	D	D	D	D	VREF+	VREF-	D	A	AN3	AN2	1/2

A = Analog input D = Digital I/O
C/R = # of analog input channels/# of A/D voltage references

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

Note: On any device Reset, the port pins that are multiplexed with analog functions (ANx) are forced to be an analog input.

CONFIGURACIÓN UART PARA LA TX DE DATOS

10.0 ADDRESSABLE UNIVERSAL SYNCHRONOUS ASYNCHRONOUS RECEIVER TRANSMITTER (USART)

The Universal Synchronous Asynchronous Receiver Transmitter (USART) module is one of the two serial I/O modules. (USART is also known as a Serial Communications Interface or SCI.) The USART can be configured as a full-duplex asynchronous system that can communicate with peripheral devices, such as CRT terminals and personal computers, or it can be configured as a half-duplex synchronous system that can communicate with peripheral devices, such as A/D or D/A integrated circuits, serial EEPROMs, etc.

The USART can be configured in the following modes:

- Asynchronous (full-duplex)
- Synchronous – Master (half-duplex)
- Synchronous – Slave (half-duplex)

Bit SPEN (RCSTA<7>) and bits TRISC<7:6> have to be set in order to configure pins RC6/TX/CK and RC7/RX/DT as the Universal Synchronous Asynchronous Receiver Transmitter.

The USART module also has a multi-processor communication capability using 9-bit address detection.

REGISTER 10-1: TXSTA: TRANSMIT STATUS AND CONTROL REGISTER (ADDRESS 98h)

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	R-1	R/W-0
CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D
bit 7							bit 0

bit 7 **CSRC**: Clock Source Select bit

Asynchronous mode:
Don't care.

Synchronous mode:
1 = Master mode (clock generated internally from BRG)
0 = Slave mode (clock from external source)

bit 6 **TX9**: 9-bit Transmit Enable bit
1 = Selects 9-bit transmission
0 = Selects 8-bit transmission

bit 5 **TXEN**: Transmit Enable bit
1 = Transmit enabled
0 = Transmit disabled

Note: SREN/CREN overrides TXEN in Sync mode.

bit 4 **SYNC**: USART Mode Select bit
1 = Synchronous mode
0 = Asynchronous mode

bit 3 **Unimplemented**: Read as '0'

bit 2 **BRGH**: High Baud Rate Select bit

Asynchronous mode:
1 = High speed
0 = Low speed

Synchronous mode:
Unused in this mode.

bit 1 **TRMT**: Transmit Shift Register Status bit
1 = TSR empty
0 = TSR full

bit 0 **TX9D**: 9th bit of Transmit Data, can be Parity bit

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 10-2: RCSTA: RECEIVE STATUS AND CONTROL REGISTER (ADDRESS 18h)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R-0	R-x
SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
bit 7							bit 0

bit 7 **SPEN**: Serial Port Enable bit
1 = Serial port enabled (configures RC7/RX/DT and RC6/TX/CK pins as serial port pins)
0 = Serial port disabled

bit 6 **RX9**: 9-bit Receive Enable bit
1 = Selects 9-bit reception
0 = Selects 8-bit reception

bit 5 **SREN**: Single Receive Enable bit
Asynchronous mode:
Don't care.
Synchronous mode – Master:
1 = Enables single receive
0 = Disables single receive
This bit is cleared after reception is complete.
Synchronous mode – Slave:
Don't care.

bit 4 **CREN**: Continuous Receive Enable bit
Asynchronous mode:
1 = Enables continuous receive
0 = Disables continuous receive
Synchronous mode:
1 = Enables continuous receive until enable bit CREN is cleared (CREN overrides SREN)
0 = Disables continuous receive

bit 3 **ADDEN**: Address Detect Enable bit
Asynchronous mode 9-bit (RX9 = 1):
1 = Enables address detection, enables interrupt and load of the receive buffer when RSR<8> is set
0 = Disables address detection, all bytes are received and ninth bit can be used as parity bit

bit 2 **FERR**: Framing Error bit
1 = Framing error (can be updated by reading RCREG register and receive next valid byte)
0 = No framing error

bit 1 **OERR**: Overrun Error bit
1 = Overrun error (can be cleared by clearing bit CREN)
0 = No overrun error

bit 0 **RX9D**: 9th bit of Received Data (can be parity bit but must be calculated by user firmware)

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

10.1 USART Baud Rate Generator (BRG)

The BRG supports both the Asynchronous and Synchronous modes of the USART. It is a dedicated 8-bit baud rate generator. The SPBRG register controls the period of a free running 8-bit timer. In Asynchronous mode, bit BRGH (TXSTA<2>) also controls the baud rate. In Synchronous mode, bit BRGH is ignored. Table 10-1 shows the formula for computation of the baud rate for different USART modes which only apply in Master mode (internal clock).

Given the desired baud rate and Fosc, the nearest integer value for the SPBRG register can be calculated using the formula in Table 10-1. From this, the error in baud rate can be determined.

It may be advantageous to use the high baud rate (BRGH = 1) even for slower baud clocks. This is because the Fosc/(16 (X + 1)) equation can reduce the baud rate error in some cases.

Writing a new value to the SPBRG register causes the BRG timer to be reset (or cleared). This ensures the BRG does not wait for a timer overflow before outputting the new baud rate.

10.1.1 SAMPLING

The data on the RC7/RX/DT pin is sampled three times by a majority detect circuit to determine if a high or a low level is present at the RX pin.

TABLE 10-1: BAUD RATE FORMULA

SYNC	BRGH = 0 (Low Speed)	BRGH = 1 (High Speed)
0	(Asynchronous) Baud Rate = Fosc/(64 (X + 1))	Baud Rate = Fosc/(16 (X + 1))
1	(Synchronous) Baud Rate = Fosc/(4 (X + 1))	N/A

Legend: X = value in SPBRG (0 to 255)

TABLE 10-2: REGISTERS ASSOCIATED WITH BAUD RATE GENERATOR

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on: POR, BOR	Value on all other Resets
98h	TXSTA	CSRC	TX9	TXEN	SYNC	—	BRGH	TRMT	TX9D	0000 -010	0000 -010
18h	RCSTA	SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D	0000 000x	0000 000x
99h	SPBRG	Baud Rate Generator Register								0000 0000	0000 0000

Legend: x = unknown, - = unimplemented, read as '0'. Shaded cells are not used by the BRG.

A continuación, se muestra el código del algoritmo realizado en el lenguaje de programación Mikro C.

CÓDIGO EN MICKO C

```
void main(void){
//Declaración de variables.
unsigned int Dato;
//Inicialización de puertos.
TRISB = 0;
TRISC = 0;
PORTB = 0;
PORTC = 0;
ADCON1 = 0b11000001;
// Inicio de ADC
ADC_init();
// Velocidad de lectura del controlador
UART1_Init(2400);
 //Bucle infinito.
while(1){
// Lectura del ADC
Dato = ADC_READ(0);
//.....INICIO DE LECTURA D ELAS FRECUENCIAS.....
// ROBOT INMOVIL
if (Dato<60) {
PortB=0b00000000;
}
// FRECUENCIAS BAJAS
if (Dato>=60 && Dato<108) {
PortB=0b00000001;
```

```
UART1_Write(60);
UART1_Write('3');
UART1_Write(90);
UART1_Write('7');
UART1_Write(90);
UART1_Write('8');
delay_ms(320);
UART1_Write(120);
UART1_Write('3');
UART1_Write(90);
UART1_Write('7');
UART1_Write(90);
UART1_Write('8');
delay_ms(320);
}
if (Dato>=108 && Dato<156) {
PortB=0b00000011;
UART1_Write(100);
UART1_Write('3');
UART1_Write(100);
UART1_Write('5');
delay_ms(220);
UART1_Write(140);
UART1_Write('3');
UART1_Write(140);
UART1_Write('5');
delay_ms(220);
}
// FRECUENCIAS MEDIAS
```

```
if (Dato>=156 && Dato<207) {
PortB=0b00000111;
UART1_Write(70);
UART1_Write('1');
UART1_Write(70);
UART1_Write('3');
UART1_Write(70);
UART1_Write('4');
delay_ms(180);
UART1_Write(110);
UART1_Write('1');
UART1_Write(110);
UART1_Write('3');
UART1_Write(110);
UART1_Write('4');
delay_ms(180);
}
if (Dato>=207 && Dato<259) {
PortB=0b00001111;
UART1_Write(40);
UART1_Write('1');
UART1_Write(60);
UART1_Write('2');
UART1_Write(50);
UART1_Write('3');
UART1_Write(120);
UART1_Write('4');
UART1_Write(150);
UART1_Write('5');
```

```
UART1_Write(85);
UART1_Write('6');
delay_ms(150);
UART1_Write(80);
UART1_Write('1');
UART1_Write(90);
UART1_Write('2');
UART1_Write(90);
UART1_Write('3');
UART1_Write(80);
UART1_Write('4');
UART1_Write(110);
UART1_Write('5');
UART1_Write(100);
UART1_Write('6');
delay_ms(150);
}
if (Dato>=259 && Dato<310) {
PortB=0b00011111;
UART1_Write(100);
UART1_Write('2');
UART1_Write(100);
UART1_Write('3');
UART1_Write(100);
UART1_Write('5');
UART1_Write(60);
UART1_Write('6');
delay_ms(130);
UART1_Write(140);
```

```
UART1_Write('2');
UART1_Write(140);
UART1_Write('3');
UART1_Write(140);
UART1_Write('5');
UART1_Write(85);
UART1_Write('6');
delay_ms(130);
}
// FRECUENCIAS ALTAS
if (Dato>=310 && Dato<423) {
PortB=0b00111111;
UART1_Write(70);
UART1_Write('3');
UART1_Write(110);
UART1_Write('5');
UART1_Write(120);
UART1_Write('6');
delay_ms(110);
UART1_Write(120);
UART1_Write('3');
UART1_Write(160);
UART1_Write('5');
UART1_Write(60);
UART1_Write('6');
delay_ms(110);
}
if (Dato>=423 && Dato<536) {
PortB=0b01111111;
```

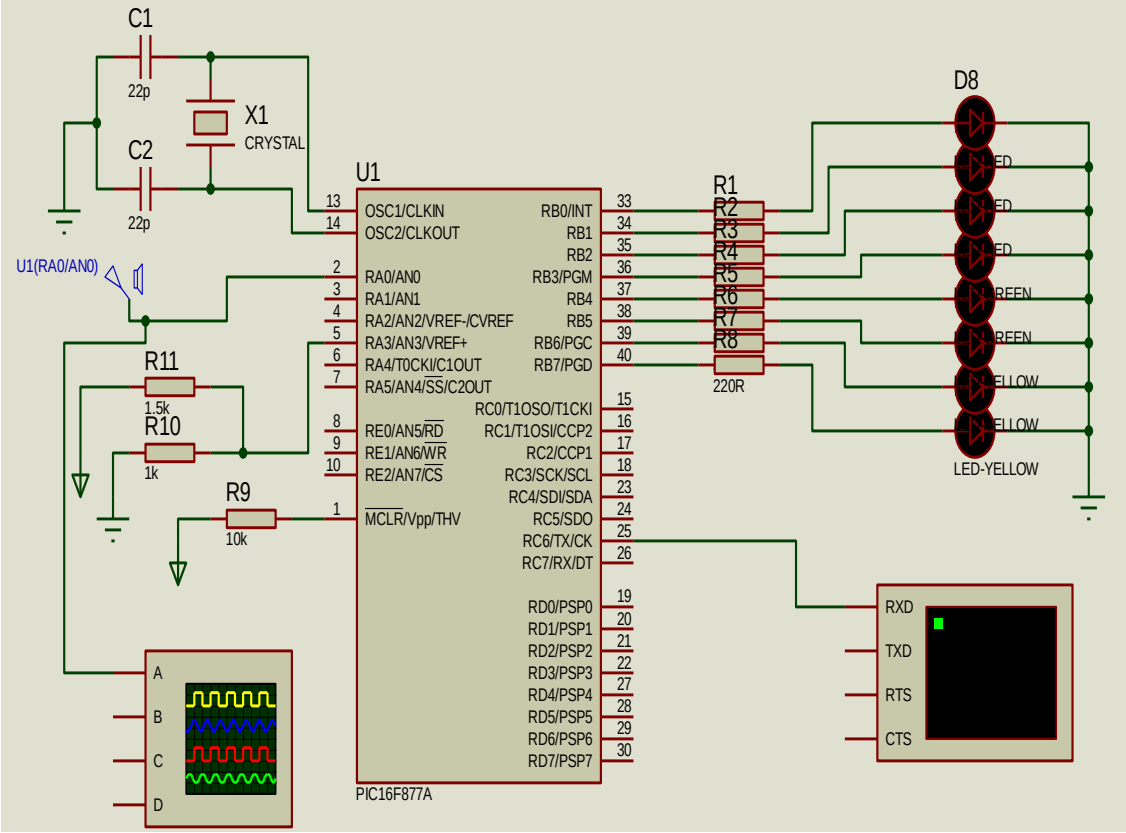
```
UART1_Write(150);
UART1_Write('1');
UART1_Write(120);
UART1_Write('2');
UART1_Write(120);
UART1_Write('3');
UART1_Write(120);
UART1_Write('4');
UART1_Write(60);
UART1_Write('5');
UART1_Write(60);
UART1_Write('6');
delay_ms(90);
UART1_Write(80);
UART1_Write('1');
UART1_Write(60);
UART1_Write('2');
UART1_Write(60);
UART1_Write('3');
UART1_Write(70);
UART1_Write('4');
UART1_Write(130);
UART1_Write('5');
UART1_Write(120);
UART1_Write('6');
delay_ms(90);
}
if (Dato>=536) {
PortB=0b11111111;
```

```
UART1_Write(50);
UART1_Write('1');
UART1_Write(100);
UART1_Write('2');
UART1_Write(150);
UART1_Write('3');
UART1_Write(120);
UART1_Write('4');
UART1_Write(80);
UART1_Write('5');
UART1_Write(40);
UART1_Write('6');
delay_ms(80);
UART1_Write(90);
UART1_Write('1');
UART1_Write(70);
UART1_Write('2');
UART1_Write(70);
UART1_Write('3');
UART1_Write(60);
UART1_Write('4');
UART1_Write(150);
UART1_Write('5');
UART1_Write(150);
UART1_Write('6');
delay_ms(80);
}
}
}
```

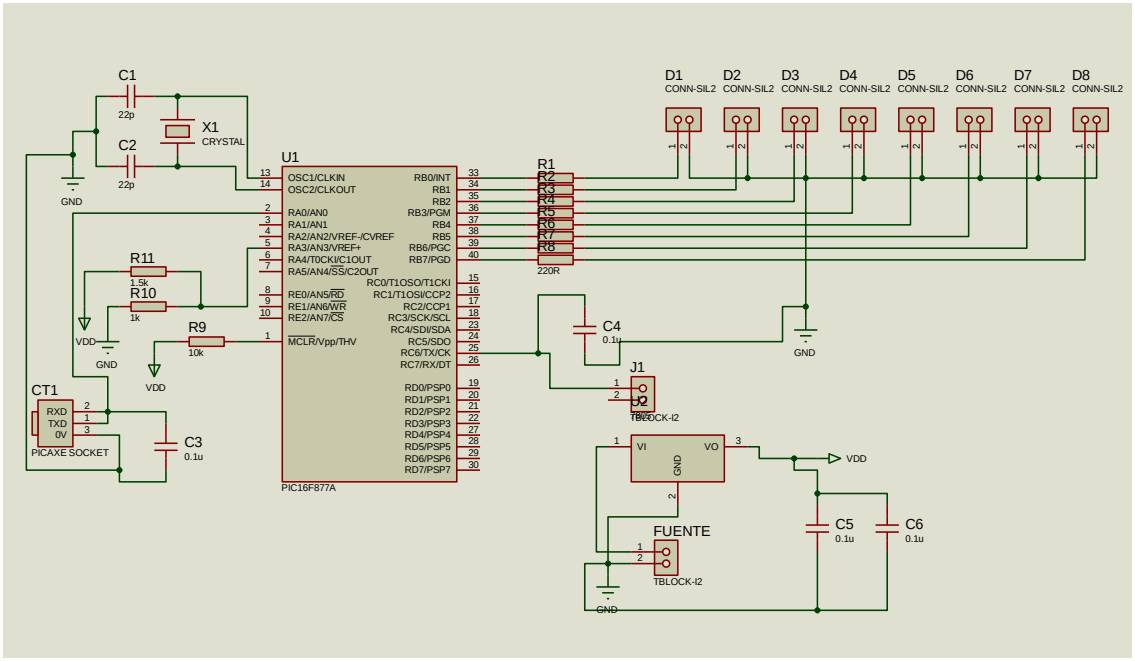
ANEXO C

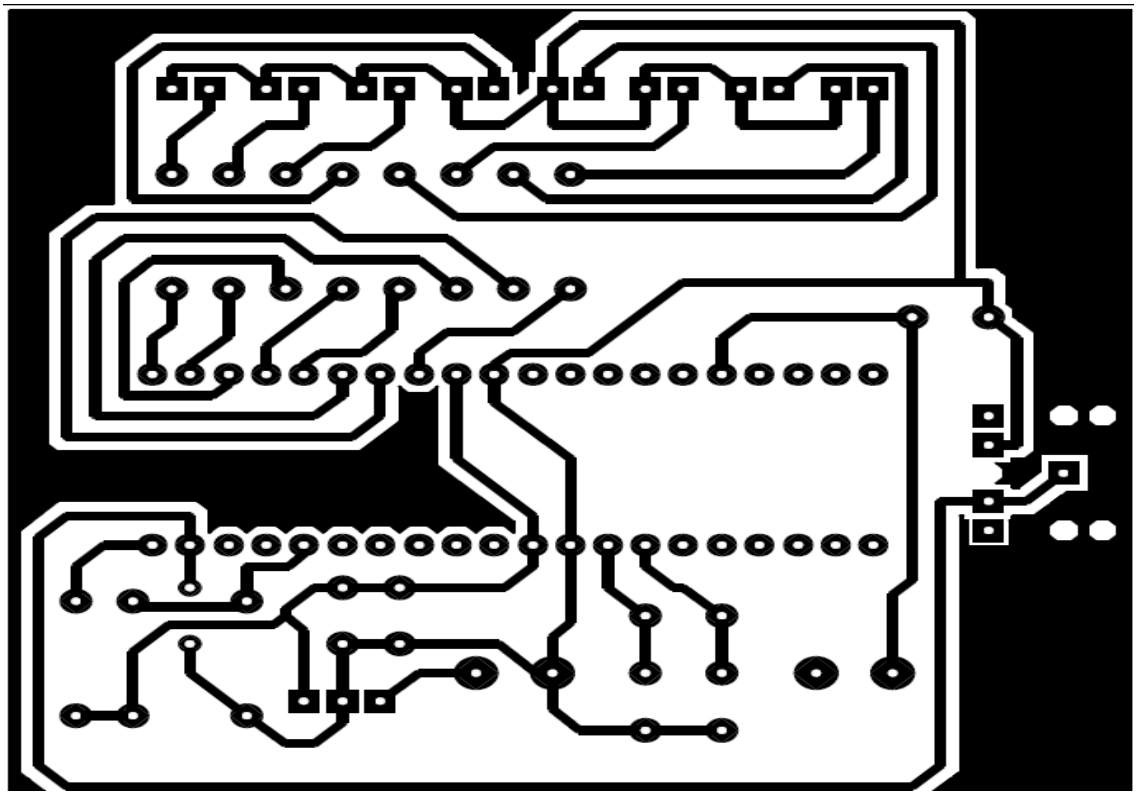
A continuación, se muestra la simulación y el diseño hecho en Isis y ares de la placa que conforma el reconocedor de frecuencias de audio.

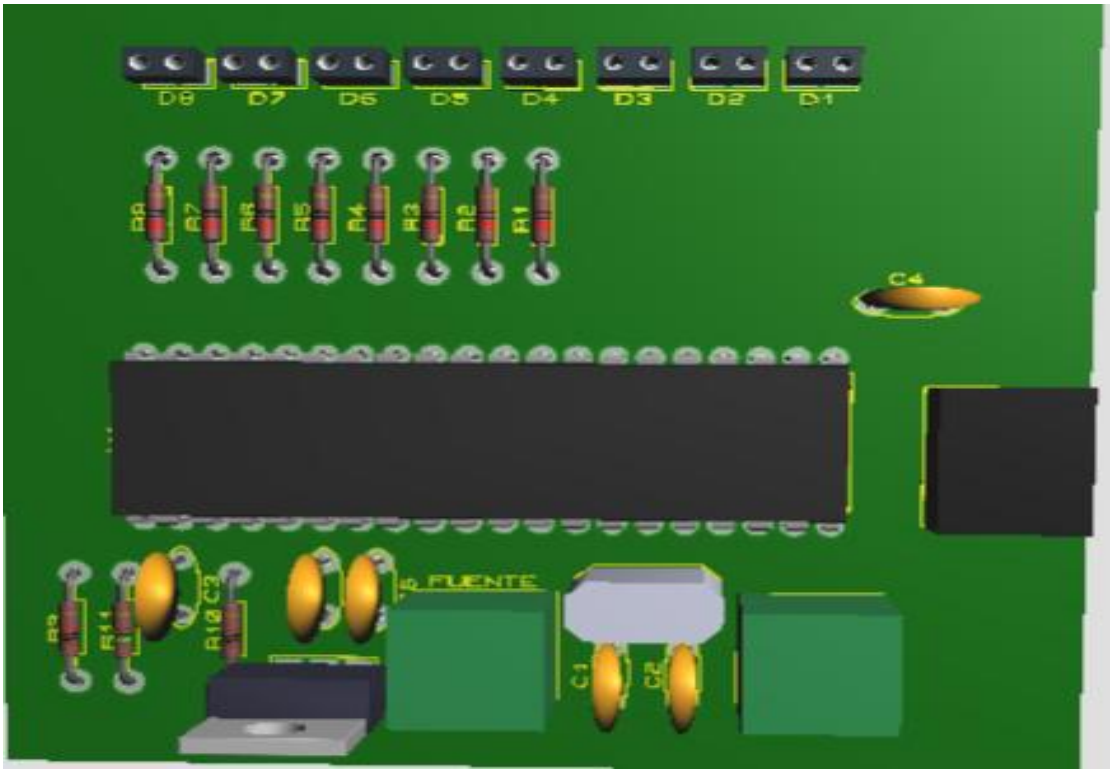
SIMULACIÓN DEL RECONOCEDOR DE FRECUENCIAS DE AUDIO



DISEÑO DEL RECONOCEDOR DE FRECUENCIAS DE AUDIO







RECONOCEDOR DE FRECUENCIAS DE AUDIO PARA UN ROBOT BAILARÍN