



**FACULTAD DE INGENIERÍA Y CIENCIAS AGROPECUARIAS**

**ESCUELA DE INGENIERÍA DE SONIDO Y ACÚSTICA**

**DISEÑO Y ELABORACIÓN DE UN DISPOSITIVO CONVERSION DE SEÑAL  
DE NIVEL DE LÍNEA A INFORMACIÓN DE MENSAJES DE CONTROL MIDI.**

**Trabajo de Titulación presentado en conformidad a los requisitos**

**establecidos para obtener el título de:**

Ingeniero en Sonido y Acústica

**Profesor Guía:**

Ing. Marcelo Lazzati

**Autor:**

Christian Fernando Moreira Sosa

**Año**

2010

## **DECLARACIÓN DEL PROFESOR GUÍA**

“Declaro haber dirigido este trabajo a través de reuniones periódicas con el estudiante, orientando sus conocimientos para un adecuado desarrollo del tema escogido, y dando cumplimiento a todas las disposiciones vigentes que regulan los Trabajos de Titulación.”

Marcelo Lazzati  
Ingeniero en Ejecución de Sonido  
CI: 171163573-8

### **DECLARACIÓN DE AUTORÍA DEL ESTUDIANTE**

“Declaro que este trabajo es original, de mi autoría, que se han citado las fuentes correspondientes y que en su ejecución se respetaron las disposiciones legales que protegen los derechos de autor vigentes.”

Christian Fernando Moreira Sosa

CI: 171691766-9

## **AGRADECIMIENTOS**

Agradezco profundamente a las siguientes personas quienes con su apoyo y consejos me han animado e inspirado a llevar a buen fin este proyecto:

Señor Jesucristo, eres mi guía, mi inspiración todo lo que tengo y soy te lo debo a ti.

Mis amados padres, son mis mejores amigos y consejeros, gracias por estar ahí siempre.

Marcelo Lazzati, tu acertada guía y consejos durante la carrera y el proyecto.

Verónica Moreira, tu apoyo ha sido incondicional, sin él me hubiese sido imposible llegar a la meta.

Raúl y Ximena Portilla, no hay palabras para agradecer todo el apoyo que me han dado.

A todos, muchísimas gracias.

## **DEDICATORIA**

Dedico este trabajo a:

René Moreira, mi padre, su vida ha sido un ejemplo de superación y constancia en medio de la adversidad.

Carmita Sosa, nadie puede animarme en momentos difíciles como usted lo hace, este trabajo es dedicado a usted.

Mis amadas hermanas Ximena y Verito, gracias por todo su apoyo material, emocional y espiritual, las amo con todo mi corazón.

## RESUMEN.

El presente trabajo de investigación, se basa en el protocolo MIDI (Interfaz Digital de Instrumentos Musicales), para desarrollar un dispositivo que convierta señal de nivel de línea, en mensajes de control MIDI.

El dispositivo se basa en la tecnología de microcontroladores, los cuales han sido programados bajo los lineamientos de la norma MIDI 1.0.

Los parámetros a tomar en cuenta para la conversión son la frecuencia y la amplitud de la señal, para lo que dentro de la programación del dispositivo se incluye un frecuencímetro para la detección de frecuencia, y un voltímetro para la medición de picos de amplitud y determinar así la velocidad de la nota MIDI.

Los controles del dispositivo son un *switch* de encendido y apagado, un *DIP SWITCH*, de cuatro puertos para el control de los canales MIDI; adicionalmente se ha implementado un *display* de cristal líquido (LCD), que proporciona información de canal, frecuencia y número de nota MIDI en tiempo real.

Para las pruebas del dispositivo han sido utilizadas muestras de audio de las distintas familias de instrumentos (cuerdas, guitarra eléctrica y acústica, bajo, piano, maderas, metales).

El dispositivo logra la conversión de señal de nivel de línea en mensajes de control MIDI en términos de frecuencia y amplitud.

## **ABSTRACT.**

The present investigation work is based on MIDI protocol (Musical Instrument Digital Interface), to develop a device that converts line level signal, to MIDI control messages.

The device is based on microcontrollers' technology, which have been programmed according to MIDI 1.0 norm lineament.

The parameters to take into account for signal conversion are frequency and amplitude for this purpose a frequency meter for frequency detection and a voltmeter for amplitude peaks measurement was included in device program for the MIDI note and velocity determination.

The device controls are the On/Off switch, a four ports DIP SWITCH for MIDI channels control; additionally it has been implemented a liquid crystal display (LCD), which provides channel, frequency and MIDI note real time information.

For device testing it have been used audio samples from different instrument families (strings, acoustic and electrical guitar, bass, piano, woods and brass).

The device gets the conversion of line level signal to MIDI control messages in frequency and amplitude terms.

## ÍNDICE

Contenido	Página
<b>1. INTRODUCCIÓN</b>	<b>1</b>
1.1. Descripción del Problema	1
1.2. Antecedentes	2
1.3. Justificación	3
1.4. Objetivos	3
1.4.1. Objetivo General	3
1.4.2. Objetivos Específicos	4
1.5. Hipótesis	4
<b>2. TEORÍA MIDI</b>	<b>5</b>
2.1. Reseña Histórica	5
2.2. Norma MIDI	8
2.3. Conectores	9
2.4. Sistema de Numeración Binario y Hexadecimal	12
2.5. Tipos de Bytes	13
2.5.1. Byte de Estado	13
2.5.2. Byte de Datos	14
2.5.3. Running Status	14
2.6. Canales MIDI	14
2.6.1. Mensajes de Canal	16
2.6.1.1. Mensajes de Voz	16
2.6.1.1.1. Note Off (Apagar la Nota)	17
2.6.1.1.2. Note On (Prender la Nota)	17
2.6.1.1.3. Poliphonic Key Pressure	17
2.6.1.1.4. Control Change	18

2.6.1.1.4.1. Controladores Continuos	18
2.6.1.1.4.2. Controlador de Switch	19
2.6.1.1.5. Program Change	19
2.6.1.1.6. Channel Pressure	20
2.6.1.1.7. Pitch Bend Change	20
2.6.1.2. Channel Mode (Mensajes de Modo)	21
2.6.1.2.1. Reset all Controllers	21
2.6.1.2.2. Local Control	22
2.6.1.2.3. All Notes Off	23
2.6.1.2.4. Omni Mode Off	23
2.6.1.2.5. Omni Mode On	24
2.6.1.2.6. Mono Mode On (Poly Mode Off)	24
2.6.1.2.7. Poly Mode On (Mono Mode Off)	25
2.6.2. Mensajes de Sistema	28
2.6.2.1. Mensajes de Sistema Común	28
2.6.2.1.1. Song Position Pointer	29
2.6.2.1.2. Song Select	29
2.6.2.1.3. Tune Request	30
2.6.2.2. Mensajes de Sistema de Tiempo Real	30
2.6.2.2.1. Timing Clock	30
2.6.2.2.2. Start	30
2.6.2.2.3. Continue	31
2.6.2.2.4. Stop	31
2.6.2.2.5. Active Sensing	31
2.6.2.2.6. System Reset	31
2.6.2.3. System Exclusive – SYSEX	32
2.6.2.3.1. Sample Dump Standard	33

2.6.2.3.2. Standard MIDI File	34
2.6.2.3.3. MIDI Time Code	34
<b>2.7. Microcontroladores</b>	<b>34</b>
2.7.1. Estructura de un Microcontrolador	35
2.7.1.1. Arquitectura Von Neumann	36
2.7.1.2. Arquitectura Harvard	36
2.7.1.3. Registros	37
2.7.1.4. Unidad de Control	37
2.7.1.5. Unidad Aritmético Lógica	38
2.7.1.6. Buses	38
2.7.1.6.1. Bus de Dirección	38
2.7.1.6.2. Datos	39
2.7.1.6.3. Control	39
2.7.1.7. Conjunto de Instrucciones	39
2.7.1.7.1. Instrucciones CISC	39
2.7.1.7.2. Instrucciones RISC	39
2.7.1.8. Memoria	40
2.7.1.8.1. Memoria RAM	40
2.7.1.8.2. Memoria ROM	41
2.7.1.8.2.1. Memoria ROM de Máscara	41
2.7.1.8.2.2. Memoria PROM ó OTP	41
2.7.1.8.2.3. Memoria EPROM	42
2.7.1.8.2.4. Memoria EEPROM	42
2.7.1.8.2.5. Memoria Flash	43
2.7.1.9. Periféricos	43
2.7.1.9.1. Entradas y Salidas	43
2.7.1.9.2. Temporizadores y Contadores	43

2.7.1.9.3. Conversor Análogo- Digital (A/D)	44
2.7.1.9.4. Puerto Serie	44
2.7.1.9.5. Comparadores	45
2.7.2. Microcontrolador PIC16F877A	45
2.7.3. Circuito Integrado LM324	46
2.7.4. Circuito Integrado 40106	46
<b>2.8. Tecnología de Conversión</b>	<b>47</b>
2.8.1. Señal Analógica	47
2.8.2. Señal Digital	48
2.8.2.1. Muestreo (Sampling)	48
2.8.2.2. Cuantización	49
<b>2.9. Transmisión de Información</b>	<b>50</b>
2.9.1. Transmisión Análoga	50
2.9.2. Transmisión Digital	51
2.9.3. Modos de Transmisión	51
2.9.3.1. Según la Dirección de Intercambios	51
2.9.3.1.1. Conexión Simple	51
2.9.3.1.1.1. Conexión Semidúplex	52
2.9.3.1.1.2. Conexión Dúplex Total	52
2.9.3.2. Modo de Transmisión en Serie y Paralelo	53
2.9.3.2.1. Conexión en Paralelo	53
2.9.3.2.2. Conexión en Serie	53
2.9.3.3. Transmisión Sincrónica y Asincrónica	54
2.9.3.3.1. Conexión Asincrónica	54
2.9.3.3.2. Conexión Sincrónica	54
<b>3. DISEÑO</b>	<b>56</b>
3.1. Alimentación	56

<b>3.2. Descripción del Circuito</b>	<b>56</b>
<b>3.3. Programación</b>	<b>58</b>
3.3.1. Subrutina CANALASIG	59
3.3.2. Subrutina NUMERODENOTA	60
3.3.3. Subrutina VELOCITY	60
3.3.4. Programa Principal INICIO	61
<b>4. DESARROLLO EXPERIMENTAL</b>	<b>62</b>
<b>4.1. Diseño del Dispositivo</b>	<b>62</b>
<b>4.2. Pruebas</b>	<b>63</b>
4.2.1. Prueba Rítmica	65
4.2.1.1. Resultados de la Prueba Rítmica	65
4.2.1.2. Análisis de Resultados Prueba Rítmica	66
4.2.2. Prueba de Altura Tonal	66
4.2.2.1. Resultados Altura Tonal (Escala C Mayor)	67
4.2.2.2. Análisis Altura Tonal (Escala C Mayor)	68
4.2.2.3. Resultados Altura Tonal (Cromática)	69
4.2.2.4. Análisis Altura Tonal (Cromática)	70
4.2.2.5. Resultados Altura Tonal (Arpeggios y Acordes)	71
4.2.2.6. Análisis Altura Tonal (Arpeggios y Acordes)	72
4.2.3. Prueba Dinámica (Velocidad)	72
4.2.3.1. Resultados Prueba Dinámica	73
4.2.3.2. Análisis Prueba Dinámica	74
4.2.3.3. Mediciones Adicionales	74
<b>4.3. Resumen de Resultados</b>	<b>75</b>
<b>5. ESTUDIO ECONÓMICO DEL PROYECTO</b>	<b>76</b>
<b>5.1. Diseño y Desarrollo</b>	<b>76</b>

5.2. Pruebas	78
5.3. Relación Costo – Beneficio	79
<b>6. IMPLEMENTACIONES FUTURAS AL PROYECTO</b>	<b>81</b>
6.1. Micrófono Dinámico	81
6.2. Teclado Hexadecimal y Display LCD	81
6.3 Variación del Tempo	82
6.4. Metrónomo	82
6.5. Variación de la Afinación del Instrumento	83
6.6. Memoria Flash de Almacenamiento de Datos	83
<b>7. CONCLUSIONES Y RECOMENDACIONES</b>	<b>84</b>
7.1. Conclusiones	84
7.1.1. Conclusión General	84
7.1.2. Conclusiones de Diseño y Desarrollo Experimental	84
7.1.3. Conclusiones de Operación	86
7.1.4. Conclusiones de la Etapa de Pruebas	87
7.1.4.1. Conclusiones Resultados Prueba Rítmica	87
7.1.4.2. Conclusiones Resultados Altura Tonal	88
7.1.4.3. Conclusiones Resultados Prueba Dinámica	90
7.2. Recomendaciones	91
<b>8. BIBLIOGRAFÍA</b>	<b>92</b>
<b>9. ANEXOS</b>	<b>94</b>
Anexo 1: Tablas de Mensajes MIDI	94
Anexo 2: Valores de Controladores	96
Anexo 3: Notas MIDI, Byte de Estado, Byte de Datos, Frecuencia y Período	97

<b>Anexo 4: Partituras del Banco de Pruebas</b>	<b>100</b>
<b>Anexo 5: Diagrama Esquemático del Dispositivo</b>	<b>104</b>
<b>Anexo 6: Diseño del Circuito Impreso</b>	<b>105</b>
<b>Anexo 7: Programa del Microcontrolador PIC16F877A</b>	<b>106</b>
<b>Anexo 8: Glosario de Términos</b>	<b>121</b>

## **1. INTRODUCCIÓN.**

### **1.1. Descripción del Problema:**

Las representaciones artísticas han sido desde tiempos inmemorables parte del desarrollo cultural de los pueblos, con distintos tipos de significados: ritual, ceremonial o simplemente por entretenimiento las representaciones artísticas han marcado un hito en los pueblos.

Más específicamente las representaciones musicales siempre han sido el deleite del ser humano, desde reyes, pasando por sabios, militares y campesinos han disfrutado de la música; y antes del aparecimiento de las primeras grabaciones el único modo de disfrutar del don de la música era a través de las representaciones artísticas en vivo.

Conforme fue avanzando la tecnología, y con ella el desarrollo instrumental estas representaciones fueron incrementándose en complejidad tanto estética como técnica; siendo así un placer del que disfrutaban solamente las clases socio-económicas más altas, ya que se volvió imposible viajar o mantener una orquesta entera solo para el disfrute de la monarquía.

Con el desarrollo de la electrónica analógica, y con ella la síntesis sonora, aparecieron instrumentos que simulaban los sonidos de los instrumentos acústicos, aunque esto fue un avance muy significativo tanto para las representaciones musicales en vivo como para las grabaciones, la movilización de estos instrumentos era muy difícil y costosa.

En el año de 1983 los ingenieros de la empresa Californiana Sequential Circuits Dave Smith y Chet Wood presentaron en la reunión anual de la AES un proyecto en el que presentaban una interfaz universal para instrumentos electrónicos denominada MIDI (Musical Instrument Digital Interface). A partir de este punto, la industria musical da un importante giro en su desarrollo. Poco a poco se fueron desarrollando nuevos dispositivos como secuenciadores

(dispositivos destinados al registro y almacenamiento de mensajes de control MIDI con opciones de edición para su posterior reproducción), cajas de ritmos (drum machines), samplers (dispositivos que almacenan muestras de audio para ser reproducidas mediante un controlador MIDI – teclado controlador –), módulos de sonido, etc.

El problema de todo este desarrollo tecnológico es que se sacrificó muchísimo la comodidad del músico, que debía conformarse con la frialdad de instrumentos completamente electrónicos en lugar de la calidez obtenida con los instrumentos acústicos tradicionales ofrecían mayor portabilidad pero su sonido era demasiado sintético y para algunas aplicaciones musicales inútiles.

Es aquí donde nace el deseo de satisfacer las necesidades interpretativas de los músicos quienes puedan tener al alcance de su concepción estética ambos mundos: por un lado la comodidad y la calidez que brinda un instrumento acústico tradicional y por otro el acceso a las posibilidades de la música electrónica a través de la manipulación de su propia interpretación mediante los dispositivos descritos anteriormente, potenciando las posibilidades interpretativas, artísticas, estéticas y técnicas de los músicos.

## **1.2. Antecedentes:**

La principal aplicación del prototipo a desarrollar es la conversión de señales de audio en mensajes MIDI; dispositivo pensado para músicos de instrumentos acústicos que por la naturaleza de sus instrumentos no han podido experimentar con esta tecnología, pero que sin embargo quiere experimentar nuevas sonoridades a través de la tecnología MIDI, además de esta área, es útil en la difusión musical ya que al convertir las señales de audio en mensajes de control MIDI estos son fácilmente reconocibles por cualquier editor de partituras para transcribir la interpretación en notación musical tradicional.

La presente investigación va a estar basada en el desarrollo de un dispositivo que sea adaptable a instrumentos reales y que por las variaciones de voltaje generados por estos sea convertidos a mensajes MIDI los cuales puedan ser manipulados con facilidad en cualquier secuenciador, sampler (sea hardware o software).

En el desarrollo del dispositivo se espera llegar a detectar los parámetros de amplitud y frecuencia de una señal de audio de nivel de línea a sus mensajes MIDI respectivos; dejando abierto el tema para que en posteriores investigaciones se pueda completar e implementar otras facetas en el desarrollo del dispositivo.

### **1.3. Justificación:**

Los principales beneficios que se espera del desarrollo del dispositivo son la comodidad y facilidad para el músico acústico (que usa instrumentos completamente acústicos), pueda experimentar de una manera más abierta y cómoda, sin sacrificar la interpretación (al tocar en un instrumento MIDI de bajas prestaciones interpretativas) además del desarrollo interpretativo se debe tomar en cuenta el desarrollo artístico que esto traerá como consecuencia.

Otra posible aplicación es la edición y almacenamiento de estos mensajes como partituras, tablaturas y otros medios de notación musical, para la difusión y almacenamiento de nuevas propuestas estéticas (tales como improvisaciones que solamente se dan una vez y por la naturaleza de esta técnica compositiva llega a ser prácticamente irreplicable).

### **1.4. Objetivos:**

#### **1.4.1 Objetivo General:**

Diseñar de un dispositivo que convierta señales de línea monofónica en mensajes de control MIDI.

#### **1.4.2 Objetivos Específicos:**

- Desarrollar el dispositivo, siguiendo las etapas de programación de los dispositivos, y el ensamblaje de los mismos.
- Evaluar y probar el dispositivo con la ayuda de señales de prueba de instrumentos musicales que incluyan escalas, arpeggios y acordes para comprobar la correcta conversión en el rango de frecuencias del instrumento interpretado, además de la velocidad (intensidad en el lenguaje analógico) en un rango de frecuencias comprendido entre 50 [Hz] a 6 [KHz], por medio de la comparación con las partituras de los ejercicios interpretados, y su uso en las aplicaciones expuestas anteriormente.

#### **1.5. Hipótesis:**

En base a la investigación y al prototipo a ser desarrollado se asume como hipótesis de este proyecto que es posible la correcta de detección y conversión de una señal de audio de nivel de línea a mensajes de control MIDI, tomando en cuenta los parámetros de frecuencia y amplitud dentro del rango de frecuencias que es capaz de generar un instrumento dado desde de 50 [Hz] a 6 [KHz].

## **2. TEORÍA MIDI.**

### **2.1. Reseña Histórica:**

La tecnología MIDI (*Musical Instrument Digital Interface*), nace de la necesidad de intercomunicar instrumentos musicales electrónicos, de distintas marcas.

El desarrollo de la tecnología MIDI viene de la mano con el desarrollo y digitalización de los hoy conocidos como sintetizadores (el término no fue usado hasta la década de 1950), aunque se conocen prototipos de dispositivos como tales como el “Telharmonium” o el “Singing Telegraph” desde finales del siglo XIX, es decir, el desarrollo de los instrumentos musicales electrónicos comenzó prácticamente a raíz del apareamiento de la energía eléctrica.

Posteriormente en las décadas de 1960 y 1970 con el gran aporte de pioneros como Robert Moog, Don Buchla, Harold Bode, Pete Zinovieff, y Dave Cockerell se desarrollaron los primeros sintetizadores como los conocemos hoy en día, la industria de la música inmediatamente adoptó estos nuevos instrumentos debido a las posibilidades tímbricas que tenían, aunque su estabilidad, notas de polifonía (los primeros modelos eran exclusivamente monofónicos) y compatibilidad entre marcas era nula, sin contar con que en un principio los sintetizadores eran tan caros y tan grandes que solamente podían ser costeados por los grandes presupuestos de las más importantes universidades.

A finales de la década de 1960 y durante de la década de 1970 se crearon modelos más accesibles para el usuario común (el modelo más barato el Prophet 5 de la Sequential Circuits costaba \$ 4000), conforme avanzaba el desarrollo de la tecnología, aumentaban en estabilidad, estabilidad y polifonía, y es a finales de la década de 1970 que la industria de los desarrolladores de sintetizadores da el primer gran salto en lo que a control externo e intercomunicación se refiere, que es la inclusión del control de tono (pitch) y de

duración (time) que se logró a través un control de corriente continua: el Control *Voltage Gate* (Puerta de Voltaje), que permitía controlar a través de un voltaje de corriente continua, la nota (número de tecla) y duración de la misma (cuando dejaba de alimentar este voltaje de corriente continua, dejaba de sonar la tecla); aunque era un control bastante limitado entre dispositivos fue el inicio para el desarrollo de sistemas de comunicación entre dispositivos, fruto de esto la Roland desarrolló su DCB (por sus siglas en inglés Digital Communication Bus – Bus de Comunicación Digital) que fue una interfaz de intercambio de archivos entre productos Roland, fue lanzado al mercado en 1982 y consistía básicamente en una interfaz muy parecida a MIDI pero con la diferencia que DCB solamente transmitía mensajes de note on/off y mensajes de cambio de programa en los primeros dispositivos como el Roland Juno-60 y el Roland Jupiter-8 se usaban cables con sockets de 14 pines, posteriormente reemplazados por el conector DIN de 5 pines que luego se tomaría como conector estándar para la transmisión de mensajes de control en la norma MIDI; cuando se estableció MIDI como norma, la Roland lanzó al mercado la denominada MD-8, que era un dispositivo en forma de una caja negra capaz de convertir mensajes MIDI en DCB y viceversa, especialmente útiles para los dispositivos mencionados anteriormente, Roland también lanzó dos secuenciadores con tecnología DCB: el JSQ-60 y el MSQ-700, este último era capaz de transmitir y recibir mensajes de control MIDI y también mensajes DCB aunque no podía hacer las veces de conversor entre los dos estándares, tampoco podía usar ambos estándares a la vez. A principios de la década de 1980 DCB fue reemplazada por MIDI.

La DCB no fue el único intento de interconexión entre dispositivos; la Yamaha desarrollo *Key Code Interface*.

Es así como en 1981 Dave Smith de Sequential Circuits y Ikutaru Takehashi de Roland se unieron para resolver el problema de la intercomunicación de dispositivos de diferentes marcas, entonces en la feria de la AES de ese mismo año presentaron un documento mostrando un sistema de

comunicación digital universal para dispositivos, cuya velocidad de transmisión de datos era de 19.2 [Kbps] y usaba un plug tradicional de audífonos (TRS de 1/4"). Para la feria de la NAMM en Enero de 1982 presentaron dos innovaciones importantes en el sistema: el primero era un incremento de la velocidad de transmisión de datos de 19.2 [Kbps] a 31.25 [Kbps], la segunda innovación era el cambio de plug TRS de 1/4" a un circuito de aislamiento óptico (el conector DIN de 5 pines que fuera usado por DCB de Roland en primer lugar); había nacido MIDI.

*Musical Instrument Digital Interface* (MIDI) fue anunciada al público en el año de 1982 y a principios de 1983 apareció por primera vez en un dispositivo: el Prophet-600, fue seguido por el Roland JX3P, las primeras pruebas de interconexión de dos dispositivos de distintas marcas, se realizó precisamente con estos dos dispositivos y fue exitosa en todas las pruebas, nunca más la industria de la electrónica musical volvería a ser la misma. Inicialmente MIDI comprendía de un documento de ocho páginas, donde se indicaba el funcionamiento básico del sistema; se indicaban cosas tales como la interconexión de dos dispositivos, como manipular el volumen, etc. Rápidamente se estableció la necesidad de crear una norma que abarcara tareas más complicadas como el control completo de un sintetizador a partir de otro (sistema maestro-esclavo), interconexión de nuevos dispositivos que en la época estaban viendo la luz, como secuenciadores, cajas de ritmo, etc.

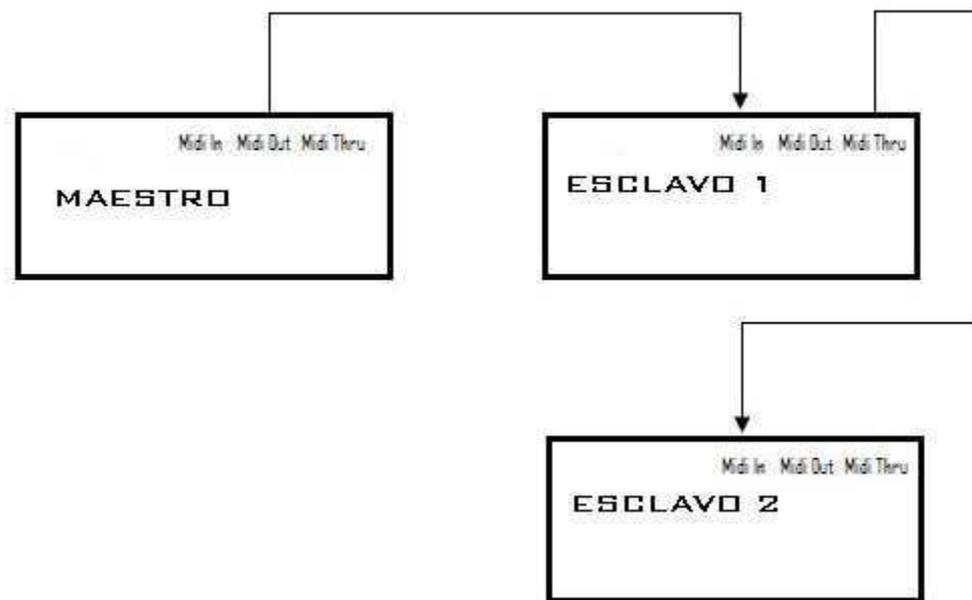
Cabe recalcar que la norma MIDI solamente ha tenido una revisión, no para corregirla, sino para establecer actualizaciones, de donde sale la norma MIDI 2.0 el año de 1999 y aunque con el paso de los años se han ido aumentando nuevos elementos como por ejemplo el formato estándar MIDI para archivos de canciones: General MIDI en el año de 1991 y el posterior General MIDI 2.0, además de la adición de distintos modos de conexión además del conector DIN de 5 pines como USB, FireWire, y Wi-Fi, además de una infinidad de aplicaciones en video juegos, telefonía celular, aplicaciones de animación virtual y un sin fin de aplicaciones; el funcionamiento de MIDI no ha

cambiado en nada desde 1983, este avance se ha dado gracias a la política libre de regalías de la interfaz que ha permitido ha desarrolladores de todo el mundo, tanto las grandes compañías como los usuarios independientes agregar y desarrollar aplicaciones basados en la tecnología MIDI, es impensable sin la tecnología MIDI el desarrollo que la industria de la electrónica musical ha tenido, la facilidad de interconexión de dispositivos, y la infinidad de aplicaciones con las que hoy en día contamos.

## 2.2. Norma MIDI:

MIDI (Musical Instrument Digital Interface) utiliza un protocolo:

- **Digital:** Utiliza el sistema de numeración binario, es decir, todos sus mensajes son codificados a partir de 1 y 0 para su posterior interpretación y decodificación en el dispositivo receptor, de tal manera que la velocidad de transmisión de datos es suficientemente elevada (31.25 [Kbps]) para transmitir los datos que el protocolo requiere sin problemas graves de latencia con una tolerancia de +/- 1%.
- **Serial:** Las conexiones inicialmente se las pensaron de manera serial, con el fin de ahorrar cableado en conexiones más complejas, aunque el protocolo de por sí no admite conexiones en paralelo, existen en el mercado dispositivos de conexión en paralelo MIDI llamadas “Cajas de Enlace Directo” (*MIDI Thru Boxes*), además la transmisión de datos es en un solo sentido, es decir, por un cable solamente se pueden enviar o recibir datos, no ambos a la vez.
- **Asíncrono:** El protocolo MIDI como tal no requiere de un código de sincronismo o time code, como es el caso de algunos dispositivos de audio (tanto analógico como digital), el protocolo utiliza un sistema de pulsos de reloj, para sincronizar dispositivos entre sí.



**Figura 2.1 Esquema de Conexión en Serie Dispositivos MIDI.**

### 2.3. Conectores:

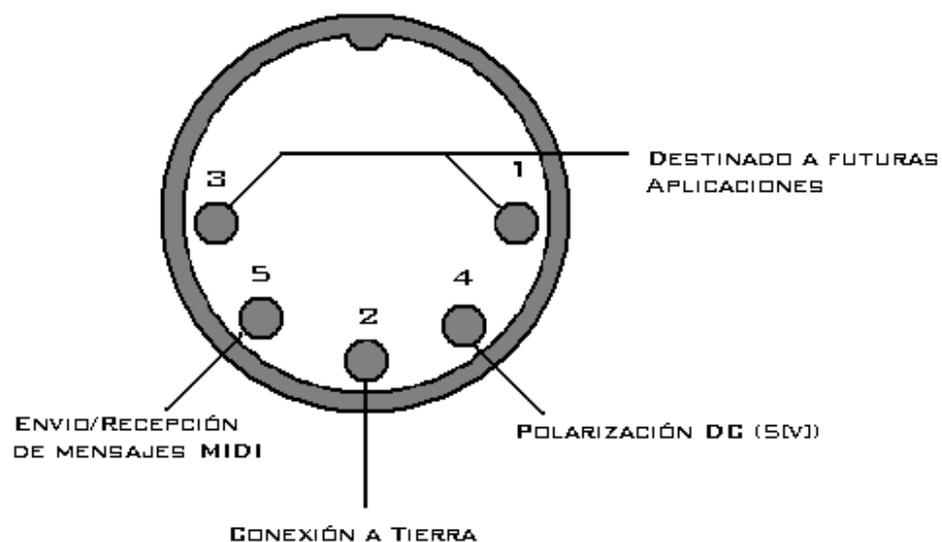
El protocolo MIDI utiliza varios tipos de conectores, tales como USB, FireWire, Wi-Fi; sin embargo el conector estándar de conexión es el DIN de 5 pines, este tipo de conector fue desarrollado y estandarizado por la *Deutches Institut für Normung* (DIN) o su traducción Instituto de Normalización Alemán, existen gran cantidad de conectores DIN, dependiendo su forma para la aplicación que se vaya a utilizar, por ejemplo se usaban para la transmisión de video analógico y en interfaces digitales como por ejemplo lo que por mucho tiempo fue estándar en el mundo de las computadoras personales para conexión de dispositivos periféricos como son el caso de teclados y mouses: el estándar PS/2 usa un conector DIN de cinco pines.

Existen siete tipos de conectores DIN, que van desde tres pines hasta ocho; que tienen características similares como por ejemplo todos los conectores macho tienen en su extremo una capucha de metal de 13.2 [mm] de diámetro para protección de los pines que sobresalen, además poseen bordes especiales para evitar la manipulación y conexión incorrecta de los dispositivos causando daños tanto al dispositivo emisor de información como al receptor.

Los estándares más conocidos son: DIN 41524 (conectores de tres y cinco pines), DIN 45322 (conectores de cinco pines a 60°), DIN 45326 (conectores de ocho pines), DIN 45329 (conectores de siete pines), estos tipos de conectores DIN varían de acuerdo a sus diferentes aplicaciones.

Dentro de los conectores DIN 41524 y DIN 45322 (ambos de cinco pines) la diferencia principal es la disposición de los pines: los conectores DIN de cinco pines de 180° (DIN 41524) y los conectores de cinco pines de 240° (DIN 45322), siendo el conector DIN de cinco pines de 180° el que se ha estandarizado como el conector para el protocolo MIDI.

En el siguiente diagrama se detalla la función de cada uno de los pines en la recepción y transmisión de información desde y hacia los dispositivos que utilizan el protocolo de comunicación MIDI.



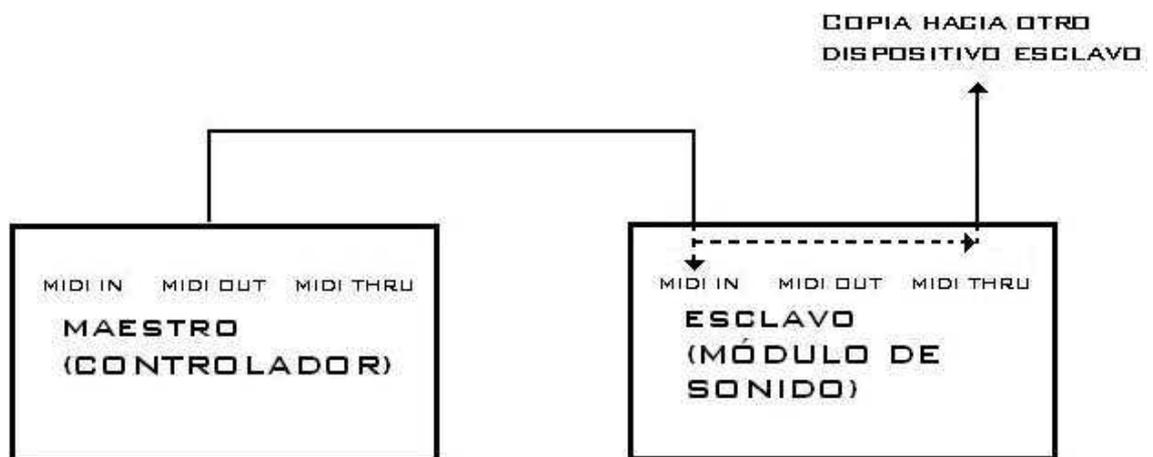
**Figura 2.2 Conector DIN 5 Pin.**

- Los conectores 1 y 3 no tienen aplicación, fueron dejados ahí para futuras aplicaciones.
- El conector número 2 tiene la función de envío a tierra.
- La información de envío/recepción MIDI es enviada por el pin 5.

- El pin 4 sirve para la polarización DC del circuito (5 [V])

En función de la aplicación, los conectores MIDI son tres:

1. **MIDI In:** Por este conector se recibe la información de los mensajes de control MIDI procedentes de un dispositivo maestro/controlador, a través del conector MIDI Out ó MIDI Thru.
2. **MIDI Out:** Por este conector se envía la información de los mensajes de control MIDI hacia un dispositivo esclavo, al conector MIDI In.
3. **MIDI Thru:** Este conector hace una copia en paralelo de la información de los mensajes de control MIDI recibidos en MIDI In desde un dispositivo maestro/controlador para enviarlos hacia otro dispositivo esclavo.



**Figura 2.3 Conexiones MIDI In, Out y Thru.**

Las principales aplicaciones del conector MIDI Thru es para la interconexión de varios dispositivos esclavos para ser controlados desde un solo dispositivo maestro; un tipo de conexión de dispositivos muy usado es el encadenado Daisy que consiste en la interconexión de varios dispositivos esclavos controlados por un dispositivo maestro, dicha interconexión se hace desde MIDI Out del dispositivo maestro, hacia MIDI In del primer dispositivo esclavo; a su vez del primer dispositivo esclavo se saca una conexión desde MIDI Thru hacia MIDI In del segundo dispositivo esclavo y así sucesivamente

hasta un máximo de ocho dispositivos conectados en serie; esta restricción es debido a los problemas de latencia existente en la transmisión de la información a través de los cables, ya que la información MIDI viaja a una velocidad de 31.25 [Kbps]; para evitar problemas serios de latencia se utiliza un máximo de ocho dispositivos conectados en serie en el encadenado Daisy.



**Figura 2.4 Diagrama de Encadenado DAISY**

#### **2.4. Sistema de Numeración Binario y Hexadecimal:**

El protocolo MIDI utiliza lenguaje binario para la transmisión de datos, es decir, que cada impulso eléctrico puede tomar uno de dos valores: 1 ó 0. El lenguaje MIDI usa bytes (palabras digitales) de ocho bits (letras digitales).

El lenguaje binario se basa en la ubicación de los dígitos (1 ó 0) para dar un valor, por lo tanto un byte podrá tomar valores entre 0 y 255 (en base 10), para el lenguaje binario se toma en cuenta dos bits: el bit de la derecha o LEAST SIGNIFICANT BIT (LSB) que puede tomar valores de 2 ó 0 (en lenguaje de base 10) debido a que al ser un lenguaje binario (base 2) entonces sí el bit está desactivado (tiene valor 0), entonces se obtiene un valor decimal de 0, por otro lado si el bit es 1 en decimal se obtiene  $2^0 = 1$  así se va calculando bit por bit hasta que se llega al bit de la izquierda o MOST SIGNIFICANT BIT (MSB) que puede tomar valores de 0 si está desactivado o  $2^7 = 128$ , en el siguiente ejemplo se aprecia cómo se pasa de lenguaje binario (base 2) a lenguaje decimal (base 10).

$$\begin{aligned} \text{Ej.:} \quad 1001\ 0110 &= 2^7 + 0^6 + 0^5 + 2^4 + 0^3 + 2^2 + 2^1 + 0^0 \\ &= 128 + 0 + 0 + 16 + 0 + 4 + 2 + 0 = 150 \end{aligned}$$

Es útil dividir el byte en grupos de cuatro bits a lo que se le llama “*Nibble*”, con un solo *nibble* se pueden representar 16 valores distintos, para lo que es útil usar el lenguaje hexadecimal, es decir, de base 16, en el cual con un solo valor se puede representar un *nibble* y con dos valores en lenguaje hexadecimal se puede representar un byte completo.

Al haber números que se repiten en el lenguaje hexadecimal y decimal es necesario especificar en qué lenguaje se está hablando, tomando el ejemplo anterior se hará la conversión a lenguaje hexadecimal:

$$\text{Ej.:} \quad 0110 = 6 \text{ (H)} = 6 \text{ (D)}$$

Además de los ocho bits utilizados en un byte, el protocolo MIDI utiliza dos bits más por cada byte: un bit al inicio del byte denominado bit de partida cuyo valor es cero y un bit al final del byte denominado bit de parada cuyo valor es uno.

## 2.5. Tipos de Bytes:

La norma MIDI reconoce dos tipos de bytes:

### 2.5.1. Byte de Estado:

Está determinado por el bit más significativo (MSB), este siempre tiene el valor de 1, es decir, de 1000 0000 (128 decimal, 80 hexadecimal) a 1111 1111 (255 decimal, FF hexadecimal).

Su función principal es determinar el tipo de mensaje que será transmitido, por ejemplo: note *on/off*, selección de bancos de sonidos, inicio o parada del secuenciador, *sustain on/off*, etc.

### **2.5.2. Byte de Datos:**

Está determinado por el bit más significativo (MSB), éste siempre tiene el valor de 0, sus valores están comprendidos entre 0000 0000 (0 decimal, 0 hexadecimal) y 0111 1111 (127 decimal, 7F hexadecimal), el byte de datos determina con exactitud la nota, la velocidad, el volumen, el rango en el *pitch bend*, número de programa de memoria, entre otros.

Para un byte de estado pueden existir varios bytes de datos, por ejemplo si el mensaje del byte de estado indica cambio de sonido en cierto canal; se necesita un byte de datos que indique que *patch* de sonido se activará.

Todos estos mensajes tienen 128 valores, por ejemplo el protocolo MIDI reconoce 128 notas (C-1 a G9), de igual manera el volumen tendrá 128 valores, en el caso de la rueda de *pitch bend* y la rueda de modulación que requieren mayor exactitud se usarán dos data bytes.

### **2.5.3. Running Status:**

No es un tipo de byte exactamente, más bien es una herramienta útil que permite ahorrar información y mejora la transmisión de la misma. Cuando se tienen varios mensajes del mismo tipo, no hace falta repetir el Status Byte para todos los mensajes del mismo tipo, solamente se envía el Status Byte del primer mensaje, y posterior a eso los Data Byte de todos los demás mensajes, lo que facilita enormemente la transmisión de datos.

## **2.6. Canales MIDI:**

El protocolo MIDI reconoce 16 canales simultáneos, es decir, en la reproducción de una secuencia se podrían tener 16 instrumentos sonando independientemente al mismo tiempo, esta cantidad de instrumentos podrían disminuir dramáticamente, dependiendo del tipo de sonido que se escoja, por ejemplo los sonidos multitímbricos (sonidos que ocupan más de un timbre o sonido a la vez), ocuparán más de un canal por lo que entre más complejo o

mayor cantidad de capas tengan los sonidos menos canales quedarán disponibles para otros instrumentos.

Como se dijo el protocolo MIDI reconoce 16 canales, debido a que un byte de estado (status byte) el primer *Nibble* (los primeros cuatro bits) se los ocupa para establecer el tipo de mensaje y el segundo *Nibble* se lo utiliza para determinar el número de canal en el que se va a efectuar la acción del mensaje enviado.

**Tabla 2.1. (Numeración de canales MIDI y su transformación a lenguaje binario, decimal y hexadecimal).**

DECIMAL (Interpretación Humana)	DECIMAL (Sistemas Digitales)	HEXADECIMAL	BINARIO
1	0	0 H	0000
2	1	1 H	0001
3	2	2 H	0010
4	3	3 H	0011
5	4	4 H	0100
6	5	5 H	0101
7	6	6 H	0110
8	7	7 H	0111
9	8	8 H	1000
10	9	9 H	1001
11	10	A H	1010
12	11	B H	1011
13	12	C H	1100
14	13	D H	1101
15	14	E H	1110
16	15	F H	1111

La norma MIDI reconoce 16 canales: del cero al quince; sin embargo la interpretación humana de dichos canales es del uno al dieciséis, es por eso que hay diferencia de un canal entre el valor decimal MIDI y la interpretación decimal humana.

La norma MIDI permite la transmisión de datos a 16 canales simultáneos, esta limitación se da debido a que para la asignación de canal MIDI se manejan únicamente cuatro bits, es decir, en todas las combinaciones posibles se obtiene un límite de 16 canales.

### **2.6.1. Mensajes de Canal:**

Los mensajes de canal, son como su nombre lo indica, aquellos que afectan únicamente al canal al que se está asignado el mensaje, es decir, el mensaje afectará única y exclusivamente al instrumento o instrumentos asignados a ese canal.

La estructura del mensaje es la siguiente: se compone de un byte de estado y uno ó dos bytes de datos: en el byte de estado el primer *nibble* corresponde al tipo de mensaje a enviar (prender notas, apagar notas, cambio de programa, presión polifónica de tecla, etc.); el segundo *nibble* corresponde al canal al que va a ser asignado dicho mensaje (de 0000 a 1111 en binario; 0 a F en hexadecimal).

A su vez los mensajes de canal se dividen en dos tipos de mensajes:

1. Mensajes de Voz (*Channel Voice*).
2. Mensajes de Modo (*Channel Mode*).

#### **2.6.1.1. Mensajes de Voz (Channel Voice):**

Este tipo de mensaje, determina la forma de emisión del sonido, entre los principales mensajes de voz son: note off, note on, polyphonic key pressure, control change, program change, channel pressure, pitch bend change, etc.

#### **2.6.1.1.1. Note Off (Apagar la Nota):**

Este mensaje se envía desde el controlador apenas se suelta la tecla, produciendo el cese inmediato del sonido asignado a dicha tecla.

Su byte de estado tiene un valor de 1000 (8H) en el primer *nibble* determinando el mensaje de apagar la nota, el primer byte de datos determina el número de nota a apagarse y el segundo byte de datos determina la velocidad de relajación de la nota, muy pocos instrumentos transmiten un valor variable para este byte, en sentido general se transmite un valor fijo de 0 ó 64 (0000 0000 ó 0100 0000 en binario).

#### **2.6.1.1.2. Note On (Prender Nota):**

Este mensaje se genera cuando se toca una tecla en el controlador. Su byte de estado tiene dos *nibbles*: el primer *nibble* siempre será 1001(9H) ya que este determina que se debe prender la nota, el segundo *nibble* determina el canal en el que se debe prender la nota, el primer byte de datos determina la nota que deberá encenderse y el segundo byte de datos determina la velocidad de ataque de la nota; cuando el valor de la velocidad de ataque es cero, quiere decir que la nota está apagada, eso es muy útil cuando se usa *Running Status*, ya que no hará falta enviar un nuevo Byte de Estado para apagar las notas, simplemente se enviará un valor de 0000 0000 en el byte de datos correspondiente a la velocidad de ataque y la nota se apagará.

#### **2.6.1.1.3. Polyphonic Key Pressure (Presión Polifónica):**

Este mensaje se transmite para variar la fuerza a las teclas ya presionadas, generalmente se relaciona con la modulación o vibrato de los sonidos, es polifónica porque cada tecla puede transmitir su propio mensaje de presión polifónica, aunque cabe aclarar que no todos los timbres son capaces de transmitir o responder a este mensaje.

El primer *nibble* de su byte de estado siempre será 1010 (AH) ya que este determina la presión que se debe aplicar a la nota, el segundo *nibble*

determina el canal al que se transmitirá el mensaje, su primer byte de datos determina el número de la nota a la que aplica el vibrato, y el segundo byte de datos determina la cantidad de presión (vibrato o modulación) que aplica a la nota.

#### **2.6.1.1.4. Control Change (Cambio de Controlador):**

Este tipo de mensajes son transmitidos por los faders, ruedas, potenciómetros, pedales, etc. Con el fin de dar mayor expresividad y control de parámetros al intérprete. Existen dos tipos de controlador:

La norma MIDI establece 128 valores de controladores estando designado el 0000 0001 (1 en decimal, 01 en hexadecimal) exclusivamente para la rueda de modulación, y los valores del 0111 1001 a 0111 1111 (de 121 a 127 en decimal y 79 a 7F en hexadecimal) están reservados para los mensajes de cambio de modo. En la siguiente tabla tenemos los principales valores de cambio de controlador.

El primer *nibble* de su byte de estado siempre será 1011 ya que este determina el cambio de controlador, el segundo *nibble* determina el canal en el que se debe cambiar el controlador, su primer byte de datos determina el número de controlador a modificar, y su segundo byte de datos determina el valor de cambio en el controlador en un rango determinado, sus valores irán de 0000 0000 a 0011 1111 (00 a 3F en hexadecimal y de 0 a 63 en decimal) para los controladores continuos y de 0100 0000 a 0111 1111 (80 a 7F en hexadecimal y de 64 a 127 en decimal) para controladores de switch.

##### **2.6.1.1.4.1. Controladores Continuos:**

Son los controladores que requieren de un rango de valores (0 a 127) como el volumen, rueda de modulación, *faders* y potenciómetros en general. Los controladores continuos pueden admitir uno o dos bytes de datos; en el primer caso, su resolución es de 7 bits (128 valores posibles); en el segundo

caso su resolución será de 14 bits (16384 valores), en este caso se necesitará un byte de datos adicional.

Los números de controlador van de 0000 0000 a 0001 1111 (0 a 31 en decimal, 00 a 1F en hexadecimal), los números de controlador de 0010 0000 a 0011 1111 (32 a 63 en decimal, 20 a 3F en hexadecimal) necesitan una resolución de 14 bits y transmitirán el segundo byte de datos.

#### **2.6.1.1.4.2. Controlador de Switch o Interruptor:**

En esta clasificación de controladores se encuentran los controladores que solamente tienen dos posiciones: encendido y apagado (on y off), en esta clasificación se encuentran los pedales (*sustain*, *soft*, etc.) y los botones. Los números de controlador para este tipo de controlador irán desde 0100 0000 a 0101 1111 (64 a 95 en decimal, 40 a 5F en hexadecimal). Este tipo de controladores solamente admitirán un byte de datos, es decir, su resolución es de 7 bits (128 valores) distribuidos así: los valores de 0000 0000 a 0011 1111 (0 a 63 en decimal, 00 a 3F en hexadecimal) serán reconocidos con la posición de apagado (off) y los valores de 0100 0000 a 0111 1111 (64 a 127 en decimal, 40 a 7F en hexadecimal) serán reconocidos con la posición de encendido (on).

#### **2.6.1.1.5. Program Change (Cambio de Programa):**

Este mensaje se utiliza para seleccionar un timbre, sonido o *patch* determinado para asignarlo a cierto canal. Existen grupos de 128 sonidos llamados “bancos de sonidos”, por lo tanto para ir al sonido “129” se deberá cambiar de banco de sonido y este pasará a ser el sonido o *patch* número 1 del nuevo banco de sonidos seleccionado.

El primer *nibble* de su byte de estado siempre es 1100 (CH) ya que este determina el cambio de programa, el segundo *nibble* determina el canal en el que se debe cambiar el programa y su byte de datos determina el número de sonido o *patch* a seleccionar.

#### **2.6.1.1.6. Channel Pressure (Presión de Teclado Monofónico):**

Se lo denomina también *Aftetouch* y tiene un efecto parecido a *Polyphonic Key Pressure* la diferencia (por lo que se lo llama Monofónico) es que afecta a todas las notas que se estén tocando en ese momento y no a cada una en particular como *Polyphonic Key Pressure*. Su principal utilidad es generar vibrato, por medio de una modulación de la frecuencia, también se lo utiliza para controlar la cantidad de brillo de un sonido o para controlar el volumen en un sintetizador esclavo.

Aunque la norma MIDI reconoce este mensaje como *Channel Pressure* en la práctica se denomina “Aftertouch Monofónico” y al *Polyphonic Key Pressure* “Aftertouch Polifónico”.

El primer *nibble* de su byte de estado siempre es 1101 ya que este determina la modulación de frecuencia ó presión en un canal determinado, el segundo *nibble* determina el canal en el que se debe aplicar la presión de teclado monofónico, su byte de datos determina la cantidad de “vibrato” o modulación que se haga a la frecuencia.

#### **2.6.1.1.7. Pitch Bend Change:**

Este mensaje se genera cuando se mueve la rueda de “Pitch Bend” o cambio de tono, lo que produce un cambio de afinación de las notas que se encuentren activas, como es un control que requiere de un gran nivel de control se usan dos bytes de datos, ya que es necesaria gran precisión y es muy poco apetecible que se perciban escalones en el efecto de cambio de tono que produce la rueda.

En el caso de los bytes de datos no hay diferencia de función entre ellos, salvo que el primero determina el byte más significativo (MSB) de la posición de la rueda; el segundo byte de datos determina el byte menos significativo (LSB) de la posición de la rueda. Los bytes de datos tienen siempre su byte más significativo (MSB) con valor de cero, lo que deja siete bits por cada byte, como

este mensaje tiene dos bytes de datos son utilizados catorce bytes para la representación del valor de la rueda de *pitch bend* siendo determinado de la siguiente manera.

$$\text{Valor} = \text{MSB} \times 128 + \text{LSB} \quad [\text{Ecuación 2.1}]$$

Por ejemplo, cuando los valores de los bytes MSB y LSB son cero, el valor resultante es:

$$0 \times 128 + 0 = 0$$

Cuando los valores de los bytes MSB y LSB son 128, el valor resultante es:

$$128 \times 128 + 128 = 16512$$

Por tanto el rango resultante es desde 0 a 16512. El primer *nibble* de su byte de estado será 1110 (EH) ya que este determina el cambio en la rueda de *pitch bend*, el segundo *nibble* determina el canal en el que se debe cambiar la rueda; su primer byte de datos determina el bit más significativo (MSB) de la posición de la rueda de *pitch bend*; su segundo byte de datos determina el valor del bit menos significativo (LSB) de la posición de la rueda de *pitch bend*.

#### **2.6.1.2. Channel Mode (Mensajes de Modo):**

Este tipo de mensaje, determina el modo de recepción y respuesta de los dispositivos esclavos a los mensajes enviados por los dispositivos maestros en los diferentes canales. Según el fabricante se tiene distintas opciones para la modificación de los mensajes de modo ya sea por botones específicos asignados en el panel frontal o por medio de los mensajes de cambio de controlador para lo cual se utiliza un control remoto maestro.

Los mensajes de modo (Channel Mode) no tienen asignado un byte de estado propio para cada mensaje; se los transmite mediante el formato de los

mensajes de Control Change (Cambio de Controlador), por medio de un número de controlador asignado: desde 0111 1001 a 0111 1111 (de 121 a 127 en decimal y 79 a 7F en hexadecimal).

#### **2.6.1.2.1. Reset All Controllers:**

Este mensaje vuelve a su posición por defecto del instrumento al encenderse a todos los controladores, es especialmente útil cuando el usuario ha cambiado la configuración de uno o varios de los controladores sin estar consciente que lo hizo, por medio de este mensaje se podrá traer al controlador a su estado original.

El primer *nibble* de su byte de estado es 1011 ya que este determina el mensaje de cambio de controlador, el segundo *nibble* determina el canal en el que se debe resetear el controlador; su primer byte de datos determina que el mensaje que se está enviando sea el mensaje de reseteo de todos los controladores; y finalmente el segundo byte de datos determina la posición de encendido o apagado del mensaje, es decir, la activación o desactivación del mensaje.

#### **2.6.1.2.2. Local Control:**

Este mensaje activa o desactiva el generador interno de sonidos del teclado controlador, es decir, si Local Control está en posición ON las notas que se toquen desde el teclado controlador sonarán con el sonido seleccionado en el generador interno del controlador, caso contrario si Local Control está en posición OFF, aunque el controlador no emita ningún sonido, esto no implica que no pueda seguir transmitiendo mensajes de control MIDI a otros teclados esclavos, es decir, cuando Local Control está en posición OFF el teclado controlador se convierte únicamente en eso: un teclado controlador inutilizando los sonidos internos del mismo.

Generalmente este mensaje se encuentra en la configuración MIDI de los dispositivos, pero ciertos fabricantes podrían incluir en el panel frontal la

activación o desactivación de este mensaje a través de un botón denominado como “MIDI”.

El primer *nibble* de su byte de estado es 1011 ya que este determina el mensaje de cambio de controlador, el segundo *nibble* determina el canal en el que actuará el controlador, su primer byte de datos determina que el mensaje que se está enviando sea el mensaje *Local Control*, el segundo byte de datos determina la posición de encendido o apagado del mensaje.

#### **2.6.1.2.3. All Notes Off:**

Este mensaje ejecuta la desactivación inmediata de todas las voces que están sonando en ese momento en ese canal específico, este mensaje se utiliza en el caso de cualquier desperfecto técnico, como por ejemplo la desconexión de un cable justo después de haber ejecutado el mensaje NOTE ON sin que haya avanzado a ejecutarse el mensaje NOTE OFF, al ejecutarse este mensaje apaga todas las notas que se quedaron sonando y que no recibieron el mensaje NOTE OFF, ciertos fabricantes ubican un botón directo que dispara este mensaje a todos los canales denominado “*Panic Button*” o “Botón de Pánico”.

El primer *nibble* de su byte de estado siempre es 1011 ya que este determina el mensaje de cambio de controlador, el segundo *nibble* determina el canal en el que se apagarán las notas, aunque en el caso del botón de pánico el mensaje se envía a todos los canales; su primer byte de datos determina que el mensaje que se está enviando sea el mensaje *All Notes Off*, el segundo byte de datos siempre tendrá valor de cero (0).

#### **2.6.1.2.4. Omni Mode Off:**

Este mensaje de control apaga el “Modo Omni”, modo en el cual el generador de sonidos responde a los mensajes transmitidos por cualquiera de los canales MIDI, este mensaje desactiva esa posibilidad y hace que el generador de sonidos responda únicamente al canal MIDI que asigna el

mensaje de control, además de esto automáticamente envía un mensaje de “All Notes Off” apagando todas las notas que se estén reproduciendo en ese momento.

El primer *nibble* de su byte de estado es 1011 ya que este determina el mensaje de cambio de controlador, el segundo *nibble* determina el canal en el que actuará el controlador; el primer byte de datos determina que el mensaje que se está enviando sea el mensaje *Omni Mode Off*; el segundo byte de datos siempre tendrá el valor de cero (0).

#### **2.6.1.2.5. Omni Mode On:**

Este mensaje de control activa el Modo Omni, es decir, el generador de sonidos recibe todos los mensajes independientemente del canal del que se originen, al igual que *Omni Mode Off*, este mensaje envía automáticamente el mensaje de “All Notes Off” apagando todas las notas que se estén reproduciendo en ese momento.

El primer *nibble* de su byte de estado es 1011 ya que este determina el mensaje de cambio de controlador, el segundo *nibble* determina el canal en el que actuará el controlador aunque la recepción de los mensajes se efectuará desde todos los canales sin importar en que canal esté actuando el controlador; su primer byte de datos determina que el mensaje que se está enviando sea el mensaje *Omni Mode On*; el segundo byte de datos siempre tendrá el valor de cero (0).

#### **2.6.1.2.6. Mono Mode On (Poly Mode Off):**

Cuando se enciende el modo monofónico, el generador asigna una sola nota por canal (la que primero es ejecutada), dichos canales son tomados en cuenta a partir del canal en el que está asignado el controlador, a éste canal se lo conoce como “*Base Channel*”. Al igual que los mensajes anteriores, este mensaje envía automáticamente el mensaje de “All Notes Off” apagando automáticamente todas las notas que se estén reproduciendo ese momento.

El número de canales activos está determinado por el byte de datos dos, si existe el caso que este byte tenga el valor de 0000 0000 (00 en hexadecimal, 0 en decimal) el número de canales activos dependerá del número de voces que tenga el receptor.

El primer *nibble* de su byte de estado es 1011 ya que este determina el mensaje de cambio de controlador, el segundo *nibble* determina el canal en el que actuará el controlador; el primer byte de datos determina que el mensaje que se está enviando sea el mensaje *Mono Mode On*; el segundo byte de datos determinará la cantidad de canales activos, en el caso de tener un valor de cero, el número de canales activos serán determinados por el número de voces que tenga el receptor.

#### **2.6.1.2.7. Poly Mode On (Mono Mode Off):**

Desactiva el modo mono y el receptor comienza a aceptar datos de manera polifónica, es decir, se admitirá más de una nota por canal, como en los anteriores mensajes de modo, con el mensaje "*Poly Mode On*", se transmitirá automáticamente el mensaje de "*All Notes Off*".

El primer *nibble* de su byte de estado es 1011 ya que este determina el mensaje de cambio de controlador, el segundo *nibble* determina el canal en el que actuará el controlador; el primer byte de datos determina que el mensaje que se está enviando sea el mensaje *Poly Mode On*; su segundo byte de datos siempre tendrá un valor de cero (0).

Las combinaciones entre los modos OMNI ON/OFF, MONO y POLY dan como resultado cuatro modos de recepción:

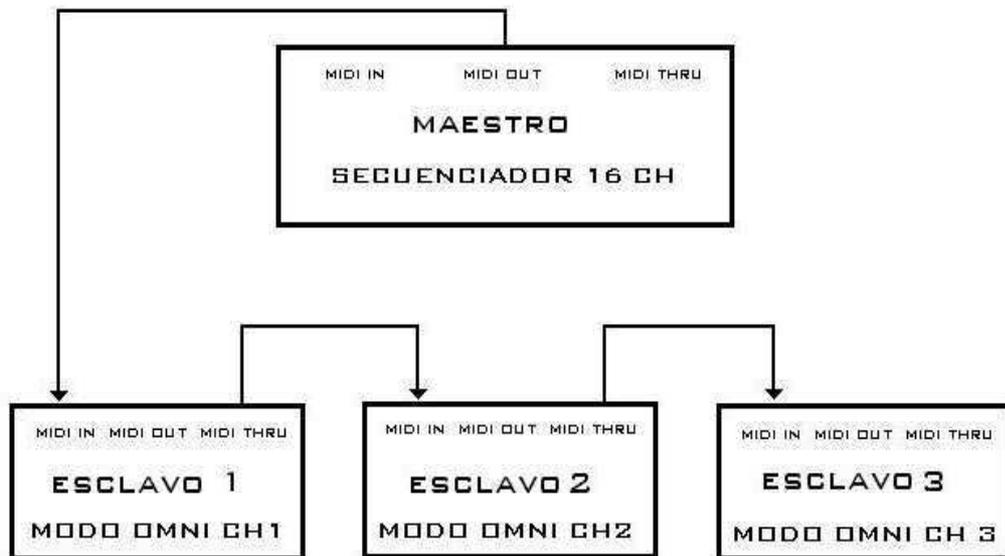
- 1. Modo Omni On/Mono:** Este modo de recepción no tiene utilidad práctica puesto que en este modo de recepción el esclavo recibe la información de los mensajes de control enviado por el maestro, pero la

reproducción la hace nota por nota según el orden en el que son transmitidas, siendo un modo de recepción muy poco práctico.

2. **Modo Omni On/Poly:** Este modo de reproducción toma la información de mensajes de control MIDI desde cualquiera de los canales y la reproduce de manera polifónica, es especialmente útil, por ejemplo, cuando se conectan dos teclados (un máster y un esclavo) y se quiere reproducir la interpretación del teclado máster con las voces de ambos teclados.
3. **Modo Omni Off/Mono:** En este modo de recepción se recibe la información de mensajes de control de un número limitado de canales, determinado por el byte de datos dos y la reproducción se hace de manera monofónica, es decir, se reproduce una nota a la vez por cada canal. Este modo de recepción es útil por ejemplo al tener como controlador a una “Guitarra MIDI”, desde la cual se asigna un canal para cada cuerda (como se sabe las guitarras solamente pueden reproducir una nota a la vez en cada cuerda), al estar en modo Mono, recepta solamente una nota por cada cuerda.
4. **Modo Omni Off/Poly:** Este modo de recepción es el más utilizado, e incluso es denominado de manera general como “Modo de recepción Omni”, en este modo de recepción el dispositivo esclavo recibirá la información únicamente del canal al que esté asignado (base channel) y su respuesta se polifónica. Existen varias aplicaciones para el modo de recepción Omni dos de ellas son:

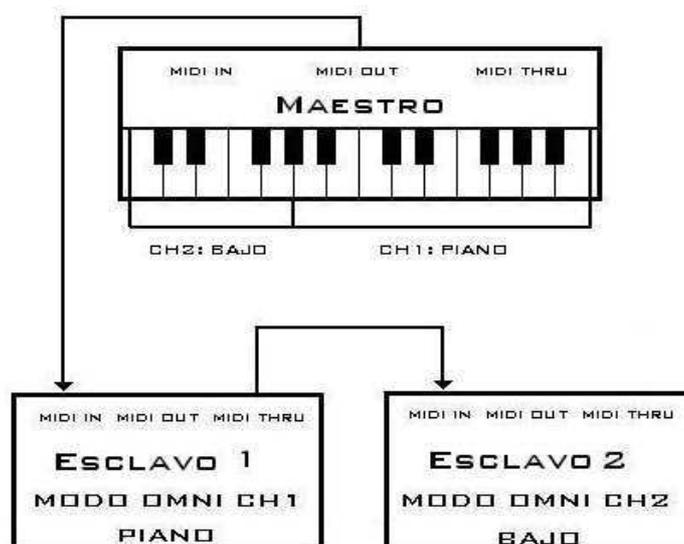
Cuando un secuenciador (dispositivo máster) está conectado a varios teclados o módulos de sonido (dispositivos esclavos), el secuenciador transmitirá la información desde su 16 canales y la enviará a cada uno de los dispositivos esclavos, pero estos recibirán la información únicamente del

canal al que estén asignados (Base Channel) sin importar lo que los otros canales estén transmitiendo recibirán únicamente la información de su canal.



**Figura 2.5 Modo de Recepción OMNI.**

Otro ejemplo de recepción de este canal es cuando se tiene el modo de división de teclado en el teclado máster (Split Mode), y cada parte del teclado se asigna a un dispositivo esclavo, recibiendo estos únicamente la información proveniente del canal la que han sido asignados (Base Channel) y respondiendo de manera polifónica.



**Figura 2.6 Ejemplo de Modo de Recepción OMNI.**

Estos cuatro modos de recepción fruto de la combinación de los mensajes de modo OMNI ON/OFF, MONO y POLY; fueron especialmente necesarios y muy utilizados en los principios del protocolo MIDI cuando la tecnología tanto de control como de reproducción (no había mayor capacidad de reproducción polifónica ni multitímbrica) era casi nula.

Hoy por hoy gracias al avance de la tecnología todos los dispositivos trabajan en modo “Multi”, es decir, cada uno de los canales MIDI tendrá su propio “Instrumento Virtual” con su propio timbre, teniendo opción de determinar el canal de recepción de cada uno de estos instrumentos virtuales y su propio sonido o “patch”.

### **2.6.2. Mensajes de Sistema.**

Este tipo de mensaje no necesita ser asignado a un canal en particular, es decir, se aplica a todo el sistema MIDI en general independientemente de cuantos dispositivos estén conectados al mismo, este tipo de mensajes sirven para manejar a los dispositivos esclavos desde un solo dispositivo máster.

Los mensajes de sistema van desde 1111 0000 a 1111 1111 (F0 a FF en hexadecimal, 240 a 255 en decimal), se dividen en tres tipos de mensaje:

- a. Mensajes de Sistema Común (System Common).
- b. Mensajes de Sistema de Tiempo Real (System Real Time).
- c. Mensajes de Sistema Exclusivo (System Exclusive).

#### **2.6.2.1. Mensajes de Sistema Común (System Common):**

Estos tipos de mensajes son utilizados para controlar dispositivos que no sean secuenciadores incorporados como computadores, cajas de ritmo, samplers, etc.

#### **2.6.2.1.1. Song Position Pointer (Puntero de Posición de Canción):**

Este puntero indica gráficamente la posición de la secuencia que se está reproduciendo, este mensaje se lo utiliza junto con los mensajes de tiempo real, también es un registro generalmente de cuarto de tiempo, del tiempo pasado desde el inicio de la reproducción hasta el momento de reproducción actual, también nos ayuda a iniciar la reproducción desde el inicio del tema reseteándose a cero cada vez que hay un comando “Start”, detenerse en “Stop” y nos permite iniciar la reproducción desde cualquier punto de la secuencia y no necesariamente desde el inicio de la misma con el comando “Continue”.

Puede tener varias formas de representación: min:seg (minutos y segundos); bars:beats (compases y tiempos); samples (muestras); feet:frames (pies y cuadros); time code (horas, minutos, segundos y cuadros).

El byte de estado determina el puntero de posición de canción, su primer byte de datos determina el bit más significativo (MSB) de posición de canción; su segundo byte de datos determina el valor del bit menos significativo (LSB) de la posición de canción.

En el protocolo MIDI se pueden tener un máximo de  $2^{14}$ , es decir, 16384 compases.

#### **2.6.2.1.2. Song Select:**

Este mensaje es similar a “Program Change” (Cambio de Programa) con la diferencia que en lugar de cambiar *patches* (timbres, voces o sonidos) se cambian canciones o secuencias, es un mensaje de control sumamente útil en presentaciones en vivo, donde por cuestiones de tiempo es imposible darse el trabajo de cambiar de canción instrumento por instrumento, este mensaje de control nos ayuda a cambiar de una vez todos los canales de la canción seleccionada desde el dispositivo maestro.

El byte de estado determina mensaje de selección de canción; su byte de datos determina el número de canción a ser reproducida a partir de la recepción del comando “Start”.

#### **2.6.2.1.3. Tune Request (Solicitud de Afinación):**

Este mensaje de control solicita que todos los instrumentos del sistema realicen el proceso de afinación de acuerdo a su referencia interna, generalmente se realiza según el A3 (440 Hz) en instrumentos analógicos con sistema de conexión MIDI tales como sintetizadores, etc.

#### **2.6.2.2. Mensajes de Sistema de Tiempo Real (System Real Time):**

Este tipo de mensajes de control, sirven para sincronizar los dispositivos esclavos del sistema con el maestro.

Estos mensajes de control se aplican además, en el funcionamiento y sincronización de los secuenciadores, de modo que uno de ellos funcione como maestro y los demás como esclavos, es decir, el reloj del dispositivo maestro funciona como guía para todo el sistema. En el protocolo MIDI el reloj realizará 96 divisiones por nota negra (unidad de tiempo); aunque en la actualidad se utiliza una resolución mucho mayor, es decir, se utiliza una resolución de 960 *ticks* por unidad de tiempo, que es el estándar hoy por hoy en la industria.

##### **2.6.2.2.1. Timing Clock (Tiempo de Reloj MIDI):**

Controla y sincroniza el funcionamiento de los dispositivos esclavos con respecto al dispositivo maestro, los esclavos dejan de utilizar sus relojes internos y se sincronizan al reloj del dispositivo maestro, este pulso de sincronización es enviado en intervalos iguales, como se dijo anteriormente, a razón de 24 veces por cada unidad de tiempo.

##### **2.6.2.2.2. Start (Inicio de la Reproducción):**

Este mensaje de control da la orden del inicio de reproducción para la secuencia seleccionada desde el principio de la misma.

#### **2.6.2.2.3. Continue:**

Este mensaje de control de gran utilidad sirve para evitar reproducir la secuencia siempre desde el principio, al contrario, este mensaje de control nos da la opción de reproducir la secuencia seleccionada desde la última posición del puntero de canción o en su defecto nos da la opción de seleccionar mediante el puntero de canción el sitio exacto desde donde se requiere que comience la reproducción de la secuencia.

Consta de un solo byte de estado que actúa junto con el mensaje “Song Position Pointer” para continuar la reproducción donde se había quedado.

#### **2.6.2.2.4. Stop:**

Este mensaje de control manda a detener la reproducción en la posición de puntero de canción actual.

#### **2.6.2.2.5. Active Sensing (Sensor Activo MIDI):**

Este es un mensaje que envía el dispositivo maestro a los dispositivos esclavos del sistema para comprobar el estado de conexión entre ellos, si no recibe mensajes de Timing Clock dentro de 320 [ms] automáticamente apagará las notas que se estén reproduciendo en ese momento.

#### **2.6.2.2.6. System Reset:**

Este mensaje de control se envía desde el dispositivo maestro hacia todos los dispositivos esclavos del sistema y les ordena restablecer los valores que tienen por defecto.

Para que este mensaje de control tenga efecto, el modo de recepción OMNI debe estar activado, ya que *System Reset* debe ir simultáneamente a todos los canales del sistema MIDI.

### 2.6.2.3. System Exclusive (Mensaje de Sistema Exclusivo – SYSEX):

Los mensajes de Sistema Exclusivo (SYSEX) han sido creados para enviar información entre dispositivos manufacturados por un mismo fabricante, sean del mismo modelo o no; estos mensajes servirán para enviar información como el cambio de parámetros en la programación de fábrica de los *patches* (sonidos, timbres o voces), transmisión de programas (*patches*, ritmos pre-programados, actualización del sistema, etc.); también permite la transmisión de información entre un secuenciadores e instrumentos relacionada con la variación de timbres durante la reproducción de las secuencias, también son muy útiles para comunicación entre dispositivos MIDI hardware (instrumentos, secuenciadores, cajas de ritmo, etc.) y software especializado en gestión de *patches* (*samplers*, sintetizadores virtuales, etc.)

La estructura del mensaje depende directamente del fabricante, aunque deben respetar el formato de la norma MIDI para los mensajes de Sistema Exclusivo:

#### 1. Encabezamiento:

Consta de tres bytes uno de estatus y dos de datos; el byte de estatus determina el inicio de Sistema Exclusivo cuyo valor es 1111 0000 (F0 en hexadecimal, 240 en decimal).

El segundo byte es un byte de datos que determina la identificación del fabricante, dicho número es determinado por la MMA (*MIDI Manufactures Association*) y su homóloga japonesa JMSC (*Japanese MIDI Standar Comitee*).

El tercer byte de datos está a consideración del fabricante, sin embargo, generalmente corresponde al modelo del dispositivo. Generalmente cuando los dispositivos tienen una arquitectura similar el fabricante asigna el mismo código con el fin de ahorrar números para dispositivos distintos ya que únicamente hay 128 códigos disponibles

porque al ser un byte de datos el bit más significativo (MSB) será 0 lo que nos deja únicamente 7 bits restantes, lo que en lenguaje binario son 128 valores posibles:  $2^7 = 128$  desde 0000 0000 (00 en hexadecimal, 0 en digital) hasta 0111 1111 (7F en hexadecimal, 127 en decimal).

## 2. Bytes de Datos:

Esta sección de los mensajes de Sistema Exclusivo está abierta a criterio del fabricante, únicamente con una restricción, todos los bytes de esta sección deberán empezar en 0 (cero) para denotar que se tratan de bytes de datos, no se pueden poner en esta sección ningún byte de Estatus porque esto determinaría el fin de los mensajes de Sistema Exclusivo, esta regla se cumple con la salvedad de los mensajes de Tiempo Real (System Real Time) los cuales sí se pueden intercalar como parte de los bytes de datos de esta sección del mensaje de Sistema Exclusivo.

## 3. Tail:

En esta sección denominada “cola” del mensaje hay únicamente un byte de estatus denominado “Fin de Sistema Exclusivo” (EOX – End of SYSEX) cuyo valor es 1111 0111 (F7 en hexadecimal, 247 en decimal).

Dentro de los números de fabricante dispuestos por la MMA y la JMSC han sido asignados mensajes de tipo universal ampliando la norma MIDI, dentro de este tipo de formato de mensaje hay tres tipos: *Sample Dump Standard* (SDS), *Standard MIDI File* (SMF) y el *MIDI Time Code* (MTC).

### 2.6.2.3.1. Sample Dump Standard:

Este mensaje sirve para transferencia de sonidos muestreados, controla características como punto de inicio y final de un loop dentro de un mismo sample, admite gran cantidad de frecuencias de muestreo, es decir, que no es necesario que el dispositivo transmisor y el dispositivo receptor tengan la

misma frecuencia de muestreo y resolución de cuantización con el fin que dichas muestras de sonido puedan ser implementados en cualquier tipo de *sampler* (hardware o software), sin embargo no todos los modelos son compatibles con este tipo de comunicación MIDI para control de *samples*.

#### **2.6.2.3.2. Standard MIDI File (Archivo MIDI Estándar):**

Es un formato de almacenamiento de información de mensajes de control MIDI de manera ordenada y secuencial (secuencias) y que puedan ser correctamente reproducidas en cualquier tipo de secuenciador (independientemente si es hardware o software), los cuales tienen la capacidad no solamente de leer archivos Standard MIDI File sino también de crearlos.

Existen tres tipos de SMF: los archivos formato 0 contienen un solo *track* que contiene toda la información de la canción, los archivos formato 1 tienen varios *tracks* de almacenamiento, respetando la estructura de secuenciación original de la canción; finalmente los archivos formato 2 tienen varios *tracks* pero esta vez cada *track* contiene una canción por separado, es muy poco frecuente que los secuenciadores admitan archivos formato 2.

#### **2.6.2.3.3. MIDI Time Code (Código de Tiempo MIDI):**

El protocolo MIDI es asíncrono, sin embargo cuando se requiere la interconexión entre dispositivos que utilizan un código de tiempo como grabadores de audio, dispositivos de control de video, entre otros, y dispositivos MIDI (asíncronos), es imperiosa la necesidad de un código que permita la comunicación entre ellos. El MIDI time code, permite este tipo de comunicación entre dispositivos que manejan un código de tiempo y los dispositivos MIDI que son asíncronos.

### **2.7. Microcontroladores**

Son circuitos integrados y programables que tienen en sí mismos unidades de procesamiento, memoria y unidades de entrada y salida.

Su versatilidad y complejidad dependen de la función que va a realizar (mientras más compleja sea la función a realizar y por lo tanto el programa empleado para la misma, requiere de un procesador con mayor grado de complejidad).

El funcionamiento general de los microcontroladores, es el mismo que se usa en las computadoras, es decir, los sistemas microcontrolados, tendrán a breves rasgos, periféricos de entrada (por donde ingresa la información para su procesamiento), memorias (RAM y ROM), una Unidad Central de Proceso (CPU por sus siglas en inglés) y salidas por donde se enviará los resultados del procesamiento, además de un generador de reloj integrado, es decir, lo que un microcontrolador necesita para funcionar es instrucciones de control (programas) y un cristal de sincronización (reloj digital).



**Figura 2.7 Diagrama de un Microcontrolador.**

Los dispositivos de entrada y salida son muy variados como conversores análogo/digital (A/D), temporizadores, etc.

### **2.7.1. Estructura de un Microcontrolador:**

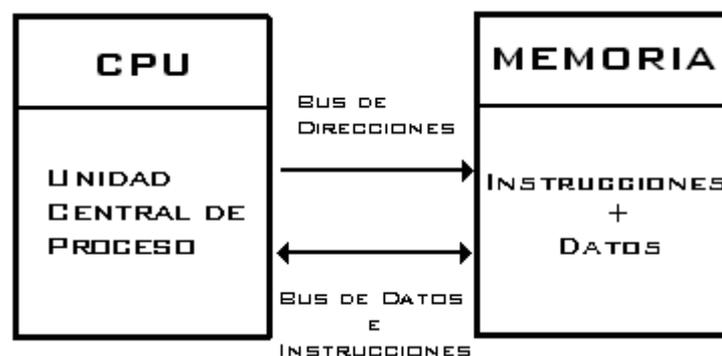
Los microcontroladores son dispositivos computacionales (con funciones y estructuras similares a un computador) pero dentro de un solo circuito integrado. Esta estructura se compone principalmente de un núcleo, donde se localizan el procesador y las memorias; y varios circuitos adicionales, donde se encuentran los dispositivos de entrada y salida.

Debido a su diseño y funcionamiento los microcontroladores utilizan básicamente dos tipos de arquitectura (dependiendo de su función):

#### 2.7.1.1. Arquitectura Von Neumann:

En este tipo de arquitectura el sistema se compone de dos partes: la unidad central de proceso (CPU), y la memoria donde se ubican indistintamente las instrucciones de los programas y los datos, accedidos por un bus de datos, dirección y control. En un computador convencional, cuando se carga un programa se asignan segmentos distintos para las distintas funciones de memoria (código fuente, datos y pila), en este tipo de arquitectura aunque se tiene conciencia de estos segmentos de memoria distintos, se habla de la memoria como un todo.

En los microcontroladores hay dos tipos de memoria: memoria de datos o SRAM y memoria de programas; dentro de los principales tipos de memoria de programa están: ROM, PROM, EEPROM, FLASH, etc. Los microcontroladores no se asignan segmentos distintos para las funciones; al contrario el acceso a los distintos tipos de memoria (programas o datos) está supeditadas a las órdenes dadas por el procesador.

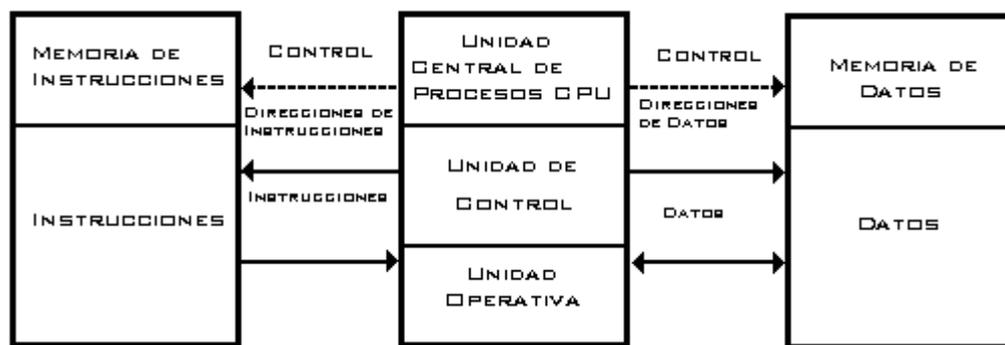


*Figura 2.8 Diagrama Arquitectura Von Neumann.*

#### 2.7.1.2. Arquitectura Harvard:

Este tipo de arquitectura es aparentemente más compleja debido a que tiene dos memorias independientes: la una tiene solamente las instrucciones

para el procesamiento, y la otra memoria tiene solamente los datos a ser procesados teniendo cada memoria un buses de datos, de direcciones y de control independientes, así como también son independientes la lectura/escritura de cada una de las memorias, lo que significa un importante incremento en la velocidad de procesamiento. Además este tipo de arquitectura, permite adecuar el tamaño de los buses en función de cada tipo de memoria, lo que implica un ahorro recursos y un incremento de eficiencia en el microcontrolador.



**Figura 2.9 Diagrama Arquitectura Harvard.**

### 2.7.1.3. Registros:

Los registros tienen dos funciones principalmente: almacenar los resultados de la ejecución de operaciones y comunicarse con la memoria externa (transmisión o recepción de datos desde o hacia la memoria externa).

Aunque el espacio designado para ellos es realmente pequeño dentro de la estructura del microprocesador, dichos registros determinan que cantidad de bits (4, 8, 16, 32 ó 64) usará el microprocesador para el procesamiento de datos, lo que implica una diferencia abismal entre la velocidad, la versatilidad y la eficiencia del dispositivo.

### 2.7.1.4. Unidad de Control:

Las unidades de control son el elemento más complejo de un microcontrolador ya que dependiendo del tipo de procesador agrupan entre

otras, a las unidades de codificación, ejecución, controladores de memoria caché, controladores de buses, controladores de interrupción, etc.

Además de tener estas subunidades dentro de sí, la unidad de control tiene funciones como: el tipo de conjunto de instrucciones, velocidad de ejecución, tiempo del ciclo de máquina, tipo de buses del sistema, manejo de interrupciones, etc.

#### **2.7.1.5. Unidad Aritmético Lógica:**

En la Unidad Aritmético Lógica se realizan además de sumas y restas las operaciones del algebra de Boole, básicas para el procesamiento de datos, además, se adiciona un procesador secundario llamado “coprocesador matemático” que se encarga de las operaciones en coma flotante, y otras operaciones más complejas.

Dependiendo de la potencia del procesador, la Unidad Aritmético Lógica hará los cálculos a mayor o menor velocidad, lo que reflejará una disminución del tiempo de procesamiento y por ende del tiempo de respuesta.

#### **2.7.1.6. Buses:**

Son las vías usadas para la transmisión y comunicación de datos entre los distintos componentes del microcontrolador, por ejemplo, comunicará a la Unidad Central de Procesos (CPU) y la memoria, etc.

En los microcontroladores estos buses son usados generalmente para las entradas y salidas de propósito general y periféricos del sistema. Existen tres tipos de buses:

##### **2.7.1.6.1. Bus de Dirección:**

Este tipo de bus permite la comunicación con un tipo de dispositivo externo con el que se quiere trabajar, también transmitir datos desde o hacia las memorias.

#### **2.7.1.6.2. Datos:**

Es un tipo de bus de dirección, una vez que éste se comunica con una sección determinada, el bus de datos transmite los mismos hacia el sitio establecido para dicho fin.

#### **2.7.1.6.3. Control:**

Como su nombre lo indica, este tipo de bus controla la operación de los dispositivos del sistema, además de gestionar los procesos de lectura y escritura en las memorias.

#### **2.7.1.7. Conjunto de Instrucciones:**

Aunque no es un elemento físico del microcontrolador, el conjunto de instrucciones (*software* o programa) serán las que determinen la función o funciones que son realizadas por el microcontrolador.

Existen dos tipos de repertorio de instrucciones:

##### **2.7.1.7.1. Instrucciones CISC (*Complex Instruction Set Computer* – Computadora de Conjunto Complejo de Instrucciones):**

Con un amplio conjunto de instrucciones, las instrucciones CISC permiten realizar procesos complejos entre operandos situados en la memoria o en los registros internos, tiene como desventaja, la dificultad de la simultaneidad entre instrucciones de distinto tipo.

##### **2.7.1.7.2. Instrucciones RISC (*Reduced Instruction Set Computer* – Computadora de Conjunto Reducido de Instrucciones):**

Los procesadores de este tipo de instrucciones, tienen ciertas características muy convenientes como: instrucciones de tamaño fijo, número reducido de instrucciones, solamente las instrucciones de carga y almacenamiento tienen acceso a la memoria de datos.

Las ventajas de los microcontroladores con este tipo de instrucciones es la simultaneidad de instrucciones de distinto tipo, y una reducción bastante significativa de los tiempos de accesos de memoria lo que significa un aumento de la velocidad de procesamiento.

Actualmente existen microcontroladores tipo CISC que son capaces de convertir instrucciones complejas en varias instrucciones simples tipo RISC lo que los hace muy versátiles en cuanto a su programación.

#### **2.7.1.8. Memoria:**

Los microcontroladores, tienen dos tipos de memoria, en la mayoría de los casos el espacio físico de la memoria dentro de los microcontroladores se encuentra dentro del mismo encapsulado del microcontrolador, ya que lo que se busca, es disminuir al mínimo la circuitería (tanto interna como externa), con el fin de disminuir el tiempo de procesamiento de información y consecuentemente aumentar la velocidad.

Existen dos tipos de memoria distintos:

##### **2.7.1.8.1. Memoria RAM:**

La memoria RAM, también llamada memoria “volátil” tiene básicamente dos funciones: la primera es de almacenamiento de información temporal con el fin de hacer cálculos u operaciones lógicas, la segunda función es la de guardar registros de trabajo y configuración del procesador, y de los periféricos del sistema.

La memoria RAM generalmente utilizada por los microcontroladores es la memoria tipo SRAM que aunque es más costosa que las memorias DRAM (utilizadas por los computadores personales) es más conveniente para el funcionamiento de los microcontroladores por dos razones: la primera es que éste tipo de memoria no necesita de sistema de refrescamiento de datos como

lo hacen las memorias DRAM, este refrescamiento de datos es posible gracias a que los computadores personales poseen gran cantidad de memoria RAM.

La segunda razón por la que las memorias SRAM son mayormente utilizadas para los microcontroladores es que ocupan muy poco espacio y como se dijo antes es muy apetecible tener a la memoria dentro del mismo encapsulado del microcontrolador, la desventaja es que como ocupa poco espacio, también tiene poca capacidad (no excede los 5 [Kb]).

#### **2.7.1.8.2. Memoria ROM:**

Este tipo de memoria llamada también “memoria de programa” es en la que se ubica todo lo relacionado con las instrucciones o programas para el procesamiento de datos que llegan a través de los periféricos cuya configuración está en la memoria RAM.

Dependiendo del tipo de función de los microcontroladores existen distintas tecnologías para el uso de las memorias ROM, las principales son cinco:

##### **2.7.1.8.2.1. Memoria ROM de máscara:**

La producción de este tipo de memorias es muy alto y se usa en aplicaciones exclusivas, ya que este tipo de memoria no “graba” el programa como lo hacen otro tipo de memorias, sino que el microcontrolador es fabricado con el programa, es decir, el programa es incluido dentro de la fabricación del microcontrolador, lo que eleva su costo de producción en un porcentaje considerable, por su alto costo este tipo de memoria se fabrica a gran escala, para aplicaciones específicas de dispositivos específicos.

##### **2.7.1.8.2.2. Memoria PROM ó OTP (*One Time Programmable*):**

Este tipo de memoria se utiliza principalmente en aplicaciones que no requieran actualización de programas, los cuales además deben ser pequeños, o para partes intermedias de un programa mayor en la que cierta cantidad de

datos deban ser guardados como una variable para futuros usos en otras partes del programa.

#### **2.7.1.8.2.3. Memoria EPROM (*Erasable Programmable Read Only Memory*):**

Actualmente este tipo de memorias ha caído en desuso, han sido desplazadas por las memorias EEPROM y las memorias FLASH. Este tipo de memoria se utiliza cuando se requiere actualizaciones de programas. Su forma de escritura es similar a las memorias OTP, es decir, se utiliza cualquier tipo de programador/escritor para elaborar y escribir (grabar) el programa en él, debido a que es reprogramable, antes de volver a escribir un programa en ella, es necesario borrarla, para esto se la debe exponer a una fuente de luz ultravioleta, lo que dificulta mucho su reprogramación. Se las reconoce fácilmente por su forma de construcción: su encapsulado es de cerámica con una cubierta de cristal de cuarzo a través de la cual se puede observar el interior del microcontrolador.

#### **2.7.1.8.2.4. Memoria EEPROM (*Electrical Erasable Programmable Read Only Memory*):**

La principal diferencia entre este tipo de memorias y las EPROM es que el borrado de este tipo de memorias se lo puede hacer eléctricamente, es decir, no hace falta exponerla a la fuente de luz ultravioleta, lo que hace innecesaria en su construcción el encapsulado de cerámica y la cubierta de cristal de cuarzo, esto significa que serán mucho más manejables y mucho más económicas también. Una característica revolucionaria de este tipo de memoria es que no hace falta sacar el microcontrolador de su tarjeta contenedora para hacer actualizaciones al programa del mismo, esto gracias a que este tipo de memoria es compatible con ICSP (In Circuit Serial Programming – Sistema de Programación en Circuito), lo que lo hace aún más accesible al desarrollador medio que no tiene el dinero ni las implementaciones técnicas para trabajar con los otros tipos de memoria.

#### **2.7.1.8.2.5. Memoria Flash:**

Este tipo de memoria es de las más populares hoy en día y está desplazando rápidamente a sus hermanas mayores, debido principalmente a sus prestaciones, la capacidad de memoria (cada vez mayor) y su económico precio. Las memoria Flash, utilizan las mismas tensiones de alimentación que el microcontrolador, además la velocidad de programación y actualización de programas es mucho mayor y más fácil, además el acceso de lectura es considerablemente mayor, lo que significa un incremento significativo en la velocidad de procesamiento.

#### **2.7.1.9. Periféricos:**

La función principal de los periféricos es la comunicación del microcontrolador con “el mundo exterior”, es decir, a través de los periféricos es posible recibir la información para ser procesada de acuerdo a los datos recibidos para dicho fin, y después del proceso se entrega los resultados de dicho procesamiento. Los principales periféricos son:

##### **2.7.1.9.1. Entradas y Salidas de Propósito General:**

En un microcontrolador dado cualquier pin de entrada/salida (E/S) se lo puede utilizar como entrada y salida de propósito general, pero esto debe ser claramente especificado mediante los registros destinados a ellos, es decir, qué pines serán de entrada y qué pines serán de salida.

Generalmente los periféricos de E/S están agrupados en puertos de ocho bits, como se señaló anteriormente, su función principal es recibir datos desde el exterior para escribirlos en la memoria de los microcontroladores para su posterior procesamiento.

##### **2.7.1.9.2. Temporizadores y Contadores:**

Las funciones principales de los temporizadores y de los contadores son: sincronización con otros periféricos que requieran un conteo de tiempo para

sus funciones, implementación de relojes, medición de frecuencia de recepción y transmisión de datos.

Estos circuitos sincrónicos pueden ser utilizados frecuentemente conjuntamente con varios temporizadores y contadores, incluso llegando a formar arreglos de contadores, la fuente de conteo puede ser tanto interna como externa, para el último caso es necesario contar con un pin asociado configurado como entrada (modo contador), caso contrario, es decir, cuando la fuente de conteo es interna, no hace falta dicho pin asociado, a este se le denomina el modo temporizador.

#### **2.7.1.9.3. Conversor Analógico - Digital (A/D):**

Este es uno de los periféricos más útiles y más necesarios ya que, muchas veces a la entrada se tiene señales analógicas, y para su procesamiento es necesario convertirlas al formato digital. Dependiendo de la resolución necesaria para el procesamiento de datos, se puede utilizar conversores A/D externos, ya que la resolución máxima de los conversores internos es de 12 bits, desde 8 bits.

#### **2.7.1.9.4. Puerto Serie:**

Este periférico permite la comunicación entre el microcontrolador con un computador personal (PC) o con otro microcontrolador. Existen dos tipos de puerto serie: puerto serie UART (*Universal Asynchronous Receiver Transmitter*), este tipo de puerto no permite el modo sincrónico de comunicación. El otro tipo de puerto serie es el puerto serie USART (*Universal Synchronous Asynchronous Receiver Transmitter*), este puerto puede transmitir y recibir datos de modo sincrónico o asíncrono (dependiendo de la configuración del dispositivo receptor/transmisor respectivamente).

Este no es el único modo de intercomunicar dispositivos (microcontroladores o computadores), existen otros puertos de comunicación como: SPI (*Serial Peripheral Interface*), I2C (*InterIntegrated Circuit*) ambos

puertos de comunicación sincrónicos con dispositivos como memorias, pantallas LCD, conversores A/D, etc.

Otros puertos de comunicación son los puertos USB (*Universal Serial Bus*), CAN (*Controller Area Network*), Ethernet, etc.

#### **2.7.1.9.5. Comparadores:**

Estos periféricos tienen la función de comparar señales analógicas para dar como resultado valores binarios (niveles lógicos 0 ó 1), esto sirve para discriminar únicamente a las señales útiles para el procesamiento de datos, y no gastar recursos de procesamiento en señales parásitas innecesarias.

#### **2.7.2. Microcontrolador PIC16F877A:**

El PIC 16F877 de Microchip, que posee una memoria de programa de 8192 palabras (8 bits cada una), utiliza una memoria de datos EEPROM de 256 *bytes*, memoria RAM de 368 *bytes* y 33 pines de entrada/salida, los cuales se dividen en cinco puertos: A, B, C, D y E, además posee ocho conversores análogos/digitales distribuidos en las siguientes entradas/salidas: RA0, RA1, RA2, RA3, RA5, RE0, RE1 y RE2; aunque es un micro controlador muy potente su única desventaja es no contar con un oscilador interno, razón por la cual es necesario incorporar un cristal externo de 4 [MHz] y una resistencia *pull-up*  $R_1$  de 4.7 [K $\Omega$ ].

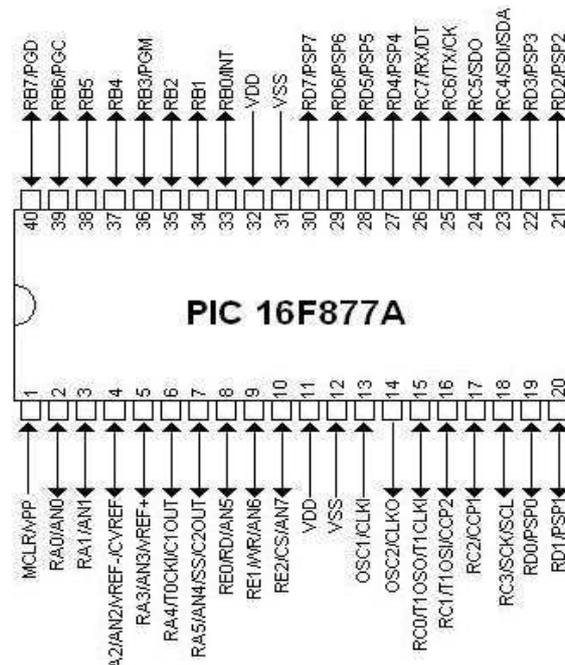


Figura 2.10. Diagrama del PIC 16F877A.

### 2.7.3. Circuito Integrado LM324:

Este circuito integrado contiene cuatro amplificadores operacionales, que funcional con una sola fuente de alimentación para todo el circuito integrado (pines 4 y 11).

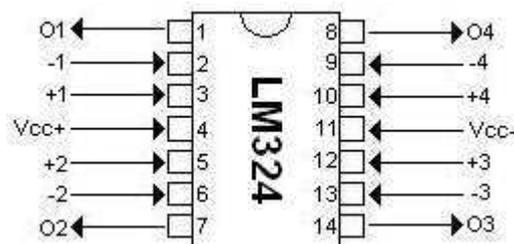


Figura 2.11. Diagrama del Circuito Integrado LM324.

### 2.7.4. Circuito Integrado 40106:

El circuito integrado 40106 es un "Disparador Schmitt", que es una especie de circuito comparador con realimentación, para su funcionamiento este circuito utiliza un ciclo de histéresis: el circuito cambia su voltaje de salida cuando el voltaje de entrada sobrepasa un cierto umbral, con la salvedad que su voltaje de salida no cambia cuando la entrada baja del voltaje del umbral, al contrario, cambia a otro voltaje distinto que será más bajo que el primero, esto

sirve para mantener una relación lineal entre el voltaje de entrada y el de salida evitando que la señal sea distorsionada y que se inserte ruido parásito a la señal original.

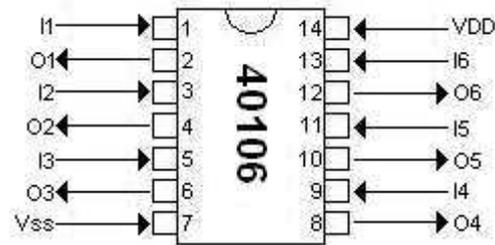


Figura 2.12. Diagrama del Circuito Integrado 40106.

## 2.8. Tecnología de Conversión:

La conversión analógica – digital es el proceso mediante el cual se transduce una señal del lenguaje analógico (amplitud y frecuencia) al lenguaje digital (cuantización y muestreo).

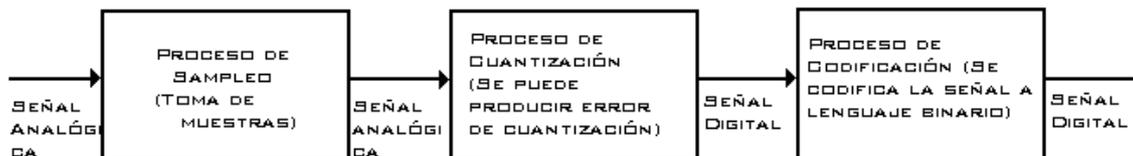


Figura 2.13. Esquema de Conversión Analógico – Digital.

### 2.8.1. Señal Analógica:

La señal analógica viene determinada por un voltaje que caracteriza dicha señal, es decir, es una señal continua de amplitud y frecuencia variable, aunque es muy común considerarla como una señal totalmente fiel a lo grabado, no es así, pues la señal analógica tiene intrínsecamente un ruido interno denominado ruido de fondo, lo que la hace inexacta aunque sea una señal continua, por lo que para un procesamiento y almacenamiento más fiel se hace necesaria la digitalización de la señal, ya que, una señal digitalizada se maneja en el lenguaje binario no es susceptible a distorsión, y el procesamiento que se haga con la misma es mucho más exacto, puesto que lo que se procesa es básicamente ceros y unos.

### **2.8.2. Señal Digital:**

La señal digital es una señal discreta, es decir, la señal toma valores fijos, dependiendo de la frecuencia de muestreo (cuantas veces por segundo se toman datos) y de la tasa de cuantización a la que se está digitalizando la señal.

La mayor ventaja de este tipo de señal es que su nivel de ruido es casi nulo, además si ésta en su transmisión sufriera alguna alteración, mediante los sistemas de regeneración de señales puede ser restauradas casi al completo de su señal original (depende de el nivel de distorsión recibido por la señal original).

El proceso mediante el cual se digitaliza la señal es bastante simple: este consiste en tomar muestras (fotografías) de la señal analógica original en lapsos de tiempo definidos (tasa de muestreo) y registrar los valores de amplitud (tasa de cuantización) estos datos serán registrados en un soporte digital para su posterior procesamiento, compresión, reproducción, etc.

El proceso de conversión analógica – digital consta de dos procesos:

#### **2.8.2.1. Muestreo (Sampling):**

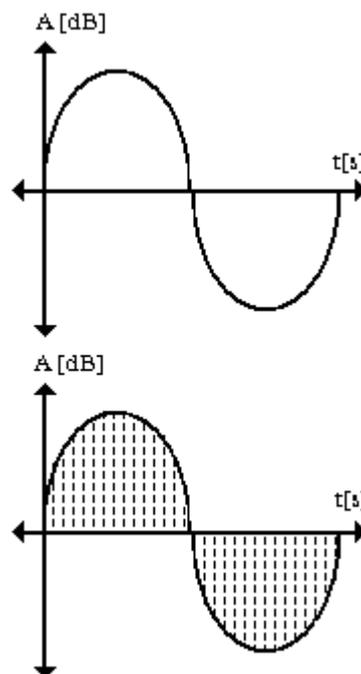
El muestreo consiste en tomar muestras de la amplitud de la señal original cada cierto tiempo determinada por la frecuencia de muestreo (número de muestras que se toma por segundo de una señal analógica).

Las muestras tomadas son retenidas en un circuito (*sample & hold*) mientras es analizada la amplitud de cada una de ellas, para su posterior cuantización y registro, este no es un proceso matemático, al contrario, es un proceso técnico para facilitar su análisis y procesamiento.

Las unidades de la frecuencia de muestreo es los Hertz (Hz), dicha frecuencia determina el ancho de banda de la señal digitalizada, entre mayor

es esta frecuencia la señal original será representada más exacta, pero al mismo tiempo mayor ancho de banda ocupará, lo que se traduce en mayor cantidad de información y mayor espacio de almacenamiento de la señal.

La frecuencia de muestreo está determinada por el teorema de Nyquist que dice que para que una señal después de ser digitalizada pueda ser reconstruida con exactitud y sin errores, la frecuencia de muestreo debe ser mayor al doble de su ancho de banda, por ejemplo, el oído humano puede distinguir sonidos en un rango de frecuencia entre 20 – 20000 [Hz], entonces si se desea digitalizar una señal que tenga ancho completo de banda para el oído humano, la señal deberá ser digitalizada a más de 40000 [Hz] (44100 [Hz] frecuencia de muestreo en los discos compactos).



**Figura 2.14. Ejemplo del Muestreo de una Onda Sinusoidal.**

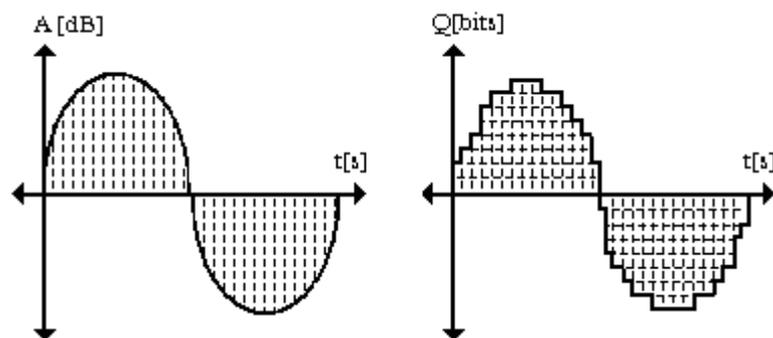
#### **2.8.2.2. Cuantización:**

La cuantización es el proceso mediante el cual se mide la amplitud de la onda analógica en un punto determinado y se lo representa con un valor entero, la distancia entre un valor y otro viene dada por el radio de

cuantización, entre mayor sea este, con mayor precisión será tomada la muestra en el rango de la amplitud, aunque al igual que en caso del muestreo, entre mayor sea este radio, mayor espacio de almacenamiento ocuparán los datos digitalizados.

Cuando la amplitud de una muestra no llega a un punto de cuantización definido se produce el denominado error de cuantización, este error es debido a que entre los puntos que se toman las muestras, se pierde información la que es promediada y esta imprecisión es el error de cuantización que se puede aminorar incrementando la tasa de cuantización y la frecuencia de muestreo.

Producto del proceso de cuantización se obtiene la señal digital, es decir, la información de la señal se dará en ceros y unos, para su registro o procesamiento.



**Figura 2.15. Ejemplo de Cuantización de una Señal Sinusoidal.**

## 2.9. Transmisión de Información

La transmisión de información se da entre dos o más dispositivos de iguales características, existen distintas vías de transmisión, y dependiendo de la tecnología usada puede ser de manera análoga o digital.

### 2.9.1. Transmisión Análoga:

En la transmisión de información análoga, los datos transmitidos son variables debido al cambio continuo de la amplitud de la señal, ya que

únicamente se posee el valor de la corriente y la presencia o no de la misma, un ejemplo de esto es la telefonía análoga, el telégrafo, la televisión análoga, etc.

### **2.9.2. Transmisión Digital:**

La transmisión digital se basa en la transmisión de datos mediante cualquier medio físico en forma de señales digitales, esta información no puede ser enviada como ceros y unos, es necesario, que sea codificada como una señal de dos estados: diferencias de potencial (entre un cable y tierra ó entre dos cables), presencia o ausencia de luz, etc.

Esta codificación se realiza mediante varios sistemas: la codificación en dos niveles es aquella en la que la información a transmitirse solamente puede tomar un valor positivo o un valor negativo.

La codificación de tres niveles es en la que la señal además de tomar valores negativos o positivos también puede tomar un valor nulo (0).

### **2.9.3. Modos de Transmisión:**

Existen tres modos principales de transmisión, estos modos están determinados por la dirección de los intercambios, la cantidad de información enviada a la vez (número de bits) y la capacidad de sincronismo entre el transmisor y el receptor.

#### **2.9.3.1. Modo de Transmisión según la Dirección de Intercambios:**

Existen tres modos de transmisión según la dirección de intercambios: conexión simple, conexión semidúplex, conexión dúplex total(o full-duplex).

##### **2.9.3.1.1. Conexión Simple:**

En este tipo de conexión los datos fluyen en una sola dirección, se la usa para ahorrar circuitería y recursos, como por ejemplo entre la comunicación del computador con la impresora, etc.



**Figura 2.16. Esquema del Flujo de Información de una Conexión Simple.**

#### 2.9.3.1.1. Conexión Semidúplex:

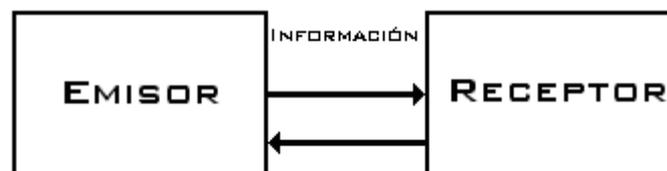
Utiliza todo el ancho de banda, este tipo de conexión es bidireccional, pero con la característica que la información solamente puede fluir de un solo lado a la vez y no los dos al mismo tiempo, es decir, primero tendrán que transmitirse los datos en una dirección y después en otra.



**Figura 2.17. Esquema del Flujo de Información de una Conexión Semidúplex.**

#### 2.9.3.1.1.2. Conexión Dúplex Total:

En este tipo de conexión el ancho de banda se divide en dos, ya que, dicha conexión es totalmente bidireccional, es decir, que al mismo tiempo ambos extremos de la conexión pueden enviar y recibir datos, pero con la consecuencia de perder la mitad de la banda en ambos sentidos.



**Figura 2.18. Esquema del Flujo de Información de una Conexión Dúplex Total.**

### 2.9.3.2. Modo de Transmisión en Serie y Paralelo:

Este modo de transmisión tiene que ver con la cantidad de información que se transmite a la vez, ésta está medida por el número de bits por segundo que se puede transmitir, aunque cabe aclarar que los procesadores realizan sus procesos con una cuota mínima de ocho bits (un byte).

#### 2.9.3.2.1. Conexión en Paralelo:

La conexión en paralelo se realiza cuando se transmiten varios bits por diferentes canales, en el caso de tener varias líneas físicas, se transmite un bit por cada línea, en el caso de tener una sola línea física subdividida en varios canales se transmitirá un bit por cada canal, este caso es especialmente peligroso porque se puede dar interferencias y degradación de la señal; el medio físico por el que se transmite la información puede ser un cable, un alambre o cualquier tipo de conductor soldado en una placa.



*Figura 2.19. Esquema de Conexión en Paralelo.*

#### 2.9.3.2.2. Conexión en Serie:

En este tipo de conexión se transmite un bit a la vez, ya que hay solamente una vía física de transmisión, ya que la mayoría de los procesadores procesan los datos en paralelo, es necesario hacer transformaciones de los datos de serie a paralelo en el receptor y de paralelo a serie en el transmisor. Este proceso se hace de la siguiente manera:

Cuando se quiere realizar la transformación de paralelo en serie (para transmisión de datos), el registro de desplazamiento junto con el reloj desplazará los registros hacia la izquierda y transmitirá el bit más significativo (MSB).

Si el caso es contrario, es decir, se requiere hacer la transformación de serie en paralelo (para recepción de datos) el registro de desplazamiento desplaza el registro a la izquierda cada vez que transmite un bit, para posteriormente transmitir el registro entero en paralelo cuando está completo, esto debido a que como en un inicio se transmitió el bit más significativo en primer lugar, el sistema automáticamente irá ordenando los bits hasta conformar bytes enteros de información.



**Figura 2.20. Esquema de Conexión en Serie.**

### **2.9.3.3. Modo de Transmisión Sincrónica y Asincrónica:**

Debido a la necesidad de reordenar los bytes transmitidos en la conexión en serie, existe un serio problema de sincronización entre los dispositivos, de acuerdo a cada aplicación existen dos tipos de conexión para solucionar dicho conflicto:

#### **2.9.3.3.1. Conexión Asincrónica:**

La conexión asincrónica se da cuando los bits se transmiten sin que exista relación temporal entre el transmisor y el receptor, para lo cual se envía con cada byte información acerca de su posición. Esta información de posición de los bytes se denomina bit de partida y bit de parada respectivamente, aunque puede haber para un solo byte varios bits de parada.

#### **2.9.3.3.2. Conexión Sincrónica:**

En este tipo de conexión los datos se transmiten de manera constante ya que, los dispositivos están sincronizados con un solo reloj, en este tipo de transmisión no existe ninguna separación entre bits, por lo que se hace absolutamente necesario elementos de sincronización entre bits, este modo de conexión es bastante eficiente, excepto cuando los relojes de los dispositivos

no están correctamente sincronizados, es por esto que en este tipo de conexión se debe cumplir con dos principios: la velocidad de conexión nunca debe ser alta ya que entre mayor velocidad de transmisión de datos se tenga, hay mayor posibilidad de retraso o adelanto en uno de los relojes. El segundo principio que se debe respetar, es que la conexión debe permanecer el tiempo suficiente como para que el receptor pueda distinguir la información que le está llegando.

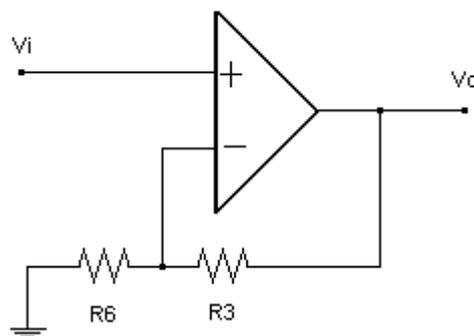
### 3. DISEÑO.

#### 3.1. Alimentación:

La alimentación de energía está dada mediante una fuente de corriente continua (DC) de 9 [V] a 12 [V]; en la entrada de corriente del circuito hallamos un regulador de voltaje 78L05 que permite el paso máximo de 5 [V] hacia el circuito, para la alimentación de los componentes del mismo, también se tiene un arreglo de dos capacitores para de 100 [nF] con el fin de proveer alta frecuencia para proteger al circuito de posible ruido generado por la línea de generación de corriente eléctrica.

#### 3.2. Descripción del Circuito:

El circuito está diseñado para que a la entrada de la señal de audio de nivel de línea (a través de un conector Jack TRS), vaya a un amplificador operacional LM324 configurado como un amplificador no inversor.



**Figura 3.1. Esquema de Amplificador Operacional No Inversor.**

La señal que viene del conector Jack TRS, entra a la entrada no inversora del amplificador operacional, la ganancia del amplificador viene dada por las resistencias  $R_3$  de 1 [M $\Omega$ ] y  $R_6$  de 1 [k $\Omega$ ]; la resistencia  $R_3$  está conectada a la salida del amplificador operacional por un lado y por otro está conectada a la entrada inversora, la misma que a la vez está conectada a la resistencia  $R_6$ , dando como resultado una ganancia de 101 calculada de la siguiente manera:

$$V_o = V_i \left( 1 + \frac{R_3}{R_6} \right)$$

$$V_o = V_i \left( 1 + \frac{1 \times 10^6}{1 \times 10^3} \right)$$

$$V_o = V_i(1 + 100)$$

$$V_o = V_i(101) \text{ [Ecuación 3.1]}$$

La salida del amplificador no inversor alimenta a un circuito “Disparador Schmitt” provisto por un circuito integrado 40106. Para su funcionamiento utiliza desde su salida hasta su entrada una resistencia de  $R_9$  de 1 [M $\Omega$ ], y de su entrada hasta tierra un capacitor de  $C_3$  de 5.7 [pF]; el propósito de este integrado es transformar la señal de entrada en pulsos, la salida del circuito integrado 40106 está conectada al pin 40 (puerto B7) del PIC16F877A donde los pulsos procedentes del circuito integrado 40106 son contados para determinar así la frecuencia de la señal entrante.

La señal que sale de este amplificador operacional no inversor (LM324), también alimenta un circuito rectificador, comprendido por un diodo 1N4148, una resistencia  $R_{11}$  de 1 [M $\Omega$ ] y un capacitor  $C_7$  de 100 [nF], el objetivo de este circuito rectificador es alimentar el pin 39 (puerto B6) del PIC 16F877 para determinar la activación de los mensajes de nota encendida y nota apagada (note on y note off); el circuito rectificador también alimenta al pin 2 (puerto A1) donde la señal es digitalizada para determinar la intensidad (velocidad) de la nota MIDI.

El control de los canales MIDI se los realiza a través de un *Dip Switch* de cuatro puertos los cuales se conectarán por un lado al voltaje  $V_{cc}$  a través de cuatro resistencias:  $R_{13}$ ,  $R_{14}$ ,  $R_{15}$  y  $R_{16}$  de 10 [K $\Omega$ ] cada una, y por el otro lado a los puertos del PIC 16F877A: C0 (pin 15), C1 (pin 16), C2 (pin 17) y C3 (pin 18) respectivamente, desde este *switch* usando lenguaje binario, se determina el canal al que irá transmitida la información, adicionalmente el dispositivo cuenta



### 3.3.1. Subrutina CANALASIG:

Esta subrutina condiciona el estado (1 o 0 lógico) de los puertos C0, C1, C2 y C3; guardando el resultado obtenido en la variable CANAL, estos unos y ceros lógicos están determinados por los switches del DIP SWITCH que al activar o desactivar sus puertos, envía la señal de uno o cero a los distintos puertos, lo que da como resultado el canal al que será enviada la información.

El protocolo MIDI trabaja con 16 canales (del 0 al 15 en lenguaje informático), el DIP SWITCH que controla los canales está configurado para trabajar con información binaria (los números se forman con ceros y unos) para lo que se implementó una pantalla LCD (Pantalla de Cristal Líquido por sus siglas en inglés) donde se mostrará el canal MIDI en notación decimal para el usuario, para lo que se definió la variable CANDEC donde se suma la información de la variable CANAL más uno, dando así el número de canal MIDI para el usuario (canales del 1 al 16); el resultado de la variable CANDEC es el que se muestra en la pantalla LCD.

La información de canal es el segundo *nibble* del byte de estado, el primer *nibble* se lo obtiene definiendo las variables NOTEOFF y NOTEON que tienen valores decimales de 128 y 124 respectivamente, como se necesita el byte de estado entero (formado en su primer *nibble* por el mensaje de *note on* o *note off* y su segundo *nibble* por la información de canal), se definen dos variables más: ESTATUS1 y ESTATUS2 en esta variable se hace la suma de las variables NOTEON + CANAL y NOTEOFF + CANAL respectivamente. El byte de estado final se lo obtiene haciendo una comparación de la información recibida en el puerto B6, proveniente del circuito rectificador, si se tiene un uno lógico, la variable NOTE se cargará con el valor de la variable ESTATUS1 (correspondiente al mensaje de *note on* más el canal), por otro lado si se obtiene un cero lógico en el puerto B6, la variable NOTE se cargará con el valor de la variable ESTATUS2 (correspondiente al mensaje de *note off* más el canal).

Posterior a esto se tiene la condición que si la variable NOTE se carga con el valor de la variable ESTATUS1 (*note on*) se debe encender el led conectado al puerto A2; si la variable NOTE se carga con el valor de la variable ESTATUS2 (*note off*), el led debe apagarse.

### **3.3.2. Subrutina NUMERODENOTA:**

Esta subrutina determina el primer byte de datos, es decir, el número de nota MIDI que deberá dispararse. La señal convertida en pulsos que viene del disparador Schmitt alimenta al puerto B 7, donde el programa cuenta el número de pulsos en un tiempo de 250 [ms], el resultado de este conteo se guarda en la variable FRECUENCIA, para posteriormente hacer una comparación entre la frecuencia obtenida y el número de nota MIDI a disparar, el número de nota se guarda en la variable DATOS1, el dispositivo reconoce frecuencias desde los 27 [Hz] hasta los 12544 [Hz], es decir, desde la nota MIDI 21 hasta la 127 respectivamente.

La limitación existente con este método de detección del tono es que el conteo solamente admite valores enteros, lo que podría ocasionar errores de medio tono en la interpretación de la frecuencia, ya que en la programación fue necesario redondear las frecuencias de las notas para su correcta detección y posterior comparación con el dato obtenido por el contador y guardado en la variable FRECUENCIA.

### **3.3.3. Subrutina VELOCITY:**

La subrutina VELOCITY determina la intensidad (velocidad MIDI) de las notas, constituyendo la velocidad el segundo byte de datos. La señal proveniente del circuito rectificador alimenta al puerto A1, cuando entra la señal es digitalizada y además se mide su amplitud, este dato es guardado en la variable CALCULO, como el protocolo MIDI admite 127 niveles de velocidad, y el voltaje máximo a obtener será de 5 [V], en la variable INTENSIDAD se toma el valor de la variable CALCULO, se la multiplica por 5 (el nivel de voltaje

máximo) y se la divide para 127 (el número de niveles de velocidad), para obtener así el dato necesario para el segundo byte de datos.

#### **3.3.4. Programa Principal INICIO.**

Este programa llama a cada una de las subrutinas para ser ejecutadas, después que cada rutina es ejecutada, se envía de manera serial a través del puerto C 6, a una velocidad de 31.25 [Kbps]; el dato que contiene la variable que almacena el resultado final de cada subrutina: para el byte de estado, la variable NOTE; para el byte de datos uno, la variable DATOS1 y para el byte de datos dos, la variable VELOCIDAD.

## **4. DESARROLLO EXPERIMENTAL.**

### **4.1. Diseño del Dispositivo:**

La construcción del dispositivo fue realizada en tres etapas básicas: etapa de alimentación, cuya base es el regulador de voltaje 78L05 de 5 [V]. La segunda etapa es el circuito de entrada de señal de línea que alimenta a un amplificador LM324, que sube el nivel de la señal, que pasa a un circuito integrado 40106 que es un “Disparador Schmitt”, el cual convierte la señal de nivel de línea (previamente amplificada) en una señal de pulsos que es posteriormente contabilizados para la determinación de la frecuencia; el amplificador LM324 también alimenta a un circuito rectificador compuesto por un diodo 1N4148, una resistencia  $R_9$  de 1 [M $\Omega$ ] y un capacitor de  $C_3$  de 5.7 [pF]. Este circuito tiene el fin de rectificar la señal, para una correcta detección de la misma, para el envío de mensaje de nota encendida y nota apagada; de este circuito rectificador sale también la alimentación para la detección de la velocidad MIDI. Un DIP SWITCH de cuatro puertos sirve para determinar el número de canal MIDI por el que se transmite la información, mediante comparaciones sucesivas de sus puertos, con la información ingresada en el PIC16F877A (ver apartado 3.2.).

La tercera etapa está compuesta principalmente por un microcontrolador PIC 16F877 en donde se realiza el procesamiento de la señal, es decir, la señal de pulsos procedente del circuito integrado 40106 entra al puerto B7 donde mediante un contador se realiza la contabilización de los pulsos para obtener la frecuencia de la señal entrante y mediante sucesivas comparaciones determinar el número de nota MIDI que debe dispararse, que determina el byte de datos 1 (ver apartado 3.3.2.).

El circuito rectificador alimenta a los puertos B6 y A1; en el puerto B6 se determina si está ingresando voltaje para enviar el mensaje de nota encendida y nota apagada (*note on* y *note off* respectivamente) que junto con la información del número de canal (determinada por el DIP SWITCH y la

información obtenida de los puertos C0, C1, C2 y C3) forma el byte de estado del mensaje (ver apartado 3.3.1.). El puerto A1 digitaliza la señal que está alimentada desde el circuito rectificador para medir el voltaje y mediante comparaciones determinar la velocidad de la nota MIDI, obteniendo así el byte de datos 2 (ver apartado 3.3.3.).

Finalmente los datos son enviados de manera serial por el puerto de transmisión serial de datos C6 para llegar al dispositivo MIDI de destino a través del pin número 5 de un conector DIN cinco pines.

#### 4.2. Pruebas:

Las pruebas fueron realizadas tomando muestras de los bancos de sonido del conocido *software* de producción musical “Reason 4.0”. Las muestras orquestales se las obtuvo del banco de sonidos “Orkester” y las muestras de instrumentos electro-acústicos se las obtuvo del banco de sonidos “Factory” de dicho *software*, las muestras de piano se las obtuvo del banco de sonidos “Steinway Refill” de 24 bits de cuantización, para la distorsión de la guitarra eléctrica con distorsión fue utilizado el efecto “Scream 4 Distortion” efecto Fuzz y finalmente las muestras de voz humana fueron grabadas con un micrófono de condensador “Shure SM81”.

Los instrumentos con los que fueron realizadas las pruebas del dispositivo, se determinaron tomando en cuenta varios aspectos:

- **Altura:** Los instrumentos extremos en cada familia orquestal: violín y contrabajo (cuerdas frotadas); pícolo, flauta y fagot (maderas); trompeta y tuba (metales).
- **Timbre:** Algunos instrumentos aunque no están en los extremos de los rangos de frecuencia de la orquesta se tomaron en cuenta debido a la particularidad de su timbre, tal es el caso del chelo (cuerdas frotadas) y el corno francés (metales), además se tiene el caso especial de la

trompeta debido a que la amplitud de sus armónicos es mayor que la de la frecuencia fundamental del mismo.

- **Generación de Sonido:** También fueron utilizados instrumentos electroacústicos propios de una banda de pop-rock tales como bajo, guitarra eléctrica, guitarra eléctrica con distorsión (fuzz), guitarra acústica (instrumentos de cuerda pulsada). Especial mención merece el piano ya que por la generación de sonido (cuerda percutida) y por ser un instrumento melódico-armónico se le dio un tratamiento distinto en las pruebas, ya que, la respuesta del dispositivo ante los acordes (tres o más sonidos tocados simultáneamente) fue efectuada con dicho instrumento.

Las pruebas fueron realizadas a través del *software* de edición musical “Finale 2008”, dicho *software* generó archivos MIDI para cada instrumento, los mismos que fueron abiertos en el *software* “Reason 4.0” obteniendo así las muestras de sonido de las librerías ya citadas, en archivos .WAV a una frecuencia de muestreo de 41000 [Hz] y a una tasa de cuantización de 16 [bits].

Posteriormente estas muestras fueron reproducidas a través del *software* “QuickTime Player” donde se ruteó la señal hacia el dispositivo usando la interfaz “Mbox 2 Pro” a través de las salidas analógicas de dicha interfaz.

De la salida MIDI Out del dispositivo se ingresan mensajes de control MIDI al computador vía USB a través de la interfaz MIDI-USB M-Audio MIDIsport UNO, para grabar los mensajes de control provenientes del dispositivo en el secuenciador MIDI que posee el *software* de producción musical Reason 4.0, para el posterior análisis de los mensajes de control grabados, y determinar qué porcentaje de efectividad se obtuvo del dispositivo.

Se han dividido las pruebas en tres tipos: rítmica, de altura tonal y de dinámica (velocidad) de la siguiente manera:

#### 4.2.1. Prueba Rítmica:

La velocidad elegida para las pruebas es 60 BPM (golpes por minuto), es decir, existirán 60 negras por minuto, a partir de ahí se escribió la escala de C mayor en dos octavas, en figuración de negra, para posteriormente escribir la escala cromática (12 notas por octava) en dos octavas y finaliza la prueba con arpeggios en semicorcheas, para terminar en una redonda (4 tiempos).

Por consiguiente, la prueba rítmica es realizada en tres etapas: la primera en notas negras (una pulsación por segundo); la segunda en corcheas (dos pulsaciones por segundo) y la tercera en semicorcheas (cuatro pulsaciones por segundo) en un total de 17 compases (la prueba total por cada instrumento dura 68 segundos).

##### 4.2.1.1. Presentación de Resultados de la Prueba Rítmica:

**Tabla 4.1. Resultados de la Prueba Rítmica.**

<u>Instrumento</u>	<u>Prueba Rítmica [%]</u>
Pícolo	72.31
Flauta	70.00
Fagot	70.31
Corno Francés	67.19
Trompeta	54.00
Tuba	59.87
Piano	90.00
Violín	68.00

Chelo	65.63
Contrabajo	70.31
Guitarra Acústica	71.83
Guitarra Eléctrica	70.23
Guitarra Eléctrica-Distorsión	73.60
Bajo Eléctrico	70.31
Soprano (voz)	87.60
Contralto (voz)	85.00
Tenor (voz)	83.60
Bajo (voz)	81.26
<b>Promedio</b>	<b>72.84</b>

#### **4.2.1.2. Análisis de Resultados de la Prueba Rítmica:**

Los resultados arrojados de la prueba rítmica muestran que el dispositivo tuvo un resultado positivo el 72.84% de los casos, se observa que la mejor respuesta es de los timbres vocales, siendo el timbre de soprano el que mejor resultado rítmico tuvo con un 87.60%; por otro lado el timbre que peor respuesta rítmica tiene es la tuba con un 59.87%. Estos resultados son parciales ya el dispositivo transmite al receptor un mensaje de control por segundo, por tanto para la prueba rítmica se toma en cuenta la primera nota de cada tiempo del compás (ya que las pruebas fueron realizadas a 60 BPM).

#### **4.2.2. Prueba de Altura Tonal:**

Esta prueba también fue realizada en tres etapas: la primera toma como referencia la escala de C mayor en dos octavas (ascendentemente y descendentemente), cabe recalcar que se utiliza esta tonalidad en particular porque es la tonalidad guía para el resto de tonalidades mayores, ya que en un piano se la forma tocando todas las teclas blancas partiendo desde C hasta el C de la siguiente octava.

Al analizar tonalmente la escala se concluye que cualquier escala mayor está compuesta por cincotonos y dos semitonos, los mismos que están dispuestos entre el tercer y cuarto grados de la escala y el séptimo y octavo grados de la escala respectivamente; por otro lado la escala menor natural está compuesta igualmente por cinco tonos y dos semitonos, pero esta vez los semitonos están entre el segundo y tercer grados de la escala y el quinto y sexto grados de la escala respectivamente.

Por la similitud tonal entre las escalas: mayor y menor natural, se decidió por la escala de C mayor para la primera parte de la prueba tonal.

#### 4.2.2.1. Presentación de Resultados de la Prueba de Altura Tonal con una Escala de C Mayor:

**Tabla 4.2. Resultados de la Prueba de Altura Tonal con una Escala de C Mayor.**

<u>Instrumento</u>	<u>Prueba de Altura Tonal 1 [%]</u>
Pícolo	51.26
Flauta	53.57
Fagot	50.35
Corno Francés	57.14
Trompeta	47.82
Tuba	57.96
Piano	-
Violín	63.00
Chelo	46.43
Contrabajo	42.86
Guitarra Acústica	60.25

Guitarra Eléctrica	59.10
Guitarra Eléctrica-Distorsión	63.80
Bajo Eléctrico	46.43
Soprano (voz)	70.25
Contralto (voz)	65.00
Tenor (voz)	63.14
Bajo (voz)	68.92
<b>Promedio</b>	<b>56.90%</b>

#### **4.2.2.2. Análisis de Resultados de la Prueba de Altura Tonal con una Escala de C Mayor:**

En la prueba de altura tonal con una escala de C mayor, se obtuvo un resultado general del 56.90%, se puede apreciar que los timbres vocales tuvieron la mayor respuesta de detección de la nota, siendo el timbre de soprano el que mejor resultado obtuvo con un 70.25%. Los resultados con los timbres vocales varían con respecto al resto de instrumentos porque las pruebas de altura tonal para los timbres vocales, es distinta que para el resto de instrumentos. De los timbres instrumentales el que mejor respuesta tuvo fue la guitarra eléctrica con distorsión con un 63.80%; por otro lado el timbre que peor respuesta tuvo fue el contrabajo con un 42.86%.

La segunda parte de la prueba de altura tonal fue realizada con la escala cromática en dos octavas, dicha escala se la obtiene al tocar todas las notas en semitonos partiendo desde C hasta el C de la siguiente octava, esta prueba se la realizó para determinar el comportamiento del dispositivo en la menor distancia existente en el sistema tonal dodecafónico: el semitono.

#### 4.2.2.3. Presentación de Resultados de la Prueba de Altura Tonal con una Escala Cromática:

**Tabla 4.3. Resultados de la Prueba de Altura Tonal con una Escala Cromática.**

<u>Instrumento</u>	<u>Prueba de Altura Tonal 2 [%]</u>
Pícolo	38.00
Flauta	37.50
Fagot	36.48
Corno Francés	37.50
Trompeta	26.33
Tuba	25.28
Piano	-
Violín	43.00
Chelo	25.00
Contrabajo	33.33
Guitarra Acústica	28.65
Guitarra Eléctrica	24.33
Guitarra Eléctrica-Distorsión	40.25
Bajo Eléctrico	29.17
Soprano (voz)	70.25
Contralto (voz)	65.00
Tenor (voz)	63.14
Bajo (voz)	68.92
<b>Promedio</b>	<b>40.71</b>

#### **4.2.2.4. Análisis de Resultados de la Prueba de Altura Tonal con una Escala Cromática:**

En la prueba de altura tonal con una escala cromática, el dispositivo obtiene un resultado de 40.71%, siendo ésta la prueba de altura tonal donde se obtienen los peores resultados. Los timbres vocales son los que mejores resultados obtienen, siendo el timbre de soprano el que mejor resultado obtiene con un 68.33%; de los timbres instrumentales el que mejor resultado obtiene es el violín con un 43.00%; por otro lado el timbre que peor resultado obtiene es la guitarra eléctrica con un 24.33% de efectividad.

Finalmente la tercera parte de la prueba de altura tonal fue realizada en base a arpeggios (acordes interpretados una nota a la vez) basados en la tonalidad de C, de la siguiente manera: la prueba se inicia con el arpeggio de C mayor compuesto por las notas C, E y G ; seguido por el arpeggio de C menor compuesto por las notas C, Eb y G; en tercer lugar el arpeggio de C aumentado compuesto por C, E, G# y finalizando esta parte de la prueba con el arpeggio disminuido compuesto por las notas C, Eb y Gb; cabe señalar que todos los arpeggios se los realizó en dos octavas de manera ascendente y descendente.

Dentro de la tercera prueba de altura tonal cabe resaltar la prueba realizada al piano, que fue una prueba netamente armónica, es decir, con el acorde de C mayor tanto en su estado fundamental como en sus inversiones: C, E, G; E, G, C y G, C, E; estos acordes tienen figuración de blancas (un acorde cada dos segundos) y han sido tocados en cuatro octavas.

#### 4.2.2.5. Presentación de Resultados de la Prueba de Altura Tonal con Arpeggios y Acordes:

**Tabla 4.4. Resultados de la Prueba de Altura Tonal con Arpeggios y Acordes.**

<u>Instrumento</u>	<u>Prueba de Altura Tonal 3 [%]</u>
Pícolo	68.35
Flauta	66.67
Fagot	57.66
Corno Francés	58.33
Trompeta	50.25
Tuba	55.00
Piano	33.33
Violín	67.90
Chelo	66.67
Contrabajo	58.33
Guitarra Acústica	72.65
Guitarra Eléctrica	72.30
Guitarra Eléctrica-Distorsión	76.48
Bajo Eléctrico	75.00
Soprano (voz)	70.25
Contralto (voz)	65.00
Tenor (voz)	63.14
Bajo (voz)	68.92
<b>Promedio</b>	<b>63.68</b>

#### **4.2.2.6. Análisis de Resultados de la Prueba de Altura Tonal con Arpeggios y Acordes:**

Especial mención requiere la prueba de altura tonal con arpeggios y acordes, ya que esta es la única prueba de altura tonal en la que interviene el timbre de piano (solamente con acordes) con un 33.33% de efectividad, ya que en las pruebas realizadas de las tres notas de cada acorde el dispositivo solamente reconoce una, el timbre instrumental que mejores resultados obtiene con la prueba de altura tonal con arpeggios es la guitarra eléctrica con distorsión con un 76.48%; por otro lado el timbre instrumental que peor respuesta obtiene en la prueba de altura tonal con arpeggios es la trompeta con un 50.25%. En la prueba de altura tonal con arpeggios y acordes el sistema obtiene una respuesta general del 63.68%.

#### **4.2.3. Prueba Dinámica (Velocidad):**

Un parámetro muy importante dentro de la interpretación musical es la intensidad o dinámica (cambios de volumen) con la que se tocan las notas, dentro del protocolo MIDI, a esta diferencia dinámica se le denomina velocidad.

La prueba en voces humanas se las realizó por separado, utilizando una obra inédita para cuatro voces: soprano, contralto, tenor y bajo de una duración de 30 segundos y con 28 notas por cada voz, en las que se toma en cuenta para su evaluación la cantidad de notas acertadas y las variaciones de amplitud (velocidad).

La prueba de velocidad empieza desde un *piano* (nivel de intensidad sonora equivalente a suave) hasta un *forte* (nivel de intensidad sonora equivalente a fuerte), dispuesto de la siguiente manera, en cada una de las distintas etapas: escala de C mayor en negras, escala cromática en corcheas y arpeggios en semicorcheas, al inicio de cada fase la dinámica es equivalente a *piano* llegando a su máximo nivel (*forte*) a mitad de cada fase, para a partir de ahí retornar a *piano* al finalizar cada fase de pruebas, con lo cual se

determinará la velocidad medida por el secuenciador en el que se realiza la grabación de los mensajes de control resultantes del dispositivo.

#### 4.2.3.1. Presentación de Resultados de la Prueba Dinámica:

**Tabla 4.5. Resultados de la Prueba Dinámica.**

<u>Instrumento</u>	<u>Prueba Dinámica [%]</u>
Pícolo	57.00
Flauta	56.25
Fagot	57.00
Corno Francés	56.25
Trompeta	57.00
Tuba	57.00
Piano	57.00
Violín	57.00
Chelo	57.00
Contrabajo	56.25
Guitarra Acústica	57.00
Guitarra Eléctrica	57.00
Guitarra Eléctrica-Distorsión	57.00
Bajo Eléctrico	56.25
Soprano (voz)	90.00
Contralto (voz)	90.00
Tenor (voz)	90.00
Bajo (voz)	90.00
<b>Promedio</b>	<b>64.17</b>

#### **4.2.3.2. Análisis de Resultados de la Prueba Dinámica:**

En la prueba dinámica se obtiene un resultado general del 64.17%, se observa que los timbres instrumentales tienen una respuesta general del 57.00% ya que se obtiene una respuesta débil a la intensidad de la señal original, lo que se traduce en una curva de velocidad MIDI casi plana. Por otro lado, en el caso de los timbre vocales, la respuesta del dispositivo a la intensidad de la señal original es muy buena, con un resultado favorable en el 90% de las pruebas.

Tanto la prueba rítmica como la de altura tonal se las realiza sobre base a 131 notas para cada instrumento evaluado, con una duración de 68 segundos por cada instrumento, la prueba dinámica es evaluada por la velocidad obtenida en cada una de las notas registradas en el secuenciador MIDI del *software* Reason 4.0.

#### **4.2.3.3. Mediciones Adicionales:**

Al realizar las pruebas con tonos puros se obtiene una efectividad en la respuesta del dispositivo de un 90% en el rango completo de frecuencias (de 20 [Hz] a 20 [KHz]), por otro lado en el rango de frecuencias entre los 20 [Hz] y los 16 [KHz] la respuesta del dispositivo fue de un 100%, es decir, se producen errores en las frecuencias comprendidas entre los 16 [KHz] y los 20 [KHz].

## 4.3. Resumen de Resultados:

Tabla 4.6. Resumen de resultados.

Instrumento	Prueba Rítmica[%]	Prueba de Altura Tonal [%]			Prueba Dinámica (Velocidad) [%]
		1	2	3	
<b>Pícolo</b>	72.31	51.26	38.00	68.35	57.00
<b>Flauta</b>	70.00	53.57	37.50	66.67	56.25
<b>Fagot</b>	70.31	50.35	36.48	57.66	57.00
<b>Corno Francés</b>	67.19	57.14	37.50	58.33	56.25
<b>Trompeta</b>	54.00	47.82	26.33	50.25	57.00
<b>Tuba</b>	59.87	57.96	25.28	55.00	57.00
<b>Piano</b>	90.00	-	-	33.33	57.00
<b>Violín</b>	68.00	63.00	43.00	67.90	57.00
<b>Chelo</b>	65.63	46.43	25.00	66.67	57.00
<b>Contrabajo</b>	70.31	42.86	33.33	58.33	56.25
<b>Guitarra Acústica</b>	71.83	60.25	28.65	72.65	57.00
<b>Guitarra Eléctrica</b>	70.23	59.10	24.33	72.30	57.00
<b>Guitarra Eléctrica-Distorsión</b>	73.60	63.80	40.25	76.48	57.00
<b>Bajo Eléctrico</b>	70.31	46.43	29.17	75.00	56.25
<b>Soprano (voz)</b>	87.60	70.25	70.25	70.25	90.00
<b>Contralto (voz)</b>	85.00	65.00	65.00	65.00	90.00
<b>Tenor (voz)</b>	83.60	63.14	63.14	63.14	90.00
<b>Bajo (voz)</b>	81.26	68.92	68.92	68.92	90.00
<b>Total</b>	<b>72.84</b>	<b>56.90</b>	<b>40.71</b>	<b>63.68</b>	<b>64.17</b>

## 5. ESTUDIO ECONÓMICO DEL PROYECTO.

El estudio económico del presente proyecto, está dividido en dos partes, ya que se trata de un prototipo *hardware* que además de su diseño y desarrollo se han realizado las pruebas pertinentes en función de comprobar su funcionamiento en las instancias para las que fue diseñado.

Es así que el presente estudio económico se compone de dos secciones: la sección de diseño y desarrollo, y la sección de pruebas.

### 5.1. Diseño y Desarrollo:

La etapa de diseño fue elaborada bajo medios electrónicos utilizando un computador de gama media, y software de simulación electrónica, además se utilizó un software de programación y de compilación para la parte de la programación del microcontrolador PIC16F877, en la etapa de desarrollo se usó varios componentes electrónicos, tales como resistencias, capacitores, amplificadores operacionales, bornes, zócalos, etc.

**Tabla 5.1. Detalle de Costos de la Etapa de Diseño y Desarrollo.**

RUBRO	CANTIDAD	PRECIO	TOTAL
Computador	1	850.00	850.00
Conexión a Internet (Mensual)	10	28.12	281.20
Componentes Electrónicos	80	66.15	66.15
Software "Proteus 7.6"	1	249.00	249.00
Material Bibliográfico	1	15.00	15.00
Asesoría (Proyecto)	1	55.82	55.82
Movilización	1	100.00	100.00
<b>TOTAL</b>			<b>1617.17</b>

A continuación se detalla las características de cada uno de los componentes de las etapas de diseño y desarrollo.

- **Computador:** El computador utilizado para las etapas de diseño, desarrollo y además para la etapa de pruebas, es un computador con las siguientes características: Procesador Intel Core 2 Duo de 2.66 [GHz] de velocidad con una memoria RAM de 4 [GB].
- **Software:** El software utilizado en la etapa de diseño del dispositivo es un software de diseño y simulación electrónica llamado “Proteus 7.6” de la desarrolladora “*Labcenter Electronics*”; este software a su vez contiene dos partes: el software de diseño esquemático “ISIS” y el software de diseño de circuitos impresos “ARES”. Parte del software utilizado para el desarrollo es el software de uso libre “MPLAB 8.43”, dicho software fue utilizado para la programación y compilación del microcontrolador PIC16F877 utilizado como cerebro del dispositivo.
- **Componentes Electrónicos:** Dentro de los componentes electrónicos utilizados para el desarrollo del dispositivo están incluidos resistencias, capacitares, reguladores de voltaje, baquelita, suelda, leds, cristal oscilador, dentro de este apartado, cabe tomar en cuenta la compra del microcontrolador PIC16F877 con su respectivo programador.
- **Bibliografía:** Para el desarrollo del dispositivo fue necesario la investigación y consulta de textos de electrónica y programación de PICs, es así como se utilizó el texto “ELECTRÓNICA PRÁCTICA CON MICROCONTROLADORES PIC” de Santiago Corrales.
- **Asesoría:** Se requiere contratar los servicios de un experto en electrónica el cual fue de mucha utilidad tanto en el diseño del circuito impreso como en el ensamblaje de los componentes.
- **Movilización:** Este rubro es un estimado del gasto en movilización.

## 5.2. Pruebas.

La etapa de pruebas se realiza bajo la configuración de un estudio casero. Dentro de la descripción de los elementos utilizados para las pruebas del dispositivo, no es tomado en cuenta el computador, ya que es el mismo utilizado en la etapa de diseño y desarrollo.

**Tabla 5.2. Detalle de Costos de la Etapa de Pruebas.**

RUBRO	CANTIDAD	PRECIO	TOTAL
Digidesign Mbox 2 Pro	1	699.00	699.00
Reason 4.0	1	299.00	299.00
M-Audio MIDIsport Uno	1	39.00	39.00
Shure SM-81	1	349.00	349.00
<b>TOTAL</b>			<b>1386.00</b>

- **Interfaz de Audio Digidesign Mbox 2 Pro:** Interfaz hardware de audio de conexión *FireWire* con 6 entradas y 8 salidas, además de una entrada y salida MIDI (conectores DIN 5 pines).
- **Software de Producción Musical:** El *software* utilizado para obtener el banco de pruebas y para grabar los mensajes de control MIDI es Reason 4.0 que es un sistema de producción musical que incluye un secuenciador, varios sintetizadores, efectos y procesadores dinámicos, además de dos grandes librerías de sonido denominados “Factory Sound Bank” y “Orkester”.
- **Interfaz MIDI-USB:** Para la conexión del dispositivo hacia el computador para las pruebas se utilizó el interfaz MIDI-USB “M-Audio MIDIsport Uno” que tiene una entrada y una salida MIDI de 16 canales y un conector USB para la conexión al computador.
- **Micrófono Shure SM-81:** Micrófono de condensador cardiode con una respuesta de frecuencia de 20 [Hz] a 20 [KHz], fue usado para la grabación de voces para las muestras del banco de pruebas.

Como se dijo anteriormente los elementos usados para la evaluación y pruebas del dispositivo son pertenecientes a la configuración de un estudio casero, por tanto no fue necesaria la adquisición de ningún elemento para la evaluación del dispositivo, dichos elementos han sido tomados en cuenta en el presente estudio económico con el fin de dar un monto realista del costo del proyecto, el cual sumando las etapas de diseño y desarrollo, y la etapa de pruebas se tiene:

**Tabla 5.3. Detalle del Costo Total del Proyecto.**

<b>RUBRO</b>	<b>TOTAL</b>
Total de la etapa de diseño y desarrollo	1617.17
Total de la etapa de pruebas	1386.00
<b>TOTAL DEL PROYECTO</b>	<b>3003.17</b>

Hay que tomar en cuenta que en el caso que se requiera la industrialización del dispositivo, los costos bajarían porque ya no sería necesaria la etapa de diseño, a no ser que se hagan implementaciones futuras al proyecto, en tal caso, se volvería a utilizar tanto el *software* como el *hardware* ya existente, por tanto el costo de cada unidad es aproximadamente:

**Tabla 5.4. Detalle del Costo Unitario del Dispositivo.**

<b>RUBRO</b>	<b>CANTIDAD</b>	<b>PRECIO</b>	<b>TOTAL</b>
Componentes	80	11.50	11.50
Mano de Obra	1	5.00	5.00
<b>TOTAL</b>			<b>16.50</b>

### **5.3. Relación Costo – Beneficio.**

Aunque el proyecto tiene un costo relativamente elevado, hay que tomar en cuenta que el costo por unidad es bajo, ya que las ventajas que representa y sus posibilidades de aplicación son enormes en comparación con su precio. Característica general de los dispositivos electrónicos a nivel mundial.

En primer lugar el dispositivo puede ser usado como un controlador MIDI en tiempo real, lo que quiere decir que cualquier músico con un instrumento

acústico puede controlar cualquier dispositivo que funcione bajo el protocolo MIDI, como secuenciadores, *samplers*, sintetizadores, etc.

Otro gran beneficio del dispositivo es que puede ser utilizado conjuntamente con cualquier editor de notación musical, es decir, el usuario podría obtener la partitura de su interpretación en tiempo real, únicamente bastaría con conectar el dispositivo a un computador que contenga dicho programa (por ejemplo: Finale, Encore, Sibelius entre otros), y dicho programa guardaría la interpretación (los mensajes de control MIDI recibidos) como archivos de notación musical, los cuales podrán ser impresos o archivados para futuras interpretaciones.

Otra implementación del dispositivo es conectarlo a un sistema de iluminación que tenga como base el control MIDI, el cual podrá ser programado para que responda a la altura (número de nota MIDI), a la intensidad (velocidad) o ambas, y así la interpretación en un escenario por ejemplo estará relacionada directamente con la iluminación dando un mayor impacto tanto visual como auditivo.

Al funcionar como controlador MIDI, los instrumentistas más tradicionales, podrán experimentar otro tipo de sonoridades distintas, al usar *samplers* o sintetizadores como bancos de sonidos unidos al sonido emitido por su propio instrumento acústico.

Al ser un dispositivo concebido bajo los estándares del protocolo MIDI, las posibilidades de control son muy altas ya que el protocolo MIDI es un estándar en la industria de la música a nivel internacional.

## **6. IMPLEMENTACIONES FUTURAS AL PROYECTO.**

El presente proyecto está limitado a convertir las señales de audio de nivel de línea en mensajes de control MIDI, sin embargo al ser un prototipo puede incluirse funciones adicionales que ampliarían notablemente su rango de acción, entre otras están:

### **6.1. Micrófono Dinámico.**

Ya que el dispositivo se alimenta de señal de nivel de línea para procesarla y convertirla en mensajes de control MIDI, un micrófono dinámico, sería un complemento ideal, ya que en situaciones donde no se tenga un preamplificador (como el de una consola de audio por ejemplo) sería de gran utilidad agregar un sistema de transducción preamplificado que alimente al amplificador operacional LM324 (ver gráfico 3.2.).

La limitación con este micrófono vendría dada en instrumentos de baja frecuencia tales como el contrabajo, el fagot, la tuba, etc. Ya que se deberán hacer las pruebas necesarias en cuanto a respuesta de frecuencia del micrófono para que no excluya dichas frecuencias de los instrumentos antes citados. Dentro de esta implementación sería muy conveniente agregar un *switch* que permita controlar la entrada de la fuente (sea por línea o por el micrófono) ya que al no hacer esta diferenciación se corre el riesgo de ingresar sonidos ambientales o parásitos que produzcan mensajes de control MIDI incorrectos, así el usuario tendrá el control del origen de la señal a ser procesada.

### **6.2. Teclado Hexadecimal y *Display* LCD.**

En el diseño original del dispositivo, el canal MIDI es asignado mediante un *Dip Switch*, lo cual es incómodo para el usuario por dos razones: la primera es que los cuatro *switch* de dos posiciones, son muy pequeños, por tanto de difícil acceso y manejo; la segunda razón es que el número de canal se maneja

mediante numeración binaria, lo cual es un limitante grande para quien no sabe este tipo de numeración.

Mediante la inclusión de un teclado hexadecimal y un *display* LCD fácilmente se puede asignar el número de canal MIDI a través del cual se transmitirá la información, además se podrá incluir información como el *tempo* (golpes por minuto – BPM), número de compás, tonalidad, etc.

### **6.3. Variación del *Tempo*.**

El dispositivo trabaja con una velocidad constante de 60 golpes por minuto (60 BPM), ya que el microcontrolador PIC16F877 trabaja con un oscilador externo de 4 [MHz], es muy factible variar la programación para cambiar a voluntad el *tempo* del dispositivo a través del teclado hexadecimal, esta característica es muy útil debido a que el protocolo MIDI es asíncrono, es decir, trabaja simplemente con pulsos de reloj y no con un código de tiempo, de esta manera será mucho más efectiva la transmisión e interconexión entre dispositivos.

### **6.4. Metrónomo.**

Aprovechando la capacidad de variar el *tempo* a voluntad, se puede programar una salida del microcontrolador para que emita una frecuencia a través de un parlante y encienda un led por cada golpe, así se implementará un metrónomo en el sistema, el cual obviamente emitirá este sonido a voluntad del usuario que lo podrá desconectar cuando lo crea pertinente.

La utilidad de esta implementación se da tanto en la práctica del usuario como en los procesos de grabación. El parlante que estará en el dispositivo (el que emitirá la frecuencia) estará a su vez conectado con un *Jack* TRS de 1/8" que servirá para la inclusión de audífonos, el momento en que estos son conectados se desconectará el parlante y viceversa.

### **6.5. Variación de la Afinación del Instrumento.**

La detección de la frecuencia o *pitch* de la señal entrante se hace a través del período de la señal, el cual determina la nota MIDI a ser disparada. El estándar internacional de afinación es de 440 [Hz] (equivalente a la nota A de la tercera octava del piano), en muchos grupos instrumentales como las orquestas de cámara que interpretan repertorio barroco, la afinación varía, ante lo cual el dispositivo dispararía notas equivocadas.

Para resolver este inconveniente, el usuario podrá determinar la frecuencia a la cual afinó su instrumento, esto se hará añadiendo varias tablas de período correspondientes a distintas afinaciones, cuando el usuario determine la frecuencia a la que está afinado, está seleccionando la tabla en la que el microcontrolador basará su cálculo para disparar las distintas notas, el usuario deberá variar también la afinación en el dispositivo receptor (módulo de sonidos, *sampler*, etc.).

### **6.6. Memoria *Flash* de Almacenamiento de Datos.**

El dispositivo está diseñado para convertir señales de nivel de línea en mensajes de control MIDI, los cuales son transmitidos en tiempo real a cualquier dispositivo receptor (módulo de sonidos, *sampler*, etc.)

Es muy conveniente incluir una memoria flash en la que el usuario pueda almacenar los mensajes de control de su interpretación, la que podrá ser recobrada mediante una conexión USB a cualquier computador, estos archivos tendrán la extensión .mid reconocible en cualquier computador, de tal manera que el usuario podrá descargar sus interpretaciones en cualquier computador para su posterior edición, transmisión, entre otros.

## **7. CONCLUSIONES Y RECOMENDACIONES.**

### **7.1 Conclusiones.**

#### **7.1.1. Conclusión General:**

Como base en la investigación del protocolo MIDI, el diseño, desarrollo y pruebas del prototipo se concluye que sí es posible la correcta detección y conversión de una señal de audio de nivel de línea a mensajes de control MIDI, tomando en cuenta los parámetros de frecuencia y amplitud dentro del rango de frecuencias comprendido entre 50 [Hz] y 6 [KHz], mediante la correcta implementación electrónica y una programación adecuada usando como base tecnología de microcontroladores y circuitos integrados

#### **7.1.2. Conclusiones de Diseño y Desarrollo Experimental:**

- La etapa de diseño del dispositivo fue realizada mediante simulación computarizada, a través de *software* de diseño electrónico, sin embargo, llegada la etapa de desarrollo experimental, se comprobó la poca veracidad de la simulación computarizada, debido principalmente a el uso de componentes “ideales” por parte del *software* de diseño electrónico, lo que consecuentemente trajo problemas de incompatibilidad tanto en la sección electrónica, como en la sección de programación del microcontrolador, por lo que fue necesario dividir el proyecto en secciones a través de la implementación de varios circuitos y programas de prueba (tanto en el simulador, como en la realidad) que permitiesen abordar las diferentes secciones para solucionarlas de manera individual, para de esta manera llegar a la solución general del proyecto, siendo indispensable en el desarrollo de un proceso de investigación y experimentación la correlación paralela de la teoría, la simulación y la práctica, ya que mediante la experiencia obtenida en esta investigación, la implementación de esta política de trabajo da como fruto un ahorro considerable de recursos tanto materiales como de tiempo.

- Las secciones de prueba en las que el proyecto se divide son: detección de frecuencia, detección de canal, detección del nivel de voltaje, detección de presencia o ausencia de señal de nivel de línea, envío serial de información según los lineamientos de la norma MIDI 1.0. Cada una de estas etapas del proyecto ha sido abordada tanto desde el punto de vista de la teoría como de la práctica, comprobándose mediante la experimentación, que no siempre la realidad es fiel reflejo de la teoría, debido a factores como errores del operador, componentes defectuosos, implementación mal efectuada, entre otras.
- La detección de la frecuencia se la aborda desde dos puntos de vista: el primero de ellos es la determinación del periodo de la señal, este sistema de detección aunque válido desde el punto de vista teórico presenta muchos problemas prácticos tales como la descalibración de los *Timer* del microcontrolador, además de un rango de error extremadamente elevado, por lo que esta opción fue invalidada. El segundo punto de vista por el que fue abordado la detección de frecuencia fue a través de convertir la señal de nivel de línea en una señal de pulsos y contabilizar dichos pulsos para la determinación de la frecuencia, opción que ofrece un rango de error muchísimo menor, y que es especialmente eficiente con tonos puros, de lo que se observa que aunque en teoría ambos puntos de vista son correctos en la práctica la elección de uno de ellos, debe hacerse a través del análisis de resultados provenientes de la experiencia.
- La detección de los picos de voltaje es implementada mediante la digitalización de la señal y la posterior medición de la misma en términos de amplitud de voltaje, este dato es guardado en una variable la misma que mediante una sencilla operación matemática, determina el dato de velocidad MIDI para ser transmitido, se observa que aunque es un método válido, incrementa el tiempo de procesamiento de la señal, lo que se traduce en una detección no siempre correcta de la amplitud y

por consecuencia produce errores en el envío de los datos de velocidad MIDI.

- La implementación del circuito físico del dispositivo debió ser realizada varias veces debido factores como: la fragilidad de los componentes, la manipulación incorrecta de los mismos, la temperatura excesiva en el momento de la soldadura; se observa la necesidad del planteamiento de un margen de error real permitiendo tener una expectativa de respuesta más cercana a los resultados reales.

### 7.1.3. Conclusiones de Operación:

- El dispositivo es de operación sencilla ya que sus mandos se componen básicamente de un *switch* de encendido y apagado, un *dip switch* de cuatro puertos, mediante el cual se determina el canal MIDI por el que se enviará la información, y un potenciómetro rotatorio cuya función es el contraste de la pantalla de cristal líquido (LCD) del dispositivo, aunque la complejidad de operación es muy baja, se observa que al exceder los niveles de entrada el dispositivo genera errores en la transmisión de datos.
- La principal complicación en la operación del dispositivo consiste en que el operador debe conocer lenguaje binario, ya que el *dip switch* que determina el canal MIDI de envío de información funciona bajo este principio, como solución a este problema se implementó la pantalla de cristal líquido (LCD) donde además del mensaje de nota encendida, la frecuencia entrante y el número de nota MIDI, sale el número de canal por el que se está enviando la información, además de la complicación del lenguaje binario, resulta incomodo la operación del propio *dip switch*, ya que por el tamaño del componente es de difícil manipulación.
- Para determinar la frecuencia el dispositivo usa un tiempo de muestreo de un segundo, por consiguiente dispara una nota MIDI en ese período

de tiempo, transmitiendo de manera constante una nota por segundo, causando errores debido al tiempo de respuesta del dispositivo tanto en la transmisión de datos como en la detección del ritmo.

- En las pruebas realizadas se observa que cuando la entrada de audio no está conectada a ambos lados, es decir, a la fuente y al dispositivo, éste detecta esta ausencia de voltaje como un uno lógico, lo que trae como consecuencia que de cómo resultado el mensaje de nota encendida, el motivo de este inesperado efecto en el comportamiento del dispositivo, es que así no se esté enviando señal a través del cable, si está conectado a la fuente, ésta transmite un voltaje bajo (0.1 [V] aproximadamente) el mismo que el dispositivo usa para transmitir el mensaje de nota apagada, si este pequeño voltaje le es quitado, el dispositivo interpreta como nota encendida, como un aviso que la conexión está incompleta. Aunque presenta el mensaje de nota encendida el dispositivo no transmite ninguna nota al receptor, es un modo útil de saber si las conexiones están bien realizadas.

#### **7.1.4. Conclusiones de la Etapa de Pruebas:**

##### **7.1.4.1. Conclusiones de los Resultados de la Prueba Rítmica (ver apartado 4.2.1.)**

De los resultados de la prueba rítmica, se concluye que:

- No es posible disparar notas con distintas figuraciones musicales, debido al tiempo de muestreo del microcontrolador (1 [s]) para determinar la frecuencia y por consiguiente el número de nota MIDI a ser disparada, sería bueno acortar el tiempo de muestreo, ya que el período de la frecuencia más baja (20 [Hz]) es 50 [ms], es decir, el dispositivo no debe tener problemas para manejar tiempos de muestreo menores.

- No es posible transmitir duraciones de notas mayores a 1 [s] debido a que el circuito rectificador es insuficiente para volver a dar continuidad a la señal de pulsos que alimenta al microcontrolador.
- El timbre instrumental con el que mejor responde el dispositivo desde el punto de vista rítmico es la flauta p colo, debido a que la amplitud de sus arm nicos es peque a con respecto a la fundamental, esto ayuda a que la se al de pulsos salga lo m s exacta posible a la fundamental, pudiendo detectar de manera efectiva la figuraci n r tmica de la se al entrante.
- El timbre instrumental con el que peor responde el dispositivo desde el punto de vista r tmico es la trompeta, debido a que en este timbre en particular, la amplitud de sus arm nicos puede sobrepasar a la de la fundamental, dando valores equivocados a la salida del disparador "Schmitt" (circuito integrado que transforma las se ales de audio en se ales de pulsos) y por consiguiente mayor rango de error desde el punto de vista r tmico.
- La conversi n de se al de audio de nivel de l nea a se al de pulsos, se realiza mejor con timbres cuyo contenido de arm nicos es lo menor posible con respecto a la amplitud de la nota fundamental.

#### **7.1.4.2. Conclusiones de los Resultados de las Pruebas de Altura Tonal (ver apartado 4.2.2.):**

De los resultados obtenidos en las pruebas de altura tonal (prueba con una escala de C mayor, prueba con una escala crom tica y prueba de arpegios y acordes) se puede concluir que:

- Los errores en la detecci n de la frecuencia se da debido a que el contador da como resultado valores discretos, gran cantidad de notas musicales ( $A3 = 440$  [Hz] como referencia de afinaci n), no tienen su

frecuencia entera, sino decimal; esto da un inevitable error en la detección de la frecuencia.

- Se necesita un rango de frecuencias discretas por cada nota MIDI, aunque el dispositivo detecta de manera correcta la frecuencia, la nota disparada es la incorrecta debido al rango de frecuencias mencionado anteriormente.
- Una desventaja del sistema es que los instrumentos deben estar perfectamente afinados para que la detección de frecuencia, y su posterior número de nota MIDI sea la correcta, condición que en la vida real no se cumple por varios motivos como temperatura, humedad relativa del aire, falta de experiencia del instrumentista para afinar su instrumento, entre otras.
- La detección de la frecuencia con tonos puros como fuente fue del 90%, es decir, entre más periódica sea la señal de audio la detección de la frecuencia es mejor.
- El timbre instrumental con que el sistema obtuvo mejor respuesta melódica es el violín, debido a su contenido armónico de fácil conversión a señal de pulsos, además la tesitura que maneja el violín (altas frecuencias) le da un rango de frecuencias amplio por cada nota MIDI, ya que entre más elevada es la altura tonal, el rango de frecuencias entre nota y nota es mayor.
- En sentido general el sistema obtuvo los resultados más bajos con los timbres de los instrumentos de la familia de los metales (corno francés, trompeta y tuba) debido a su contenido armónico es tan alto dificulta la tarea de detección de frecuencia.
- El dispositivo no reconoce acordes, cuando ingresa un acorde al dispositivo, éste automáticamente trabaja con la frecuencia más alta,

para resolver este problema, sería apetecible poner para futuras aplicaciones un divisor de frecuencias después de la entrada de señal, que permita descomponer al acorde (o distintas notas en el caso de una señal con varios tipos de timbre como una canción por ejemplo) en sus frecuencias fundamentales, para la correcta detección de cada una de las frecuencias.

#### **7.1.4.3. Conclusiones de los Resultados de la Prueba Dinámica (ver apartado 4.2.3.2.)**

De los resultados obtenidos en la prueba dinámica, se concluye que:

- Se obtuvo una respuesta dinámica similar en todos los timbres, debido a que la velocidad de la nota MIDI se la obtiene en base a una comparación de voltajes, si éste no varía significativamente, la velocidad variará de manera mínima, o simplemente no variará.
- Los instrumentos con mayor presencia de graves generan una mejor variación de velocidad, debido a que las frecuencias graves contienen mayor cantidad de energía, en términos de voltaje significa mayor variación de voltaje al aumentar o disminuir la intensidad de la interpretación, en consecuencia la intensidad es mejor detectada por el sistema.
- Las muestras de audio seleccionadas para la realización de las pruebas, no muestran diferencias significativas en la dinámica, debido a que la interpretación de dichas muestras no tiene contrastes dinámicos grandes, razón por la cual se obtuvo en la mayoría de instrumentos una respuesta casi plana en la prueba dinámica.

## 7.2. Recomendaciones:

- Se recomienda, utilizar un oscilador de mayor frecuencia (20 [MHz]) que permita el procesamiento de información a una mayor velocidad, lo que conllevará un incremento significativo en la respuesta rítmica del dispositivo.
- Para precautelar la durabilidad del circuito y de sus componentes se recomienda el uso de un disipador de calor para el regulador de voltaje, ya que la carga que implica la pantalla de cristal líquido para el circuito, trae como consecuencia un recalentamiento del regulador de voltaje.
- Se recomienda implementar un circuito rectificador más potente, que permita una detección más adecuada de la duración de las notas, de tal manera que se puedan obtener del dispositivo notas de duración mayor.
- Para futuras aplicaciones, se recomienda implementar un teclado hexadecimal de fácil acceso, ya que es bastante incómodo el uso del *dip switch* para la asignación del canal de transmisión de datos.
- Es indispensable que la salida por la que se reproduce la señal original, no sea la misma por la que está saliendo la señal ya convertida, ya que esto causa una retroalimentación al sistema que impide operar eficientemente al sistema.

## 8. BIBLIOGRAFÍA.

[1] Miyara, Federico, *Acústica y Sistemas de Sonido*, 3<sup>ra</sup> edición, Buenos Aires, UNR Editora, 2003.

[2] Corrales, S., *Electrónica Práctica con Microcontroladores PIC*, Ecuador: Autor, 2006.

[3] MIDI Manufacturers Association Incorporated (1995-2008), *Tutorial: History of MIDI*, recuperado el 29 de Abril de 2009, de [http://www.midi.org/aboutmidi/tut\\_history.php](http://www.midi.org/aboutmidi/tut_history.php)

[4] Jordá, S. (2003), *MIDI*, recuperado el 6 de Noviembre de 2008, de <http://www.tecn.upf.es/~sjorda/ME2003/7-MIDI/ME03-7Midi.pdf>

[5] Jure, L. (N/A), Implementación MIDI 1.0, referencia comentada, recuperado el 16 de Junio de 2009, de <http://www.eumus.edu.uy/docentes/jure/midi/>

[6] Luis, R. (N/A), *Sistemas Microcontrolados*, recuperado el 4 de Septiembre de 2009, de <http://r-luis.xbot.es/pic1/pic01.html>

[7] Luis, R. (N/A), *Estructura Interna del Microcontrolador*, recuperado el 4 de Septiembre de 2009, de <http://r-luis.xbot.es/pic1/pic03.html>

[8] Martínez, E. (2007), *Conversión Analógico-Digital*, recuperado el 15 de Septiembre de 2009, de <http://www.eveliux.com/mx/conversion-analogico-digital-adc.php>

[9] Ayala, F. (N/A), *Tipos de transmisión de datos*, recuperado el 15 de Septiembre de 2009, de <http://www.monografias.com/trabajos17/tipos-transmision-datos/tipos-transmision-datos.shtml>

[10] Hispasonic (2002), El protocolo MIDI, recuperado el 29 de Abril de 2009, de <http://www.hispasonic.com/revista/protocolo-midi>

[11] Kioskea (N/A), *Transmisión de datos: Transmisión digital de datos*, recuperado el 15 de Septiembre de 2009, de <http://es.kioskea.net/contents/transmission/transnum.php3>

[12] Kioskea (N/A), *Transmisión de datos: Modos de transmisión*, recuperado el 15 de Septiembre de 2009, de <http://es.kioskea.net/contents/transmission/transmode.php3>

[13] Wikipedia (N/A), *Microcontrolador*, recuperado el 4 de Septiembre de 2009, de <http://es.wikipedia.org/wiki/Microcontrolador>

[14] Wikipedia (N/A), *Conversión analógica-digital*, recuperado el 15 de Septiembre de 2009, de [http://es.wikipedia.org/wiki/Conversión\\_analógica-digital](http://es.wikipedia.org/wiki/Conversión_analógica-digital)

[15] Wikipedia (N/A), *Disparador Schmitt*, recuperado el 6 de Enero de 2010, de [http://es.wikipedia.org/wiki/Disparador\\_Schmitt](http://es.wikipedia.org/wiki/Disparador_Schmitt)

[16] Wikipedia (N/A), *Amplificador Operacional*, recuperado el 6 de Enero de 2010, de [http://es.wikipedia.org/wiki/Amplificador\\_operacional](http://es.wikipedia.org/wiki/Amplificador_operacional)

[17] Wikipedia (N/A), *Anexo: Amplificadores operacionales con realimentación negativa*, recuperado el 6 de Enero de 2010, de [http://es.wikipedia.org/wiki/Anexo:Amplificadores\\_operacionales\\_con\\_realimentación\\_negativa](http://es.wikipedia.org/wiki/Anexo:Amplificadores_operacionales_con_realimentación_negativa)

## 9. ANEXOS

### Anexo 1: Tablas de Mensajes MIDI.

#### Tabla 9.1. Mensajes de Voz (Channel Voice).

Mensajes de Voz (Channel Voice)									
Mensaje	Binario	Byte de Estado		Byte de Datos 1			Byte de Datos 2		
		Hexadecimal	Decimal	Binario	Hexadecimal	Decimal	Binario	Hexadecimal	Decimal
Note Off	1000 0000 a 1000 1111	80 a 8F	128 a 143	0000 0000 a 0111 1111	00 a 7F	0 a 127	0000 0000 a 0111 1111	00 a 7F	0 a 127
Note On	1001 0000 a 1001 1111	90 a 9F	144 a 159	0000 0000 a 0111 1111	00 a 7F	0 a 127	0000 0000 a 0111 1111	00 a 7F	0 a 127
Presión Polifónica	1010 0000 a 1010 1111	A0 a AF	160 a 175	0000 0000 a 0111 1111	00 a 7F	0 a 127	0000 0000 a 0111 1111	00 a 7F	0 a 127
Control Change	1011 0000 a 1011 1111	B0 a BF	176 a 191	0000 0000 a 0111 1111	00 a 7F	0 a 127	0000 0000 a 0111 1111	00 a 7F	0 a 127
Program Change	1100 0000 a 1100 1111	C0 a CF	192 a 207	0000 0000 a 0111 1111	00 a 7F	0 a 127	-----	-----	-----
Channel Pressure	1101 0000 a 1101 1111	D0 a DF	208 a 223	0000 0000 a 0111 1111	00 a 7F	0 a 127	-----	-----	-----
Pitch Bend Change	1110 0000 a 1110 1111	E0 a EF	224 a 239	0000 0000 a 0111 1111	00 a 7F	0 a 127	0000 0000 a 0111 1111	00 a 7F	0 a 127

#### Tabla 9.2. Mensajes de Modo (Channel Mode).

Mensajes de Modo (Channel Mode)									
Mensaje	Binario	Byte de Estado		Byte de Datos 1			Byte de Datos 2		
		Hexadecimal	Decimal	Binario	Hexadecimal	Decimal	Binario	Hexadecimal	Decimal
Reset All Controllers	1011 0000 a 1011 1111	B0 a BF	176 a 191	0111 1001	79	121	0000 0000 ó 0111 1111	00 ó 7F	0 ó 127
Local Control	1011 0000 a 1011 1111	B0 a BF	176 a 191	0111 1010	7A	122	0000 0000 ó 0111 1111	00 ó 7F	0 ó 127
All Notes Off	1011 0000 a 1011 1111	B0 a BF	176 a 191	0111 1011	7B	123	0000 0000	00	0
Omni Mode Off	1011 0000 a 1011 1111	B0 a BF	176 a 191	0111 1100	7C	124	0000 0000	00	0
Omni Mode On	1100 0000 a 1100 1111	B0 a BF	176 a 191	0111 1101	7D	125	0000 0000	00	0
Mono Mode On	1100 0000 a 1100 1111	B0 a BF	176 a 191	0111 1110	7E	126	-----	-----	-----
Poly Mode On	1100 0000 a 1100 1111	B0 a BF	176 a 191	0111 1111	7F	127	0000 0000	00	0



## Anexo 2: Valores de Controladores.

**Tabla 9.5. Valores de Controladores MIDI.**

Decimal (número de controlador)	Binario	Hexadecimal	Controlador
01	0000 0001	01	Rueda de modulación
02	0000 0010	02	Breath controller
04	0000 0100	04	Foot controller
05	0000 0101	05	Portamento
07	0000 0111	07	Volumen Principal
08	0000 1000	08	Balance
10	0000 1010	0A	Paneo
11	0000 1011	0B	Control de Expresión
64	0100 0000	40	Sustain
65	0100 0001	41	Portamento
66	0100 0010	42	Pedal Sostenuto
67	0100 0011	43	Pedal Soft
91	0101 1011	5B	Efectos
92	0101 1100	5C	Trémolo
93	0101 1101	5D	Chorus
94	0101 1110	5E	Detune
95	0101 1111	5F	Phaser
121	0111 1001	79	Reset all controllers
122	0111 1010	7A	Local Control
123	0111 1011	7B	All Notes Off
124	0111 1100	7C	Omni Off
125	0111 1101	7D	Mono On
126	0111 1110	7E	Mono Off
127	0111 1111	7F	Poly On

**Anexo 3: Notas MIDI, Byte de Estado, Byte de Datos, Frecuencia y Período.**

**Tabla 9.6. Notas MIDI, Byte de Estado, Byte de Datos, Frecuencia y Período.**

NÚMERO	NOTA	BYTE DE ESTADO	BYTE DE DATOS 1	FRECUENCIA [Hz]	PERIODO [ms]
21	A0	1001 0000	0001 0101	27,5	36,36
22	A#0	1001 0000	0001 0110	29,14	34,32
23	B0	1001 0000	0001 0111	30,87	32,40
24	C1	1001 0000	0001 1000	32,70	30,58
25	C#1	1001 0000	0001 1001	34,65	28,86
26	D1	1001 0000	0001 1010	36,71	27,24
27	D#1	1001 0000	0001 1011	38,89	25,71
28	E1	1001 0000	0001 1100	41,20	24,27
29	F1	1001 0000	0001 1101	43,65	22,91
30	F#1	1001 0000	0001 1110	46,25	21,62
31	G1	1001 0000	0001 1111	49,00	20,41
32	G#1	1001 0000	0010 0000	51,91	19,26
33	A1	1001 0000	0010 0001	55,00	18,18
34	A#1	1001 0000	0010 0010	58,27	17,16
35	B1	1001 0000	0010 0011	61,74	16,20
36	C2	1001 0000	0010 0100	65,41	15,29
37	C#2	1001 0000	0010 0101	69,30	14,43
38	D2	1001 0000	0010 0110	73,42	13,62
39	D#2	1001 0000	0010 0111	77,78	12,86
40	E2	1001 0000	0010 1000	82,41	12,13
41	F2	1001 0000	0010 1001	87,31	11,45
42	F#2	1001 0000	0010 1010	92,50	10,81
43	G2	1001 0000	0010 1011	98,00	10,20
44	G#2	1001 0000	0010 1100	103,83	9,63
45	A2	1001 0000	0010 1101	110,00	9,09
46	A#2	1001 0000	0010 1110	116,54	8,58
47	B2	1001 0000	0010 1111	123,47	8,10
48	C3	1001 0000	0011 0000	130,81	7,64
49	C#3	1001 0000	0011 0001	138,59	7,22
50	D3	1001 0000	0011 0010	146,83	6,81
51	D#3	1001 0000	0011 0011	155,56	6,43
52	E3	1001 0000	0011 0100	164,81	6,07
53	F3	1001 0000	0011 0101	174,61	5,73
54	F#3	1001 0000	0011 0110	185,00	5,41
55	G3	1001 0000	0011 0111	196,00	5,10
56	G#3	1001 0000	0011 1000	207,65	4,82

57	A3	1001 0000	0011 1001	220,00	4,55
58	A#3	1001 0000	0011 1010	233,08	4,29
59	B3	1001 0000	0011 1011	246,94	4,05
60	C4	1001 0000	0011 1100	261,63	3,82
61	C#4	1001 0000	0011 1101	277,18	3,61
62	D4	1001 0000	0011 1110	293,66	3,41
63	D#4	1001 0000	0011 1111	311,13	3,21
64	E4	1001 0000	0100 0000	329,63	3,03
65	F4	1001 0000	0100 0001	349,23	2,86
66	F#4	1001 0000	0100 0010	369,99	2,70
67	G4	1001 0000	0100 0011	392,00	2,55
68	G#4	1001 0000	0100 0100	415,30	2,41
69	A4	1001 0000	0100 0101	440,00	2,27
70	A#4	1001 0000	0100 0110	466,16	2,15
71	B4	1001 0000	0100 0111	493,88	2,02
72	C5	1001 0000	0100 1000	523,25	1,91
73	C#5	1001 0000	0100 1001	554,37	1,80
74	D5	1001 0000	0100 1010	587,33	1,70
75	D#5	1001 0000	0100 1011	622,25	1,61
76	E5	1001 0000	0100 1100	659,26	1,52
77	F5	1001 0000	0100 1101	698,46	1,43
78	F#5	1001 0000	0100 1110	739,99	1,35
79	G5	1001 0000	0100 1111	783,99	1,28
80	G#5	1001 0000	0101 0000	830,61	1,20
81	A5	1001 0000	0101 0001	880,00	1,14
82	A#5	1001 0000	0101 0010	932,33	1,07
83	B5	1001 0000	0101 0011	987,77	1,01
84	C6	1001 0000	0101 0100	1046,50	0,96
85	C#6	1001 0000	0101 0101	1108,73	0,90
86	D6	1001 0000	0101 0110	1174,66	0,85
87	D#6	1001 0000	0101 0111	1244,51	0,80
88	E6	1001 0000	0101 1000	1318,51	0,76
89	F6	1001 0000	0101 1001	1396,91	0,72
90	F#6	1001 0000	0101 1010	1479,98	0,68
91	G6	1001 0000	0101 1011	1567,98	0,64
92	G#6	1001 0000	0101 1100	1661,22	0,60
93	A6	1001 0000	0101 1101	1760,00	0,57
94	A#6	1001 0000	0101 1110	1864,66	0,54
95	B6	1001 0000	0101 1111	1975,53	0,51
96	C7	1001 0000	0110 0000	2093,00	0,48
97	C#7	1001 0000	0110 0001	2217,46	0,45
98	D7	1001 0000	0110 0010	2349,32	0,43
99	D#7	1001 0000	0110 0011	2489,02	0,40

100	E7	1001 0000	0110 0100	2637,02	0,38
101	F7	1001 0000	0110 0101	2793,83	0,36
102	F#7	1001 0000	0110 0110	2959,96	0,34
103	G7	1001 0000	0110 0111	3135,96	0,32
104	G#7	1001 0000	0110 1000	3322,44	0,30
105	A7	1001 0000	0110 1001	3520,00	0,28
106	A#7	1001 0000	0110 1010	3729,31	0,27
107	B7	1001 0000	0110 1011	3951,07	0,25
108	C8	1001 0000	0110 1100	4186,01	0,24
109	C#8	1001 0000	0110 1101	4434,92	0,23
110	D8	1001 0000	0110 1110	4698,64	0,21
111	D#8	1001 0000	0110 1111	4978,03	0,20
112	E8	1001 0000	0111 0000	5274,04	0,19
113	F8	1001 0000	0111 0001	5587,65	0,18
114	F#8	1001 0000	0111 0010	5919,91	0,17
115	G8	1001 0000	0111 0011	6271,93	0,16
116	G#8	1001 0000	0111 0100	6644,88	0,15
117	A8	1001 0000	0111 0101	7040,00	0,14
118	A#8	1001 0000	0111 0110	7458,62	0,13
119	B8	1001 0000	0111 0111	7902,13	0,13
120	C9	1001 0000	0111 1000	8372,02	0,12
121	C#9	1001 0000	0111 1001	8869,84	0,11
122	D9	1001 0000	0111 1010	9397,27	0,11
123	D#9	1001 0000	0111 1011	9956,06	0,10
124	E9	1001 0000	0111 1100	10548,08	0,09
125	F9	1001 0000	0111 1101	11175,30	0,09
126	F#9	1001 0000	0111 1110	11839,82	0,08
127	G9	1001 0000	0111 1111	12543,85	0,08

## Anexo 4: Partituras del Banco de Pruebas.

**Banco de Pruebas**

Proyecto Audio-MIDI

Christian Moreira

$\text{♩} = 60$

The image displays a musical score for the piece "Banco de Pruebas" by Christian Moreira. The score is written for a full band and includes the following instruments: Piccolo, Flute, Bassoon, Horn in F, Trumpet in C, Tuba, Piano, Violin I, Cello, Contrabass, Acoustic Guitar, Electric Guitar, and Bass. The tempo is marked as  $\text{♩} = 60$ . The score is arranged in a standard orchestral format with staves for each instrument. The key signature is one sharp (F#) and the time signature is common time (C). The score consists of 16 measures. The Piccolo, Flute, and Acoustic Guitar parts feature a melodic line of eighth notes. The Bassoon, Horn in F, Trumpet in C, Tuba, Violin I, Cello, and Contrabass parts play a steady eighth-note accompaniment. The Piano part provides harmonic support with chords and single notes. The Electric Guitar and Bass parts play a rhythmic accompaniment of eighth notes.

2

The image shows a page of a musical score, page 101, starting at rehearsal mark 2. The score is arranged in a standard orchestral layout with multiple staves. The instruments listed on the left are: Picc., Fl., Bsn., Hn., C Tpt., Tuba, Pno., Vln. I, Vc., Cb., Ac. Gtr., E. Gtr., and Bass. The music is written in a key signature of one sharp (F#) and a common time signature (C). The score consists of 16 measures. The Piccolo, Flute, Bassoon, Horn, Trumpet, Trombone, Violin I, Viola, Cello, Acoustic Guitar, Electric Guitar, and Bass parts all play a rhythmic pattern of eighth notes. The Piano part is mostly silent, with some chords in the first two measures. The overall texture is dense and rhythmic.

21

Picc

Fl

Bsn

Hr

C Tpt

Tuba

Pno

Vln. I

Vc.

Cb.

Ac. Gtr.

E. Gtr.

Bass

3

Detailed description: This page of a musical score, numbered 102, contains 14 staves of music. The instruments are arranged from top to bottom as follows: Piccolo (Picc), Flute (Fl), Bassoon (Bsn), Horn (Hr), Trumpet (C Tpt), Tuba, Piano (Pno), Violin I (Vln. I), Viola (Vc.), Cello (Cb.), Acoustic Guitar (Ac. Gtr.), Electric Guitar (E. Gtr.), and Bass. The score begins at measure 21. The Piccolo, Flute, Bassoon, Horn, Trumpet, and Tuba parts feature complex rhythmic patterns with many beamed notes. The Piano part is mostly silent, indicated by rests. The Violin I, Viola, Cello, Acoustic Guitar, Electric Guitar, and Bass parts play a consistent rhythmic accompaniment. A fermata is placed over the final measure of the Piccolo part, which is marked with a '3' above it, indicating a triplet. The key signature has one flat (B-flat), and the time signature is 4/4.

4

Pcc

Fl

Bsn

Hn

C Tpt

Tuba

Pno

Vln. I

Vc

Ch

Ac. Gtr

E. Gtr

Bass

Anexo 5: Diagrama Esquemático del Dispositivo Diseñado.

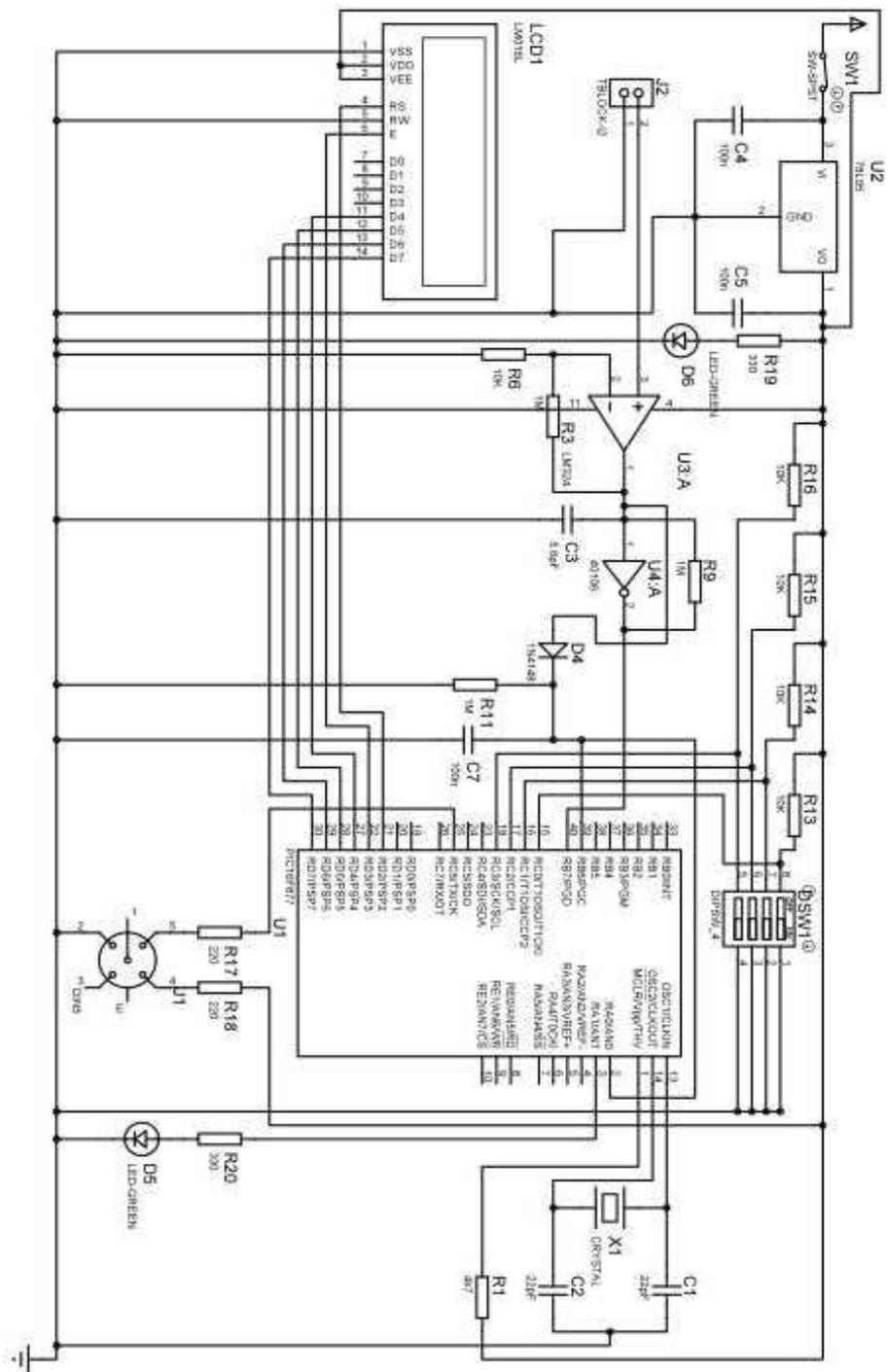


Figura 9.1. Diagrama Esquemático del Dispositivo Diseñado.

## Anexo 6: Diseño del Circuito Impreso.

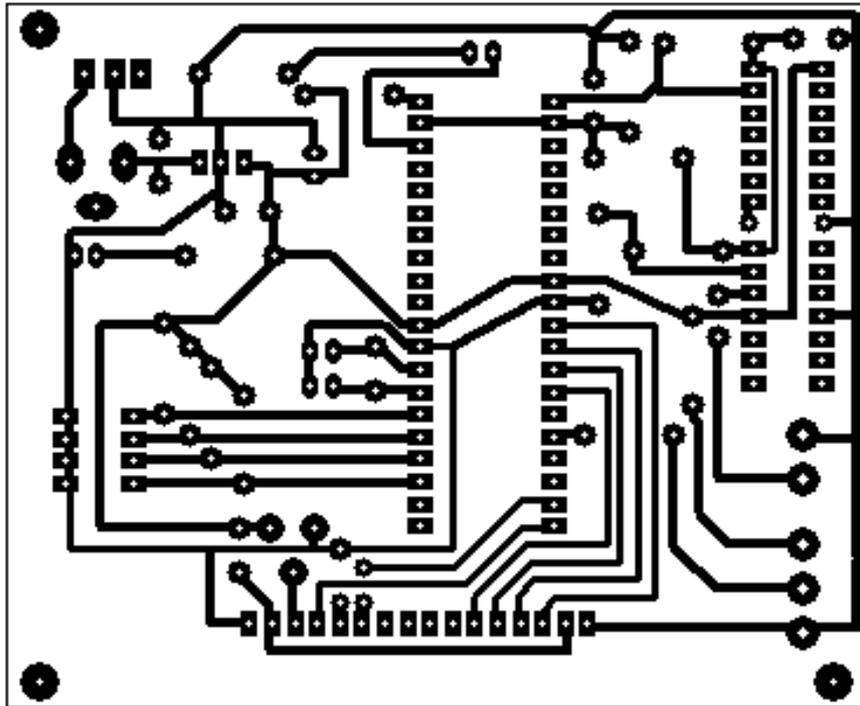


Figura 9.2. Diseño del Circuito Impreso.

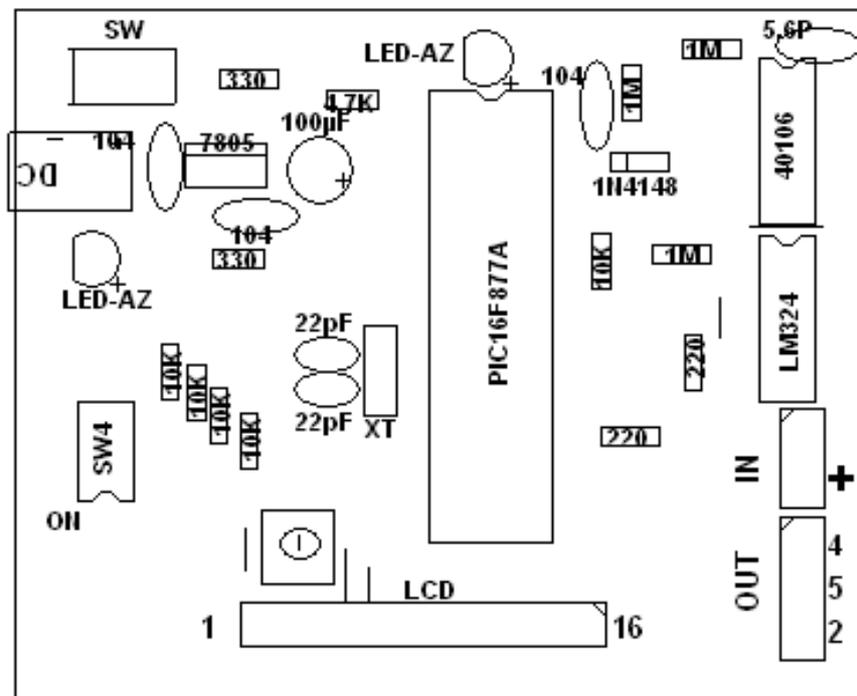


Figura 9.3. Disposición de los Elementos en el Circuito.

**Anexo 7: Programa del Microcontrolador PIC16F877A.****PROGRAMA DEL MICROCONTROLADOR PIC16F877A**

```
include "modedefs.bas"
CMCON=7
DEFINE HSER_RCSTA 90h
DEFINE HSER_TXSTA 20h
DEFINE HSER_BAUD 31250

DEFINE LCD_DREG PORTD
DEFINE LCD_DBIT 4
DEFINE LCD_RSREG PORTD
DEFINE LCD_RSBIT 2
DEFINE LCD_EREG PORTD
DEFINE LCD_EBIT 3

DEFINE ADC_BITS 8
DEFINE ADC_CLOCK 1
DEFINE ADC_SAMPLEUS 5

CANAL VAR BYTE
CANDEC VAR BYTE
NOTE VAR BYTE
NOTEON VAR BYTE
NOTEOFF VAR BYTE
ESTATUS1 VAR BYTE
ESTATUS2 VAR BYTE
LED VAR PORTA.1
FRECUENCIA VAR WORD
DATOS1 VAR BYTE
TRISA=%1
ADCON1=%00001110
```

INTENSIDAD VAR BYTE  
CALCULO VAR BYTE  
VELOCIDAD VAR BYTE

INICIO:

```
GOSUB CANALASIG
HSEROUT [NOTE]
GOSUB NUMERODENOTA
HSEROUT [DATOS1]
GOSUB VELOCITY
HSEROUT[velocidad]
LCDOUT $FE,1," ",HEX NOTE," ", DEC5 FRECUENCIA, " ", DEC3 DATOS1
```

GOTO INICIO

CANALASIG:

```
IF PORTC.0=0 and PORTC.1=0 AND PORTC.2=0 AND PORTC.3=0 THEN
CANAL =0
ENDIF
IF PORTC.0=1 and PORTC.1=0 AND PORTC.2=0 AND PORTC.3=0 THEN
CANAL =1
ENDIF
IF PORTC.0=0 and PORTC.1=1 AND PORTC.2=0 AND PORTC.3=0 THEN
CANAL =2
ENDIF
IF PORTC.0=1 and PORTC.1=1 AND PORTC.2=0 AND PORTC.3=0 THEN
CANAL =3
ENDIF
IF PORTC.0=0 and PORTC.1=0 AND PORTC.2=1 AND PORTC.3=0 THEN
CANAL =4
```

```
ENDIF
IF PORTC.0=1 and PORTC.1=0 AND PORTC.2=1 AND PORTC.3=0 THEN
CANAL =5
ENDIF
IF PORTC.0=0 and PORTC.1=1 AND PORTC.2=1 AND PORTC.3=0 THEN
CANAL =6
ENDIF
IF PORTC.0=1 and PORTC.1=1 AND PORTC.2=1 AND PORTC.3=0 THEN
CANAL =7
ENDIF
IF PORTC.0=0 and PORTC.1=0 AND PORTC.2=0 AND PORTC.3=1 THEN
CANAL =8
ENDIF
IF PORTC.0=1 and PORTC.1=0 AND PORTC.2=0 AND PORTC.3=1 THEN
CANAL =9
ENDIF
IF PORTC.0=0 and PORTC.1=1 AND PORTC.2=0 AND PORTC.3=1 THEN
CANAL =10
ENDIF
IF PORTC.0=1 and PORTC.1=1 AND PORTC.2=0 AND PORTC.3=1 THEN
CANAL =11
ENDIF
IF PORTC.0=0 and PORTC.1=0 AND PORTC.2=1 AND PORTC.3=1 THEN
CANAL =12
ENDIF
IF PORTC.0=1 and PORTC.1=0 AND PORTC.2=1 AND PORTC.3=1 THEN
CANAL =13
ENDIF
IF PORTC.0=0 and PORTC.1=1 AND PORTC.2=1 AND PORTC.3=1 THEN
CANAL =14
ENDIF
IF PORTC.0=1 and PORTC.1=1 AND PORTC.2=1 AND PORTC.3=1 THEN
```

```
CANAL =15  
ENDIF
```

```
CANDEC= CANAL+1
```

```
'LCDOUT $FE,1," CANAL= ",DEC CANDEC
```

```
NOTEOFF=128
```

```
NOTEON=144
```

```
ESTATUS1= NOTEON+CANAL
```

```
ESTATUS2= NOTEOFF+CANAL
```

```
IF PORTB.6=1 THEN
```

```
NOTE=ESTATUS1
```

```
ENDIF
```

```
IF PORTB.6=0 THEN
```

```
NOTE=ESTATUS2
```

```
ENDIF
```

```
IF NOTE=ESTATUS1 THEN
```

```
LED=1
```

```
ENDIF
```

```
IF NOTE=ESTATUS2 THEN
```

```
LED=0
```

```
ENDIF
```

```
RETURN
```

```
NUMERODENOTA:
```

```
COUNT PORTB.7,1000,FRECUENCIA
```

```
if FRECUENCIA >=27 MIN 28 THEN
```

```
DATOS1 =21
```

```
ENDIF
```

```
if FRECUENCIA=29 THEN
```

```
DATOS1 =22
```

```
ENDIF
```

```
if FRECUENCIA >=30 MIN 31 THEN
```

```
DATOS1 =23
```

```
ENDIF
```

```
if FRECUENCIA >=32 MIN 33 THEN
```

```
DATOS1 =24
```

```
ENDIF
```

```
if FRECUENCIA >=34 MIN 35 THEN
```

```
DATOS1 =25
```

```
ENDIF
```

```
if FRECUENCIA >=36 MIN 37 THEN
```

```
DATOS1 =26
```

```
ENDIF
```

```
if FRECUENCIA >=38 MIN 40 THEN
```

```
DATOS1 =27
```

```
ENDIF
```

```
if FRECUENCIA >=41 MIN 42 THEN
```

```
DATOS1 =28
```

```
ENDIF
```

```
if FRECUENCIA >=43 MIN 45 THEN
```

```
DATOS1 =29
```

```
ENDIF
```

```
if FRECUENCIA >=46 MIN 48 THEN
```

```
DATOS1 =30
```

```
ENDIF
```

```
if FRECUENCIA >=49 MIN 50 THEN
DATOS1 =31
ENDIF
if FRECUENCIA >=51 MIN 54 THEN
DATOS1 =32
ENDIF
if FRECUENCIA >=55 MIN 57 THEN
DATOS1 =33
ENDIF
if FRECUENCIA >=58 MIN 60 THEN
DATOS1 =34
ENDIF
if FRECUENCIA >=61 MIN 64 THEN
DATOS1 =35
ENDIF
if FRECUENCIA >=65 MIN 68 THEN
DATOS1 =36
ENDIF
if FRECUENCIA >=69 MIN 72 THEN
DATOS1 =37
ENDIF
if FRECUENCIA >=73 MIN 76 THEN
DATOS1 =38
ENDIF
if FRECUENCIA >=77 MIN 81 THEN
DATOS1 =39
ENDIF
if FRECUENCIA >=82 MIN 86 THEN
DATOS1 =40
ENDIF
if FRECUENCIA >=87 MIN 91 THEN
DATOS1 =41
```

```
ENDIF
if FRECUENCIA >=92 MIN 97 THEN
DATOS1 =42
ENDIF
if FRECUENCIA >=98 MIN 102 THEN
DATOS1 =43
ENDIF
if FRECUENCIA >=103 MIN 109 THEN
DATOS1 =44
ENDIF
if FRECUENCIA >=110 MIN 115 THEN
DATOS1 =45
ENDIF
if FRECUENCIA >=116 MIN 122 THEN
DATOS1 =46
ENDIF
if FRECUENCIA >=123 MIN 129 THEN
DATOS1 =47
ENDIF
if FRECUENCIA >=130 MIN 137 THEN
DATOS1 =48
ENDIF
if FRECUENCIA >=138 MIN 145 THEN
DATOS1 =49
ENDIF
if FRECUENCIA >=146 MIN 154 THEN
DATOS1 =50
ENDIF
if FRECUENCIA >=155 MIN 163 THEN
DATOS1 =51
ENDIF
if FRECUENCIA >=164 MIN 173 THEN
```

```
DATOS1 =52
ENDIF
if FRECUENCIA >=174 MIN 184 THEN
DATOS1 =53
ENDIF
if FRECUENCIA >=185 MIN 195 THEN
DATOS1 =54
ENDIF
if FRECUENCIA >=196 MIN 206 THEN
DATOS1 =55
ENDIF
if FRECUENCIA >=207 MIN 219 THEN
DATOS1 =56
ENDIF
if FRECUENCIA >=220 MIN 232 THEN
DATOS1 =57
ENDIF
if FRECUENCIA >=233 MIN 245 THEN
DATOS1 =58
ENDIF
if FRECUENCIA >=246 MIN 260 THEN
DATOS1 =59
ENDIF
if FRECUENCIA >=261 MIN 276 THEN
DATOS1 =60
ENDIF
if FRECUENCIA >=277 MIN 292 THEN
DATOS1 =61
ENDIF
if FRECUENCIA >=293 MIN 310 THEN
DATOS1 =62
ENDIF
```

```
if FRECUENCIA >=311 MIN 328 THEN
DATOS1 =63
ENDIF
if FRECUENCIA >=329 MIN 348 THEN
DATOS1 =64
ENDIF
if FRECUENCIA >=349 MIN 369 THEN
DATOS1 =65
ENDIF
if FRECUENCIA >=370 MIN 391 THEN
DATOS1 =66
ENDIF
if FRECUENCIA >=392 MIN 414 THEN
DATOS1 =67
ENDIF
if FRECUENCIA >=415 MIN 439 THEN
DATOS1 =68
ENDIF
if FRECUENCIA >=440 MIN 465 THEN
DATOS1 =69
ENDIF
if FRECUENCIA >=466 MIN 493 THEN
DATOS1 =70
ENDIF
if FRECUENCIA >=494 MIN 522 THEN
DATOS1 =71
ENDIF
if FRECUENCIA >=523 MIN 553 THEN
DATOS1 =72
ENDIF
if FRECUENCIA >=554 MIN 586 THEN
DATOS1 =73
```

```
ENDIF
if FRECUENCIA >=587 MIN 621 THEN
DATOS1 =74
ENDIF
if FRECUENCIA >=622 MIN 658 THEN
DATOS1 =75
ENDIF
if FRECUENCIA >=659 MIN 697 THEN
DATOS1 =76
ENDIF
if FRECUENCIA >=698 MIN 739 THEN
DATOS1 =77
ENDIF
if FRECUENCIA >=740 MIN 783 THEN
DATOS1 =78
ENDIF
if FRECUENCIA >=784 min 829 THEN
DATOS1 =79
ENDIF
if FRECUENCIA >=830 min 879 THEN
DATOS1 =80
ENDIF
if FRECUENCIA >=880 MIN 931 THEN
DATOS1 =81
ENDIF
if FRECUENCIA >=932 MIN 986 THEN
DATOS1 =82
ENDIF
if FRECUENCIA >=987 MIN 1045 THEN
DATOS1 =83
ENDIF
if FRECUENCIA >=1046 MIN 1108 THEN
```

```
DATOS1 =84
ENDIF
if FRECUENCIA >=1108 MIN 1173 THEN
DATOS1 =85
ENDIF
if FRECUENCIA >=1174 MIN 1243 THEN
DATOS1 =86
ENDIF
if FRECUENCIA >=1244 MIN 1317 THEN
DATOS1 =87
ENDIF
if FRECUENCIA >=1318 MIN 1396 THEN
DATOS1 =88
ENDIF
if FRECUENCIA >=1397 MIN 1479 THEN
DATOS1 =89
ENDIF
if FRECUENCIA >=1480 MIN 1567 THEN
DATOS1 =90
ENDIF
if FRECUENCIA >=1568 MIN 1660 THEN
DATOS1 =91
ENDIF
if FRECUENCIA >=1661 MIN 1759 THEN
DATOS1 =92
ENDIF
if FRECUENCIA >=1760 MIN 1863 THEN
DATOS1 =93
ENDIF
if FRECUENCIA >=1864 MIN 1974 THEN
DATOS1 =94
ENDIF
```

```
if FRECUENCIA >=1975 MIN 2092 THEN
DATOS1 =95
ENDIF
if FRECUENCIA >=2093 MIN 2216 THEN
DATOS1 =96
ENDIF
if FRECUENCIA >=2217 MIN 2348 THEN
DATOS1 =97
ENDIF
if FRECUENCIA >=2349 MIN 2488 THEN
DATOS1 =98
ENDIF
if FRECUENCIA >=2489 MIN 2636 THEN
DATOS1 =99
ENDIF
if FRECUENCIA >=2637 MIN 2793 THEN
DATOS1 =100
ENDIF
if FRECUENCIA >=2794 MIN 2959 THEN
DATOS1 =101
ENDIF
if FRECUENCIA >=2960 MIN 3135 THEN
DATOS1 =102
ENDIF
if FRECUENCIA >=3136 MIN 3321 THEN
DATOS1 =103
ENDIF
if FRECUENCIA >=3322 MIN 3519 THEN
DATOS1 =104
ENDIF
if FRECUENCIA >=3520 MIN 3728 THEN
DATOS1 =105
```

```
ENDIF
if FRECUENCIA >=3729 MIN 3950 THEN
DATOS1 =106
ENDIF
if FRECUENCIA >=3951 MIN 4185 THEN
DATOS1 =107
ENDIF
if FRECUENCIA >=4186 MIN 4434 THEN
DATOS1 =108
ENDIF
if FRECUENCIA >=4435 MIN 4697 THEN
DATOS1 =109
ENDIF
if FRECUENCIA >=4698 MIN 4977 THEN
DATOS1 =110
ENDIF
if FRECUENCIA >=4978 MIN 5273 THEN
DATOS1 =111
ENDIF
if FRECUENCIA >=5274 MIN 5586 THEN
DATOS1 =112
ENDIF
if FRECUENCIA >=5587 MIN 5918 THEN
DATOS1 =113
ENDIF
if FRECUENCIA >=5919 MIN 6271 THEN
DATOS1 =114
ENDIF
if FRECUENCIA >=6272 MIN 6644 THEN
DATOS1 =115
ENDIF
if FRECUENCIA >=6645 MIN 7039 THEN
```

```
DATOS1 =116
ENDIF
if FRECUENCIA >=7040 MIN 7457 THEN
DATOS1 =117
ENDIF
if FRECUENCIA >=7458 MIN 7907 THEN
DATOS1 =118
ENDIF
if FRECUENCIA >=7902 MIN 8371 THEN
DATOS1 =119
ENDIF
if FRECUENCIA >=8372 MIN 8869 THEN
DATOS1 =120
ENDIF
if FRECUENCIA >=8870 MIN 9396 THEN
DATOS1 =121
ENDIF
if FRECUENCIA >=9397 MIN 9955 THEN
DATOS1 =122
ENDIF
if FRECUENCIA >=9956 MIN 10547 THEN
DATOS1 =123
ENDIF
if FRECUENCIA >=10548 MIN 11174 THEN
DATOS1 =124
ENDIF
if FRECUENCIA >=11175 MIN 11839 THEN
DATOS1 =125
ENDIF
if FRECUENCIA >=11840 MIN 12543 THEN
DATOS1 =126
ENDIF
```

```
if FRECUENCIA >=12544 THEN  
DATOS1 =127  
ENDIF
```

```
RETURN
```

```
VELOCITY:
```

```
ADCIN 0,CALCULO  
INTENSIDAD=CALCULO*5/127  
PAUSE 500
```

```
IF INTENSIDAD=9 THEN  
VELOCIDAD=32  
ENDIF
```

```
IF INTENSIDAD=8 THEN  
VELOCIDAD=64  
ENDIF
```

```
IF INTENSIDAD=7 THEN  
VELOCIDAD=100  
ENDIF
```

```
RETURN
```

```
END
```

## Anexo 8: Glosario de Términos.

- **Circuito Sample & Hold:** Circuito diseñado para retener las muestras para su análisis en el proceso de digitalización de las señales análogas.
- **Conector Jack:** Denominación que se les da a los conectores hembras.
- **Key Code Interface:** Interface de comunicación entre instrumentos musicales desarrollado por Yamaha.
- **Least Significant Bit:** Bit menos significativo, se ubica en la última posición de un byte.
- **MIDI Thru Boxes:** Dispositivo para conexión en paralelo de dispositivos con protocolo MIDI.
- **Most Significant Bit:** Bit más significativo, se ubica en la primera posición de un byte.
- **Musical Instrument Digital Interface (MIDI):** Protocolo de comunicación digital, serial y asíncrona entre instrumentos musicales.
- **Nibble:** Conjunto de cuatro bits (letras digitales), dos nibbles forman un byte.
- **Patch de Sonidos:** Sonido o timbre proveniente de un módulo de sonidos, sampler u otro dispositivo MIDI, estos sonidos están agrupados por bancos de 128 sonidos.
- **Pitch Bend:** Variación gradual de la frecuencia de un sonido, generalmente manejada a través de una rueda.

- **Sampler:** Dispositivo que toma muestras de sonido y las procesa para producir diversidad de timbres.
- **Software:** Conjunto de comandos computacionales que forman un programa.
- **Track:** Pista en un secuenciador o canal en un grabador donde se guarda información acerca de la interpretación de un instrumento o de toda una canción en general.
- **Tempo:** Forma de denotar la velocidad de interpretación en una obra musical.
- **Tick:** División de la unidad de tiempo (nota negra) usada por el protocolo MIDI.
- **Voltage Gate:** Voltaje de corriente continua que permitía la comunicación entre dispositivos musicales antes de la aparición del protocolo MIDI.