



ESPE

**UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA**

UNIDAD DE GESTIÓN DE  TECNOLOGÍAS

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE TECNOLOGÍA EN ELECTRÓNICA MENCIÓN

INSTRUMENTACIÓN Y AVIÓNICA

**MONOGRAFÍA PREVIO A LA OBTENCIÓN DEL TÍTULO DE
TECNÓLOGO EN: ELECTRÓNICA MENCIÓN INSTRUMENTACIÓN Y
AVIÓNICA**

**TEMA: IMPLEMENTACIÓN DE UN SINTETIZADOR MEDIANTE
ARDUINO, PANTALLA TÁCTIL Y UN SHIELD MP3 PARA COMBINAR
SONIDOS Y FORMAR UNA MELODÍA MUSICAL**

AUTOR: MUSO TANDALLA, ANGEL MICHAEL

DIRECTOR: ING. ALPÚSIG CUICHÁN, SILVIA EMPERATRIZ

LATACUNGA

2020



ESPE

UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

UNIDAD DE GESTIÓN DE **TECNOLOGÍAS**

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE TECNOLOGÍA EN ELECTRÓNICA MENCIÓN

INSTRUMENTACIÓN Y AVIÓNICA.

CERTIFICACIÓN

Certifico que la presente monografía, *“IMPLEMENTACIÓN DE UN SINTETIZADOR MEDIANTE ARDUINO, PANTALLA TÁCTIL Y UN SHIELD MP3 PARA COMBINAR SONIDOS Y FORMAR UNA MELODÍA MUSICAL”* fue realizado por el señor *Muso Tandalla, Angel Michael* el mismo que ha sido revisado en su totalidad, analizado por la herramienta de verificación de similitud de contenido; por lo tanto cumple con los requisitos teóricos, científicos, técnicos, metodológicos y legales establecidos por la Universidad de Fuerzas Armadas ESPE, razón por la cual me permito acreditar y autorizar para que lo sustente públicamente.

Latacunga, 4 de diciembre del 2019.

FIRMA

Una firma manuscrita en tinta azul que parece decir 'Silvia' o similar, sobre un fondo blanco rectangular.

ING. ALPÚSIG CUICHÁN, SILVIA EMPERATRIZ

C.C.: 0502779697



ESPE

UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

UNIDAD DE GESTIÓN DE  TECNOLOGÍAS

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE TECNOLOGÍA EN ELECTRÓNICA MENCIÓN

INSTRUMENTACIÓN Y AVIÓNICA

AUTORÍA DE RESPONSABILIDAD

Yo, *Muso Tandalla, Angel Michael*, declaro que el contenido, ideas y criterios del trabajo de monografía: *“Implementación de un sintetizador mediante Arduino, pantalla táctil y un Shield Mp3 para combinar sonidos y formar una melodía musical”* es de mi autoría y responsabilidad, cumpliendo con los requisitos teóricos, científico, técnicos, metodológicos y legales establecidos por la Universidad de Fuerzas Armadas ESPE, respetando los derechos intelectuales de terceros y referenciando las citas bibliográficas.

Consecuentemente el contenido de la investigación mencionada es veraz.

Latacunga, 4 de diciembre del 2019.

Firma

MUSO TANDALLA, ANGEL MICHAEL

C.C.: 0503688871



ESPE

UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

UNIDAD DE GESTIÓN DE TECNOLOGÍAS

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA.

CARRERA DE TECNOLOGÍA EN ELECTRÓNICA MENCIÓN

INSTRUMENTACIÓN Y AVIÓNICA.

AUTORIZACIÓN

Yo, *Muso Tandalla, Angel Michael*, autorizo a la Universidad de las Fuerzas Armadas ESPE publicar en el trabajo de monografía: *“Implementación de un sintetizador mediante Arduino, pantalla táctil y un Shield Mp3 para combinar sonidos y formar una melodía musical”* en el Repositorio Institucional, cuyo contenido, ideas y criterios son de mi responsabilidad.

Latacunga, 4 de diciembre del 2019.

Firma

MUSO TANDALLA ANGEL MICHEL

C.C.: 0503688871

DEDICATORIA

La obtención de este título va dedicada, a mis padres Mario Muso y Mirian Tandalla que formaron en mí, los valores de responsabilidad, humildad, perseverancia y respeto, quienes me dieron la oportunidad de realizar una carrera para mi futuro.

A mis hermanos Neptali, Bryan y Marjorie, por creer en mí y brindarme su apoyo. A todas aquellas personas que nunca dudaron de mis capacidades.

Michael.

AGRADECIMIENTO

Agradezco a Dios por guiar mi camino y darme una familia maravillosa. A mis padres quienes me enseñaron a nunca darse por vencido, el valor del trabajo, esfuerzo y dedicación, gracias a ellos he podido obtener un logro más en mi vida.

A todas aquellas personas, quienes me brindaban sus sabias palabras, para nunca desistir de mi meta propuesta.

Michael.

ÍNDICE DE CONTENIDO

CARÁTULA

CERTIFICACIÓN.....	i
AUTORÍA DE RESPONSABILIDAD.....	ii
AUTORIZACIÓN.....	iii
DEDICATORIA.....	iv
AGRADECIMIENTO.....	v
ÍNDICE DE CONTENIDO.....	vi
ÍNDICE DE TABLAS.....	x
ÍNDICE DE FIGURAS.....	xi
RESUMEN.....	xiii
ABSTRACT.....	xiv

CAPÍTULO I

PLANTEAMIENTO DEL PROBLEMA DE INVESTIGACIÓN

1.1	Antecedentes.....	1
1.2	Planteamiento del Problema.....	2
1.3	Justificación e importancia.....	3
1.4	Objetivos.....	3
1.4.1	General.....	3
1.4.2	Específicos.....	3
1.5	Alcance.....	4

CAPÍTULO II

MARCO TEÓRICO

2.1	Sintetizador.....	5
2.2	Beneficios de la utilización de sintetizadores.....	6
2.2.1	Tipos de sintetizadores.....	7
2.2.2	Origen de las señales musicales.....	7
2.3	Elementos de un sintetizador.....	8
2.3.1	Teclado.....	8
2.3.2	Tarjetas de control electrónico.....	11
2.3.3	Arduino.....	11
2.3.4	Arduino Mega 2560.....	12
2.4	Descripción de los componentes de la placa de Arduino uno.....	13
2.4.1	Características.....	14
2.4.2	Shield MP3.....	15
2.5.	Características.....	16
2.5.1	Características Técnicas.....	17
2.5.2	Pantalla Nextion Editor TFT DE 7”.....	17
2.5.3	Nextion Editor.....	18

CAPÍTULO III

MARCO MEODOLÓGICO

3.1	Preliminares.....	22
3.2	Componentes para el desarrollo sintetizador de sonidos.....	23
3.3	Programación de la pantalla táctil Nextion.....	23

3.3.1	Configuración de la pantalla táctil Nextion	24
3.3.2	Programación del HMI Nextion	25
3.3.3	Programación de las diferentes páginas de la pantalla táctil.	26
3.3.4	Compilación y simulación del programa en la pantalla nextion.....	29
3.3.5	Carga del proyecto TFT	30
3.4	Estructura general del Proyecto.....	31
3.4.1	Librerías utilizadas	31
3.4.2	Pantalla nextion NX8048T070	31
3.5	Arduino uno	32
3.6	Diagrama de flujo de la programación Arduino uno	33
3.7	Programación de tarjeta Arduino uno.....	34
3.7.1	Declarar puertos.....	34
3.7.2	Declarar variables	35
3.7.3	Subrutina Principal	36
3.7.4	Subrutina para combinar sonidos	37
3.8	Programación de la tarjeta Arduino Mega.....	40
3.8.1	Diagrama de flujo	40
3.8.2	Declarar puertos.....	41
3.8.3	Declarar variables	43
3.8.4	Subrutina principal	46
3.8.5	Subrutina para los sonidos graves y agudos.....	48

CAPÍTULO IV

CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones.....57

4.2 Recomendaciones.....57

REFERENCIAS BIBLIOGRÁFICAS.....59

ANEXOS.....61

ÍNDICE DE TABLAS

Tabla 1 <i>Características técnicas Arduino Mega 2560</i>	14
Tabla 2 <i>Familia Arduino</i>	15
Tabla 3 <i>Especificación</i>	21
Tabla 4 <i>Fuente de Alimentación Recomendada: 5V, 2, DC</i>	21

ÍNDICE DE FIGURAS

<i>Figura 1.</i> Sintetizador-Teclado	5
<i>Figura 2.</i> Interfaz Usuario-Sintetizador Pro	6
<i>Figura 3.</i> Gráfico de la Onda Original.....	8
<i>Figura 4.</i> Teclado Electrónico	8
<i>Figura 5.</i> Frecuencia en HZ.....	9
<i>Figura 6.</i> Hardware Arduino Mega 2560	13
<i>Figura 7.</i> Módulo Shield MP3 VS 1053	16
<i>Figura 8.</i> Pantalla Nextion de Diferentes Tamaños.....	18
<i>Figura 9.</i> Interfaz Software Nextion.....	19
<i>Figura 10.</i> Estructura Física Pantalla Nextion.....	20
<i>Figura 11.</i> Parte frontal NX8048T070.....	20
<i>Figura 12.</i> Maqueta sintetizador de audio	22
<i>Figura 13.</i> Diagrama principal de conexiones.....	23
<i>Figura 14.</i> Selecciones HMI en Nextion Editor	24
<i>Figura 15.</i> Selecciones de orientación HMI	24
<i>Figura 16.</i> Entorno de programación de HMI en Nextion.....	25
<i>Figura 17.</i> Menú de Páginas y Toolbox en Nextion.....	26
<i>Figura 18.</i> Primera página	26
<i>Figura 19.</i> Selección de instrumento virtual o físico.....	28
<i>Figura 20.</i> Teclado Virtual	28
<i>Figura 21.</i> Teclado físico.....	29
<i>Figura 22.</i> Botones de Compilación y Simulación.....	30

Figura 23. Cargar o Subir Programa.....	30
Figura 24. Pantalla de muestro de carga de programa.....	31
Figura 25. Pantalla Nextion Subiendo Programa	31
Figura 26. Diagrama de bloques de la estructura general del software.....	32
Figura 27. Diagrama de flujo del programa principal	33
Figura 28. Diagrama de flujo del programa principal	43

RESUMEN

El presente trabajo de titulación está enfocado a la implementación de un sintetizador haciendo uso de una placa Arduino, una pantalla táctil y un Shield Mp3, para combinar sonidos y formar una melodía musical, con el cual se pretende mejorar la educación empleándolo como material didáctico para el laboratorio de instrumentación virtual de la universidad y además fomentar el interés a los estudiantes sobre el tema de investigación anteriormente mencionado. Para la realización del sintetizador de sonidos se empleó una pantalla táctil Nextion, una placa Arduino Mega y un Shield Mp3, el cual genera una interfaz entre la pantalla táctil y la placa Arduino Mega, enviando señales al teclado con el sonido seleccionado por el intérprete mediante la pantalla y el Shield Mp3 donde se hallan pregrabados los sonidos de diferentes instrumentos musicales. Con la realización de este proyecto se logra abrir un sin número de posibles campos de investigación para diferentes aplicaciones en el ámbito musical, en este caso se optó por la implementación en un sintetizador de sonido, debido a su facilidad de manejo, ya que no requiere mucho conocimiento previo para la realización en el cambio de sonidos por parte del intérprete y de esta manera conseguir formar una melodía.

PALABRAS CLAVE:

- **SINTETIZADORES**
- **SHIELD MP3**
- **PANTALLAS TÁCTIL**
- **SONIDO**

ABSTRACT

The present titling work is focused on the implementation of a synthesizer using an Arduino board, a touch screen and a Shield Mp3, to combine sounds and form a musical melody, with which it is intended to improve education using it as didactic material for the virtual instrumentation laboratory of the university and also encourage the interest to students on the aforementioned research topic. For the realization of the sound synthesizer, a Nextion touch screen, an Arduino Mega board and an Mp3 Shield were used, which generates an interface between the touch screen and the Arduino Mega board, sending signals to the keyboard with the sound selected by the interpreter through the screen and the Shield Mp3 where the sounds of different musical instruments are prerecorded. With the completion of this project it is possible to open a number of possible fields of research for different applications in the musical field, in this case we opted for the implementation in a sound synthesizer, due to its ease of handling, since it does not require much prior knowledge for the performance in the change of sounds by the interpreter and in this way get to form a melody.

KEY WORDS:

- **SYNTHESIZER.**
- **SHIELD MP3.**
- **TOUCH SCREEN.**
- **APPLICATIONS.**
- **SOUNDS.**

CAPÍTULO I

PLANTEAMIENTO DEL PROBLEMA DE INVESTIGACIÓN

1.1 Antecedentes

El uso de sintetizadores en el ámbito musical es esencial en el momento de crear melodías, debido a su funcionamiento, el incremento de sonidos musicales, en el mismo sintetizador será de mucha ayuda al realizar composiciones musicales.

Razón por la cual el uso de aprendizaje para sintetizadores en instituciones de educación superior es necesario, se presentan interpretaciones con aplicaciones. En referencia a la investigación de tesis, en favor al tema **“DESARROLLO DE UN SINTETIZADOR Y SU IMPLEMENTACIÓN EN PRODUCCIÓN MUSICAL”** realizado por Pinzón Luisa Fernanda y Rivero Ana María. El cual tiene como objetivo, realizar la experiencia del diseño sonoro de un nuevo camino a la creación artística.

En el presente trabajo investigativo con ARDUINO+MIDI, como lo menciona López, A. (2018) con su trabajo investigación **“IMPLEMENTACIÓN DE UN PIANO CONTROLADO MEDIANTE ARDUINO Y MIDI”**. Tiene por objetivo implementar sonidos pregrabados las cuales permitirán que al tocar el piano se pueda formar melodías musicales (pp.) 12.

En el trabajo de investigación con ARDUINO y MIDI con el tema **“DISEÑO Y DESARROLLO DE CONTROLADOR MIDI INALÁMBRICO DE VIENTO”**. (Ramos, 2016, p.1). El proyecto contempla el diseño y desarrollo de un dispositivo electrónico musical de viento, inalámbrico y de

tipo controlador MIDI, que permita producir música a cualquier individuo independientemente de sus conocimientos o pericia físico-técnica.

1.2 Planteamiento del Problema

El control de sonidos de un sintetizador comienza por la generación de señales eléctricas, por medio de las teclas negras, llamados sostenidos (#) o bemoles (b) y las teclas blancas que producen sonidos naturales, al combinar las teclas de una forma ordenada se obtendrá una melodía o composición musical.

En el día a día de un compositor o intérprete musical, la falta de sonidos producidos por un piano, en el instante de componer o interpretar canciones, las ha llevado a adquirir más de un piano, tomándoles costos muy altos.

Al desarrollar el sintetizador se podrá adquirir en el mismo teclado, los sonidos de los diferentes pianos como: CORG 364, CORG 264, YAMAHA 530 y YAMAHA 270, facilitando al compositor o intérprete a desenvolverse con más facilidad en los diferentes escenarios, en el momento de componer o interpretar.

El desarrollo de un “sintetizador” con visualización en una pantalla táctil, ayudará a los estudiantes de electrónica en el ámbito de programación, en distintos softwares. Dando una mejora a los compositores en sus habilidades y destrezas en la utilización de esta tecnología, de no darse esta implementación el aprendizaje de los futuros intérpretes se verá truncado puesto que no tendrían el conocimiento de estos dispositivos muy utilizados en el ámbito musical y electrónico.

1.3 Justificación e importancia

El presente proyecto tiene como objetivo principal desarrollar en una plataforma digital, un instrumento que ofrezca gran versatilidad sonora, así como la implementación del mismo en la producción musical de distintos géneros usando los conocimientos adquiridos en la carrera de electrónica. Con este argumento se quiso llegar a una propuesta en software y en hardware como superficie de control, la cual incluye a su vez la implementación y ampliación de los conocimientos en electrónica y diseño, así como la programación en Arduino y Nextion Editor

Con la implementación de un sintetizador se logrará solidas habilidades y destrezas en los alumnos de quinto y sexto nivel en el uso de ARDUINOS y PANTALLA TACTIL.

Su implementación es factible puesto que se dispone de manuales, trabajos de investigación y software de simulación.

1.4 Objetivos

1.4.1 General

Implementar un sintetizador mediante Arduino, pantalla táctil y un Shield Mp3 para combinar sonidos y formar una melodía musical.

1.4.2 Específicos

- Investigar los tipos de programación y características de la pantalla táctil, Nextion Editor.
- Implementar un teclado en la pantalla táctil Nextion Editor mediante una programación en el software Nextion.
- Implementar sonidos musicales mediante la programación en Arduino y Shield Mp3.

- Realizar pruebas de funcionamiento mediante la combinación de diferentes sonidos musicales.

1.5 Alcance

La implementación de un sintetizador, ayudará a la creación de composiciones o interpretaciones musicales, con la programación en un Shield Mp3 se podrá abarcar los 4 sonidos principales de los diferentes pianos, el mismo que se podrán ejecutar en la pantalla táctil.

Esto se lo realiza bajo una programación en los diferentes dispositivos que se va a utilizar, cumpliendo así con perfiles educativos en electrónica de altos estándares.

CAPÍTULO II

MARCO TEÓRICO

2.1 Sintetizador

Un sintetizador es, ante todo, y más allá de los aspectos tecnológicos, un instrumento musical: un aparato o utensilio capaz de generar sonidos a voluntad del intérprete, cuya principal finalidad o utilidad es ser usados en la producción musical. El nombre de “sintetizador” obedece a que el sonido es generado como resultado de un proceso de síntesis: un producto que es distinto de la mera yuxtaposición de los elementos que componen o intervienen en el proceso. (Martinez Zorrilla, 2008).



Figura 1. Sintetizador-Teclado
Fuente: (Hormazábal Azúa, 2017)

Debe destacarse también, para evitar confusiones, que, al menos estrictamente hablando, los sintetizadores no son “teclados”, o instrumentos accionados mediante un teclado (similar al de un piano), ya que, en sentido estricto, el sintetizador es el aparato que genera el sonido, independientemente del mecanismo utilizado para que un intérprete lo controle. Es cierto que la mayoría de sintetizadores son accionados mediante teclados, pero existen también otros que son controlables mediante guitarras eléctricas u otros instrumentos, e incluso módulos de sintetizador

(sintetizadores sin teclado, que pueden ser controlados mediante cualquier dispositivo externo del tipo adecuado que se le conecte, como un teclado, una guitarra, un controlador de viento –similar a una flauta o un saxofón). (Martinez Zorrilla, 2008).

2.2 Beneficios de la utilización de sintetizadores.

La mayoría de compositores e intérpretes musicales requieren sonidos que no se puede obtener en un solo piano, la cual el sintetizador abarca una memoria vacía en la cual se puede cargar o programar sonidos de distintos instrumentos musicales.



<	>	Название	S	M	Port	Ch	Ch2	Инструмент	Громкость	Баланс	Cho	Rev	Pha	Tre
1		Guitar 1			1	1	2	30 - Distortion Guitar	11	0	0	0	0	0
2		Guitar 2			1	3	4	30 - Distortion Guitar	12	0	0	0	0	0
3		Guitar 3			1	5	6	30 - Distortion Guitar	10	0	0	0	0	0
4		Piano			1	7	8	0 - Acoustic Grand Piano	10	0	0	0	0	0
5		Violin			1	9	11	40 - Violin	4	0	0	0	0	0
6		Bass			1	12	13	34 - Electric Bass (pick)	16	0	0	0	0	0
7		Drums			1	10	10	0 - Ударные 0	11	0	0	0	0	0

Figura 2. Interfaz Usuario-Sintetizador Pro

Fuente: (Hormazábal Azúa, 2017).

2.2.1 Tipos de sintetizadores

- **Sintetizadores monofónicos y polifónicos.** Un sintetizador es monofónico si únicamente es capaz de reproducir una nota (voz) simultáneamente, mientras que es polifónico si puede ejecutar dos o más notas de manera simultánea. La capacidad polifónica de los sintetizadores ha ido por regla general de la mano de los avances tecnológicos. Así, en sus inicios (hasta bien entrada la década de los 70) la práctica totalidad de los sintetizadores eran monofónicos, y paulatinamente los nuevos modelos fueron incrementando sus capacidades polifónicas, primero de manera modesta (con dos, cuatro, seis voces, etc.) y después de manera más significativa. Actualmente es

habitual encontrar sintetizadores con hasta 128 voces de polifonía. (Martinez Zorrilla, 2008).

- **Sintetizadores Monotímbricos y Multitímbricos.** La distinción responde a la capacidad o incapacidad de que un sintetizador reproduzca distintos timbres simultáneamente (como por ejemplo un sonido de bajo y un sonido de sección de cuerdas). Esta clasificación es relativamente independiente de la anterior, ya que, si bien es cierto que un sintetizador monofónico sólo puede ser monotímbrico, existen tanto sintetizadores polifónicos y monotímbricos como también polifónicos y multitímbricos. Cada distinto timbre se suele denominar “parte”. Actualmente existen muchos sintetizadores multitímbricos de 16, 32 y hasta 64 partes, lo que permite que un único instrumento pueda operar virtualmente como una orquesta completa, aunque siempre, claro está, dentro de los límites de su polifonía. La relación entre las partes y las voces puede ser estática (cada parte tiene asignada una cantidad determinada de voces de polifonía), o dinámica (cada parte puede utilizar tantas voces de polifonía como sea necesario, siempre que estén disponibles, y las que una parte no use, estarán disponibles para otras partes). (Martinez Zorrilla, 2008).

2.2.2 Origen de las señales musicales

El proceso de digitalización de una onda sonora supone “traducirla” a una secuencia numérica. Un CD, por ejemplo, contiene únicamente una serie de números, que son leídos por un láser y “traducidos” de nuevo a una señal analógica (a través de un DAC), que es enviada al amplificador y a los altavoces. El proceso de digitalización implica “cuadricular” en alguna medida el sonido original, con lo que se producirá siempre alguna pérdida (como contrapartida, las interferencias y el nivel de ruido propias de la tecnología analógica se reducen de manera importante, además de

disminuir el coste económico). Para entender mejor la idea, puede verse el siguiente gráfico, que representaría la onda original: (Martinez Zorrilla, 2008).

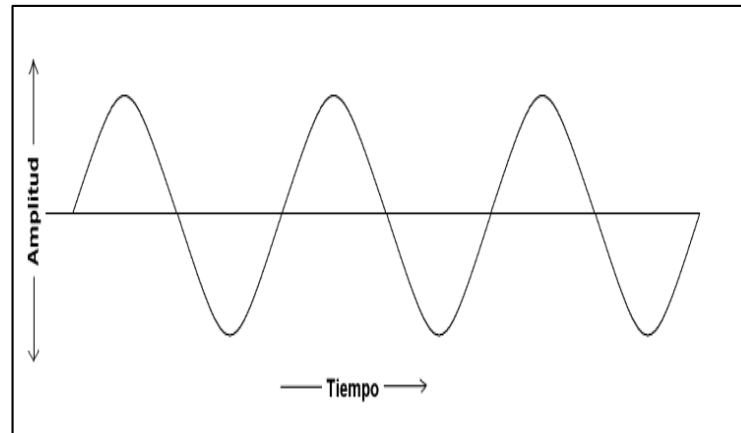


Figura 3. Gráfico de la Onda Original

Fuente: (Martinez Zorrilla, 2008)

2.3 Elementos de un sintetizador

2.3.1 Teclado

Para la producción de un sonido es necesario el teclado musical ya que es el medio por donde el ejecutante digitaliza diferentes melodías musicales, pulsando simultáneamente las teclas blancas que vendrán hacer notas naturales y las teclas negras las cuales serán notas con alteraciones ya sea sostenido (#) o (bemo). El teclado en la antigüedad era un instrumento de cuerda percutida ya que en el interior contiene cuerdas y martinetes la cual al pulsar una tecla el martinete golpea la cuerda seleccionada haciendo producir el sonido correcto, en la actualidad los teclados vienen ya con una placa electrónica la cual ya tiene programado sus notas musicales.



Figura 4. Teclado Electrónico

Fuente: (Hormazábal Azúa, 2017)

Los primeros problemas aparecieron con el ajuste de las frecuencias patrón de cada nota, tenía que encontrar una tabla de dichas frecuencias y me puse a buscar en todas partes (aún no se disponía de Internet). Y por fin como llovido del cielo y puesto que estaba abonado a ciertas revistas de electrónica, en una de ellas apareció una tabla ponderada de frecuencias de las notas, semitonos y octavas, como la que sigue. (García , 2010).

OCTAVAS		0	1	2	3	4	5	6
TONOS	UK							
EU	UK							
Do	C	16,3516	32,7062	65,4064	130,813	261,626	523,251	1046,50
Do [#]	C [#]	17,3239	34,6478	69,2957	138,591	277,183	554,365	1108,73
Re	D	18,3540	36,7081	73,4162	146,832	293,665	587,330	1174,66
Re [#]	D [#]	19,4454	38,8909	77,7817	155,563	311,127	622,254	1244,51
Mi	E	20,6017	41,2034	82,4069	164,814	329,628	659,255	1318,51
Fa	F	21,8268	43,6536	87,3071	174,614	349,228	698,456	1398,91
Fa [#]	F [#]	23,1247	46,2493	92,4986	184,997	369,994	739,989	1479,98
Sol	G	24,4997	48,9994	97,9989	195,998	391,995	783,991	1567,98
Sol [#]	G [#]	25,9565	51,9131	103,826	207,652	415,305	830,609	1661,22
La	A	27,500	55,0000	110,000	220,000	440,000	880,000	1760,00
La [#]	A [#]	29,1352	58,2705	116,541	233,082	466,164	932,328	1864,66
Si	B	30,8671	63,7354	123,471	246,942	493,883	987,767	1975,53

Figura 5. Frecuencia en HZ

Fuente: (García , 2010)

Desde el punto de vista físico, el sonido es una vibración del aire (en el vacío no hay sonido) que tiene ciertas características adecuadas para que en contacto con nuestro sistema auditivo pueda ser percibida por el cerebro como una señal audible (lo que comúnmente llamamos “un ruido” o “un sonido”). No cualquier vibración del aire da lugar a una percepción sonora, ya que las limitaciones del oído humano hacen ciertas vibraciones resulten inaudibles (por ejemplo, aquellas que se sitúan fuera de los límites de las frecuencias audibles, o aquellas vibraciones que resultan demasiado leves

para poder ser percibidas). Es posible destacar varias propiedades o aspectos básicos del sonido, que deben ser convenientemente diferenciados. Concretamente, serían los cuatro siguientes: 1) la frecuencia o altura; 2) la amplitud o intensidad; 3) la duración; y 4) el timbre. (Martinez Zorrilla, 2008).

- 1) **La frecuencia** (o altura) del sonido es, como sugiere su nombre, el lapso temporal entre las vibraciones cíclicas de las ondas sonoras. Estas vibraciones se miden en ciclos por segundo, a los que normalmente se denomina hertzios (Hz). Un hertzio corresponde a un ciclo (vibración) por segundo. La frecuencia o velocidad de las vibraciones es lo que nos permite clasificar los sonidos como “más agudos” (frecuencias más altas) o “más graves” (frecuencias más bajas): a medida que la frecuencia aumenta, el sonido se vuelve más agudo, y viceversa. Un ser humano con el oído normal es capaz de percibir sonidos aproximadamente entre los 20 Hz y los 20 KHz (20.000 Hz), si bien este límite superior va disminuyendo con la edad. (Martinez Zorrilla, 2008).

- 2) **La amplitud** (o intensidad) es lo que comúnmente asociamos al “volumen” o la fuerza del sonido, pudiendo distinguir así entre los distintos sonidos en función de si son más o menos “fuertes” o intensos (mayor o menor volumen). La amplitud se mide en decibelios (dB), situando el valor 0 en el umbral de la sensibilidad del oído humano normal (por tanto, todos los sonidos audibles tienen un valor positivo en decibelios). Teóricamente no existe un límite superior, si bien a partir de los 100 dB ya suele percibirse el sonido como desagradable y a partir de los 130 dB puede incluso producir dolor y lesiones. (Martinez Zorrilla, 2008)

- 3) **La duración** es el lapso temporal que va desde el inicio hasta la finalización del sonido. Si bien parece un concepto muy simple, en muchos casos pueden plantearse dudas, puesto del

sonido puede variar dinámicamente tanto en su frecuencia como en su intensidad o en su timbre, con lo cual puede resultar dudoso hasta qué punto estamos hablando del mismo sonido o de dos (o más) sonidos distintos. (Martinez Zorrilla, 2008).

- 4) **El timbre** es aquello que da cierto carácter distintivo al sonido, permitiendo identificarlo, por ejemplo, como un sonido de piano o de violín, o como la voz de Ana y no la de Carlos. Dos o más sonidos pueden ser idénticos en su frecuencia, amplitud y duración, y aun así distinguirse claramente por su distinto timbre. El timbre de un sonido depende de sus armónicos o parciales. Cualquier sonido, sea natural o producido artificialmente, con la excepción de una onda senoidal pura (que carece totalmente de armónicos), cuenta, además de con una frecuencia fundamental (por ejemplo, 440,0 Hz en el caso del piano), con otras frecuencias de onda de menor amplitud en otras alturas (los armónicos). En función de cuál sea la amplitud de estos armónicos y sus frecuencias (en qué lugar del espectro se encuentren), variará el timbre. En realidad, como puede demostrarse matemáticamente a Los sintetizadores). (Martinez Zorrilla, 2008).

2.3.2 Tarjetas de control electrónico

Son plataformas electrónicas basadas en hardware y software muy sencillos de usar; la plataforma basada en una placa de entradas y salidas simple y un entorno de programación libre es Arduino.

2.3.3 Arduino

Arduino es una plataforma de prototipos electrónica de código abierto (open-source) basada en hardware y software flexibles y fáciles de usar. Consta de una placa principal de componentes eléctricos, donde se encuentran conectados los controladores principales que gestionan los demás

complementos y circuitos ensamblados en la misma. Además, requiere de un lenguaje de programación para poder ser utilizado, programado y configurarlo de acuerdo con la necesidad del usuario. (Marmolejo, 2018).

2.3.4 Arduino Mega 2560

Arduino Mega 2560 es una tarjeta de desarrollo de Hardware libre construida con el microcontrolador Atmega 2560, que le da sentido a su nombre. Forma parte del proyecto Arduino que involucra una comunidad internacional dedicada al diseño y manufactura de placas de desarrollo de Hardware. (Veloso Critian, 2018).

Arduino logro su objetivo de facilitar y relacionar de manera simple y didáctica la programación de microcontroladores y la electrónica, dos áreas de la ingeniería muy complejas. (Veloso Critian, 2018).

Existen varios diseños de tarjetas diferentes, entre ellas se encuentra el **Arduino Mega 2560** que cuenta con una serie de características que en función del proyecto que necesitemos realizar pueden ser una ventaja. (Veloso Critian, 2018).

2.4 Descripción de los componentes de la placa Arduino Mega 2560

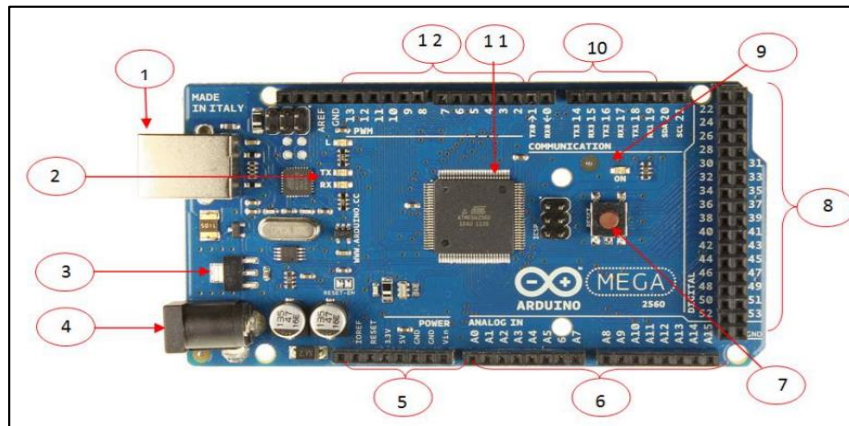


Figura 6. Hardware Arduino Mega 2560
Fuente: (Oña Chipuxi, 2015)

1. Conector USB .
2. Leds de RX y TX.
3. Regulador de voltaje 5V.
4. Plug de conexión para fuente de alimentación externa, el voltaje suministrado es 6V – 20V.
5. Puerto de conexión: Reset.
 - Pin 3.3 V.
 - Pin 5V.
 - Gnd.
 - Pin Vin.
6. Puertos de entrada analógica.
7. Botón Reset.
8. Puerto de entradas o salidas digitales.
9. Led on..
10. Pines Tx y Rx.

11. Microcontrolador atmega 2560.

12. Puertos de entrada o salida digitales, los pines en el Mega se pueden utilizar como una entrada o salida, usando (funciones) `pinMode`, `digitalWrite`, y `digitalRead`. Operan a 5 voltios. Cada pin puede proporcionar o recibir un máximo de 40 mA y tiene una resistencia de pull – up (desconectado por defecto) de 20-50 kOhms.

El Arduino Mega puede ser alimentado a través de la conexión USB o con una fuente de alimentación externa la cual se selecciona automáticamente. De fuente externa utilizando un adaptador de CA a CC o una batería conectando los pines GND y Vin de la placa. (Torrez Ismael, 2015).

2.4.1 Características

Tabla 1

Características técnicas Arduino Mega 2560

Voltaje de Operación	7 a 12 V
Voltaje de Entrada (limites)	6 – 20 V
Digital Pines E/S	54 (DE LOS CUALES 15 PROPORCIONAN SALIDA PWM)
Pines de entrada analógica	16
Corriente DC	40 Ma
Corriente CC	3.3 v 50 mA
Memoria Flash	256 KB (de los cuales 8 KB usados por bootloader)
SRAM	8KB
EEPROM	4KB
Velocidad del reloj	16 MHz

Fuente: (Ismael, 2015)

En el mercado se encuentra diferentes dispositivos que cumplen los requerimientos del sistema, pero por sus características se ha optado por Arduino Mega 2560, uno de los más utilizados en la actualidad. En la siguiente tabla se realiza un análisis del mismo con dispositivos de características similares: (Ismael, 2015).

Tabla 2
Familia Arduino

Características	Mega 2560	Uno	Leonardo	Due
Tipo de microcontrolador	Atmega 2560	Atmega 328	Atmega32U4	AT91SAM3X8E
Velocidad de reloj	16 MHz	16 MHz	16 MHz	84 MHz
Pines digitales E/S	54	14	20	54
Entradas Analógicas	16	6	12	12
Salidas analógicas	0	0	0	2 (DAC)
Memoria de programa (Flash)	256 kb	32 kb	32 kb	512 kb
Memoria de datos (SRAM)	8 kb	2 kb	2.5 kb	96 kb
Memoria Auxiliar (EEPROM)	4 kb	1 kb	1 kb	0 kb
Precio	\$ 80,00	\$ 35,00	\$ 50,00	\$75,00

Fuente: (Ismael, 2015)

2.4.3 Shield MP3

El módulo shield MP3 que contiene el CI VS2053, responsable de la generación de los distintos sonidos instrumentales.

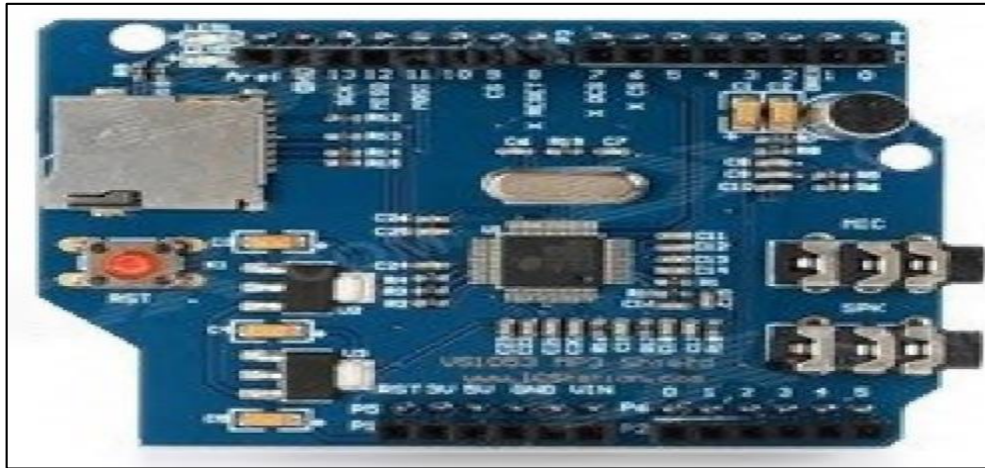


Figura 7. Módulo Shield MP3 VS 1053

Fuente: (Torres Valverde, 2015)

Este módulo posee un banco de melodías correspondiente a 128 instrumentos diferentes, que pueden ser seleccionados mediante instrucciones MIDI. Esta disponibilidad de instrumentos, con una amplia combinación de sonidos, permite diferentes opciones de estimulación al usuario en proceso de rehabilitación. (Torres Valverde, 2015)

El VS1053 recibe los datos a través de SPI. Después de que la transferencia ha sido decodificada por el IC, el audio es enviado a una entrada tipo audífonos (mini-plug), así como a un pasador de 2 pines de 0,1". Este shield viene equipado con todos los componentes que se muestran en las imágenes y en el esquemático; sin embargo, no incluye headers instalados. Recomendamos headers apilables de 6 y 8 pines. (Anónimo, s.f.)

2.5.1 Características

- Salida de audio de 3.5mm.
- Espaciamento de header de 0.1" para la bocina de salida.
- Ranura para tarjetas microSD.

2.5.2 Características Técnicas

- Puede reproducir una variedad de formatos de música, soporte para grabación en tiempo real de codificación OGG.
- Interfaz SPI, las líneas de señal de control salen a auriculares y salida estéreo.
- Un micrófono para grabar.
- Una interfaz de entrada line_in.
- Indicador de encendido.
- 3,3 V y 2,8 V de chip LDO AMS-1117 a bordo, proporciona hasta 800 mA de corriente..
- Una sola fuente de alimentación: +5 VDC..
- 12.288 Mhz de cristal.
- Ranura para tarjetas TF.

2.5.3 Pantalla Nextion Editor TFT DE 7”

Las pantallas Nextion como en la Figura 9, proporcionan una interfaz de control y visualización entre un usuario y un proceso, máquina, aplicación o sistema. Las pantallas Nextion TFT utilizan el puerto serie del Arduino para comunicarse, esto permite evitar la molestia de cableado excesivo como en otros modelos de pantallas. Nextion incluye parte de hardware (pantallas TFT) y la parte de software (el editor de Nextion) (Llumiyinga Leonardo, 2017)



Figura 8. Pantalla Nextion de Diferentes Tamaños

Fuente: (NextionHMI, 2017)

Una de las peculiaridades de Nextion es que la interfaz gráfica se carga en la memoria flash de la propia pantalla y no en el micro que controla el sistema, por tanto, el micro principal queda un poco más libre y el refresco de pantalla es mucho más rápido. Como la luz y el día. (Gómez Enrique & Gómez, 2017)

2.5.4 Nextion Editor

Nextion cuenta con un editor, en el cual se puede realizar interfaces para el usuario con botones, texto, barras de progreso, control deslizante, panel de instrumentos, etc. Todo lo necesario para una correcta visualización y control del sistema de estampado (Llumiquinga Leonardo, 2017)

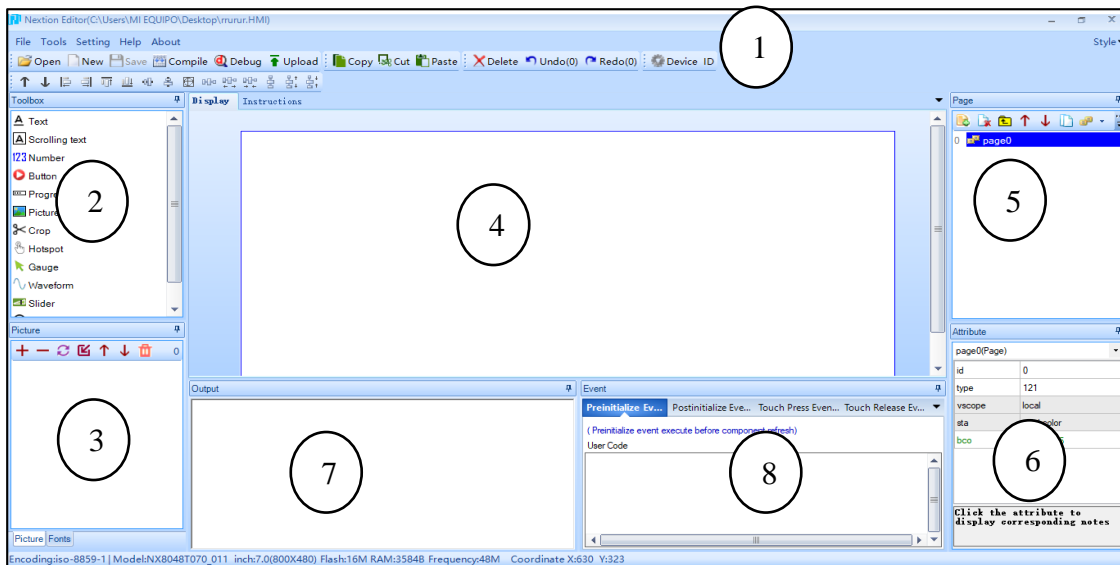


Figura 9. Interfaz Software Nextion

Fuente: (NextionHMI, 2017)

En la Figura 10 se tiene la interfaz del programa Nextion en la cual cada sección numerada es:

- 1) Menú principal.
- 2) Componentes.
- 3) Biblioteca de imágenes: aquí se ingresa las imágenes que se emplearan en el diseño de la interfaz.
- 4) Área de visualización.
- 5) Área de la página: Editar las páginas que contiene la interfaz.
- 6) Área de edición de atributos: Edite los atributos de componentes y páginas, como el valor, el color de un componente de texto.
- 7) Ventana de salida del compilador: si ocurren errores, se mostrarán en esta área.
- 8) Área del Evento: aquí se puede escribir código simple.

- **Pantalla táctil NX8048T070_011**

En su estructura presenta RGB driver, UN SD Card slot, flash, touch sensor, RGB Buffer y un puerto serial para comunicación con el computador o microcontrolador.

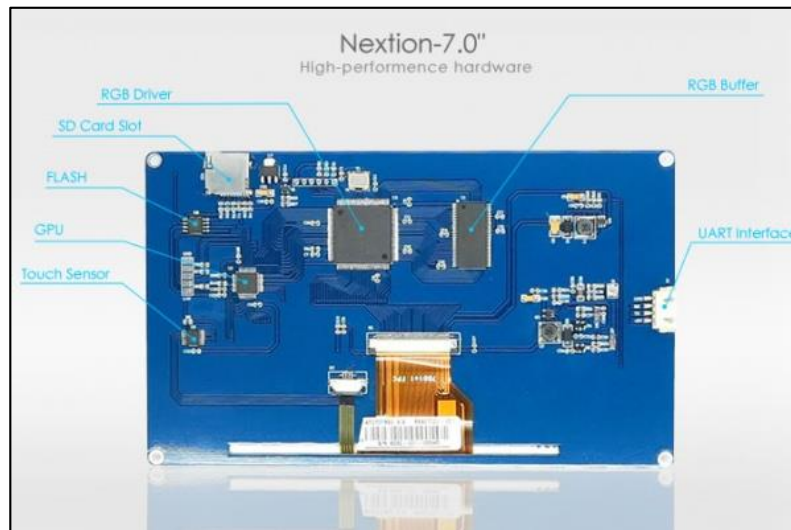


Figura 10. Estructura Física Pantalla Nextion
Fuente: (Pillajo Guitierrez, 2018)

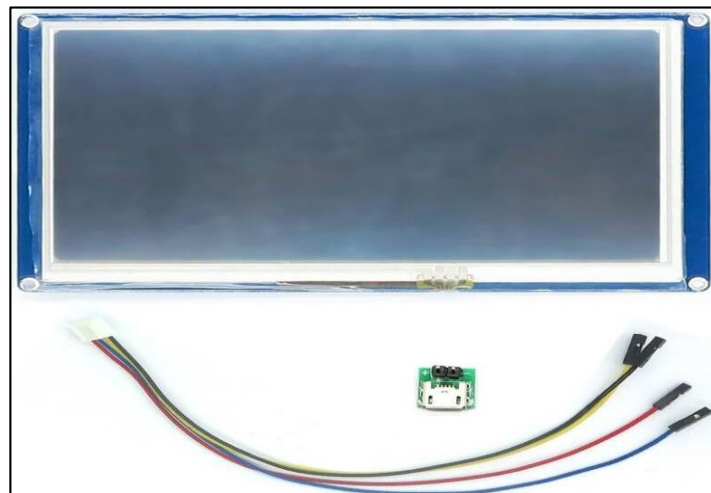


Figura 11. Parte frontal NX8048T070

- **Características**

Tabla 3*Especificación.*

	Datos	Descripción
Color	64 k 65536 colores	16 bit 565, 5R-6G-5B
Tamaño del diseño	181(L)*108(W)*7.3(H)	NX8048T070_011
Área Activa	164.9mm(L)*100mm(w)	
Área Visual	154.08mm(L)*85.92mm(w)	
Resolución	800*480 pixel	También se puede configurar como 480*800
Tipo de Toque	Resistivo	
Toques	> 1 millón	
Iluminación desde el fondo	Led	

Fuente: (NextionHMI, 2017)

Tabla 4*Fuente de Alimentación Recomendada: 5V, 2, DC.*

	Condiciones de prueba	Min	Típica	Max	Unidad
Tensión de funcionamiento		4.75	5	7	V
Tensión de corriente	VCC=+5V, el brillo es del 100%		510		mA
	Modo de sueño		15		mA

Fuente: (NextionHMI, 2017)

CAPÍTULO III

MARCO METODOLÓGICO

3.1 Preliminares

En el presente capítulo se detalla paso a paso como se realizó la ejecución de varios sonidos musicales en una pantalla táctil utilizando la tecnología Nextion Editor. La implementación de un sintetizador que se realizó se muestra en la Figura 13, que consta de una fuente de alimentación de 5 V, además de un teclado manual el cual será entonado por el intérprete. El diagrama de la Figura 14, simboliza las conexiones realizadas en la implementación de un sintetizador

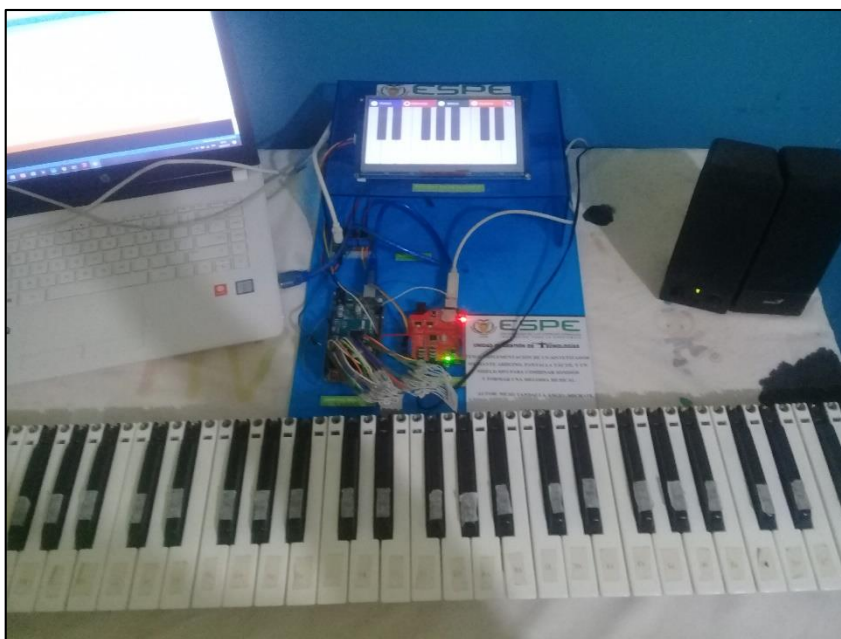


Figura 12. Maqueta sintetizadora de audio

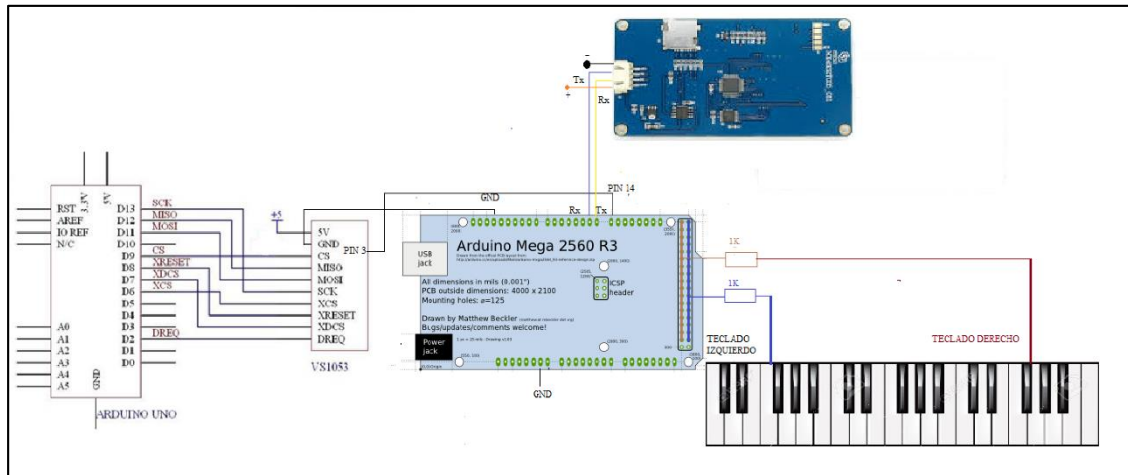


Figura 13. Diagrama principal de conexiones

3.2 Componentes para el desarrollo sintetizador de sonidos

Los componentes seleccionados para la implementación de un sintetizador y sonidos musicales son los siguientes:

- Pantalla táctil Nextion #NX8048T070.
- Shiel Mp3.
- Arduino Mega 2560.
- Teclado.
- Fuente de alimentación.
- Cables de conexión.

3.3 Programación de la pantalla táctil Nextion

Para realizar el programa principal del sintetizador de sonidos se ejecutaron los siguientes pasos:

- Configuración de la pantalla táctil Nextion.
- Programación del HMI Nextion.
- Compilación y simulación.
- Carga del proyecto TFT en la pantalla táctil.

3.3.1 Configuración de la pantalla táctil Nextion

A través del software Nextion Editor se seleccionó el modelo de la pantalla disponible, en este caso NX804T070 de 7 pulgadas con una calidad de imagen de 800 x 480 pixeles.

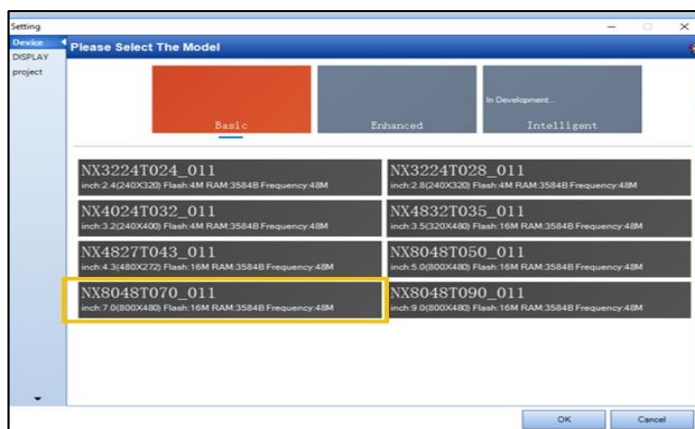


Figura 14. Selección HMI en Nextion Editor

A continuación, se debe seleccionar la orientación de la pantalla, pudiendo elegir entre cuatro opciones: Horizontal a cero grados, vertical a 90 grados, Horizontal a 180 grados y vertical a 270 grados, para este proyecto se seleccionará horizontal 0 grados.

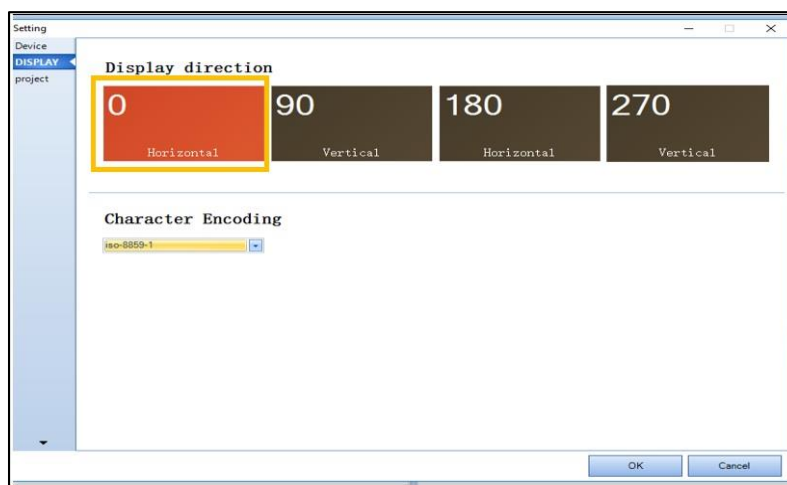


Figura 15. Selección de orientación HMI

3.3.2 Programación del HMI Nextion

Una vez configurada la pantalla, se procede a su desarrollo, en este caso Nextion Editor proporciona una interfaz muy intuitiva y con varios complementos fáciles de usar y de configurar

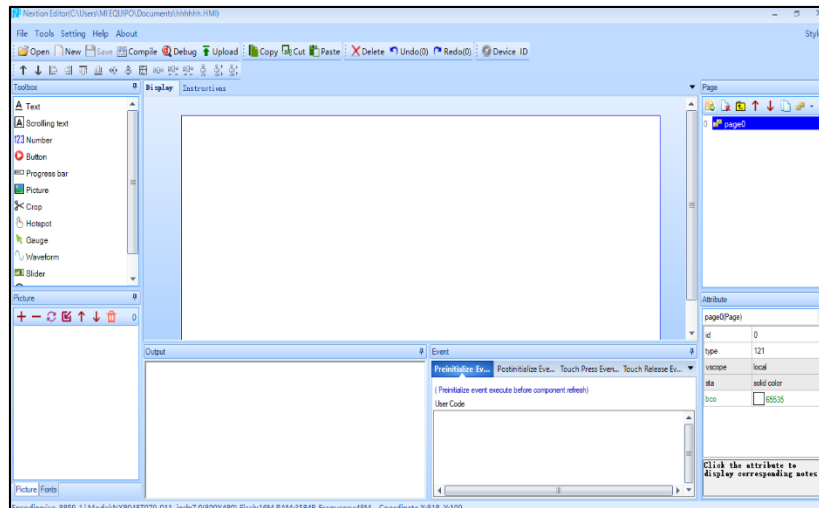


Figura 16. Entorno de programación de HMI en Nextion

Para comenzar con la programación del HMI se necesitan las imágenes que se vayan a colocar en las diferentes páginas, es importante que las imágenes a utilizar tengan una buena calidad y sean del tamaño de la pantalla en este caso 800 x 480 pixeles, al no tener esta condición las imágenes no se verán completas o se verán distorsionadas.

En la Figura 18 se detalla los diferentes toolbox de configuración presentes en el programa.



Figura 17. Menú de Páginas y Toolbox en Nextion

3.3.3 Programación de las diferentes páginas de la pantalla táctil.

a) Página 1

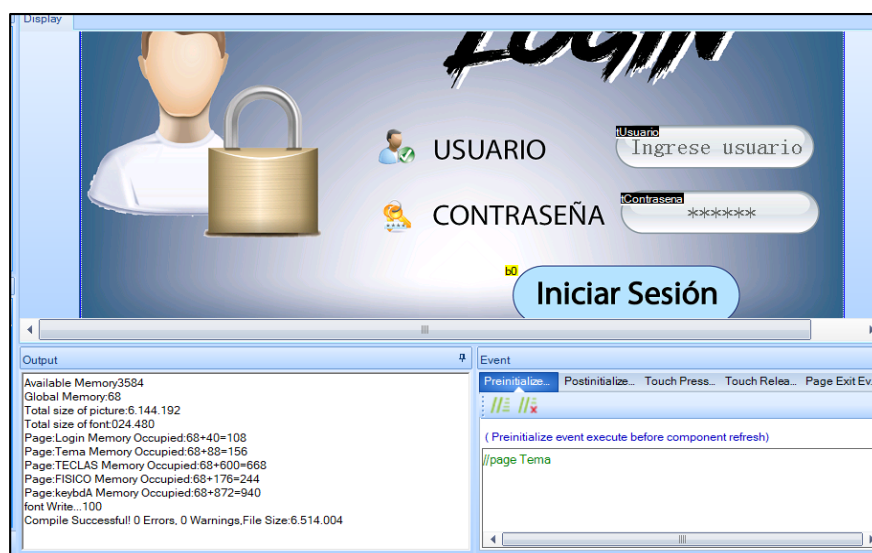


Figura 18. Primera página

Usuario

tUsuario.txt="": se ubica la contraseña seleccionada.

contraseña

tContrasena.txt="": se ubicará la contraseña seleccionada.

Iniciar sesión

Para poder ingresar a la siguiente página se deberá ingresar el usuario y la contraseña, mediante este programa se reconocerá la clave y el usuario correcto una vez realizado correctamente, abrirá la página siguiente y si alguna de las dos opciones es incorrecta no se podrá ir a la página 2.

```
if(tUsuario.txt=="ammuso")
{
  if(tContrasena.txt=="jesus1992")
  {
    tUsuario.txt="Ingrese Usuario"
    tContrasena.txt="*****"
    page Tema
  }
}
```

b) Página 2

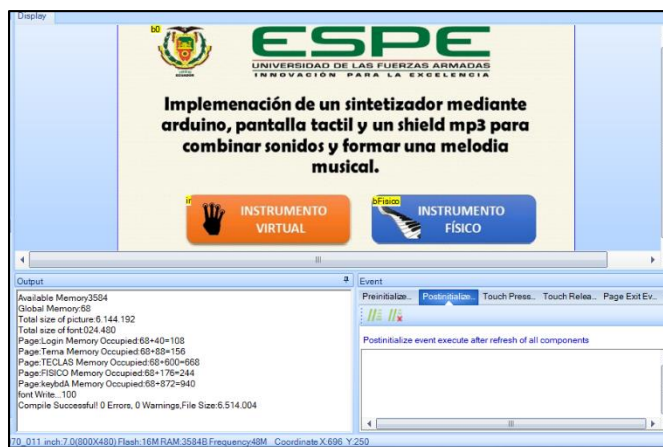


Figura 19. Selección de instrumento virtual o físico

Instrumento virtual (Para programar el teclado)

page TECLAS: Al pulsar uno de estos dos botones se ira al instrumento virtual o físico según lo seleccionado

Instrumento virtual

Al seleccionar un sonido de los 4 existentes el teclado virtual producirá una melodía mediante un intérprete con el sonido seleccionado

page TECLAS: Al seleccionar este botón se ira a la a las diferentes melodías

a) Página 3

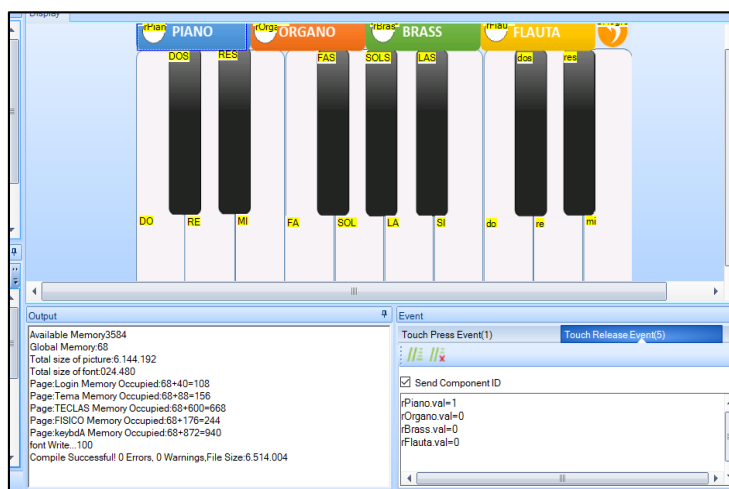


Figura 20. Teclado Virtual

Opción:**Piano**

Mediante es programa se seleccionará una sola melodía a la vez

rPiano.val=1

rOrgano.val=0

rBrass.val=0

rFlauta.val=0

Las demás opciones (Órgano, Brass y Flauta) tienen una programación similar a la del sonido Piano.

- a) **Página 4:** en esta página se puede seleccionar el tipo de sonido, piano, órgano, brass o flauta.

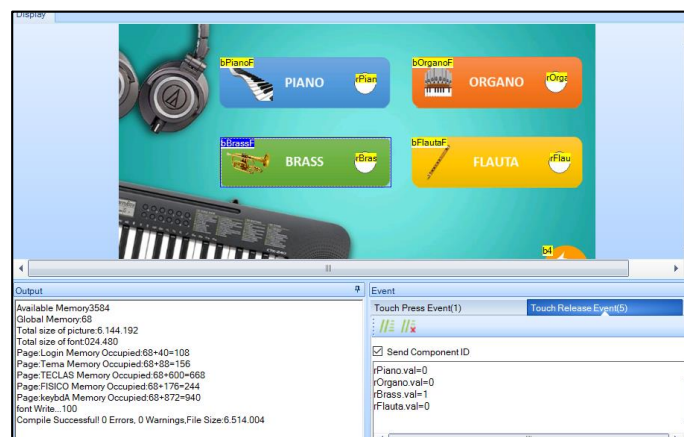


Figura 21. Teclado físico

3.3.4 Compilación y simulación del programa en la pantalla nextion

Una vez realizado la programación nos deberemos de fijar en 3 botones que aparecen en la barra de herramientas que son: **Compile**, **Debug** y **Upload**.

El botón compile nos indicará si hemos programado mal la pantalla TFT o si nos hemos excedido en el tamaño de la memoria que va a ocupar.

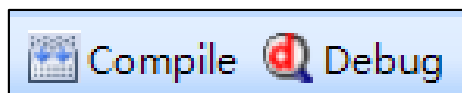


Figura 22. Botones de Compilación y Simulación

Para más seguridad, conviene hacer un Debug de la pantalla para revisar si el comportamiento que realiza la pantalla es el deseado, ya que, aunque la compilación sea satisfactoria, es posible que nos hayamos dejado algún elemento por programar correctamente y subir el programa tarda un rato.

3.3.5 Carga del proyecto TFT

Se puede realizar la carga del programa mediante una tarjeta SD o por medio de los puertos Tx y Rx de la pantalla táctil, en este caso se lo realizará por los puertos de la pantalla. Con la ayuda de un Arduino en el cual se conectará los pines en serie (Tx y Rx) de la pantalla y del puerto USB del Arduino conectar a la PC, para que la carga se exitosa.

Una vez conectado los puertos correctamente se deberá dar clic en el botón Uploap como se muestra en la Figura 24.

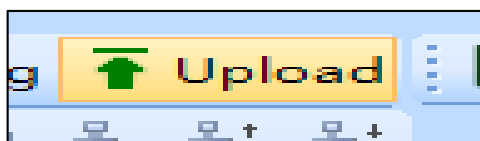


Figura 23. Opción del programa TFT para subir el programa a la pantalla

Una vez dado clic en el boto Uploap aparecerá una pantalla donde indica el estado de carga del programa, se deberá seleccionar el puerto de conexión de la pantalla en este caso COM 1 como se muestra en la Figura 25.

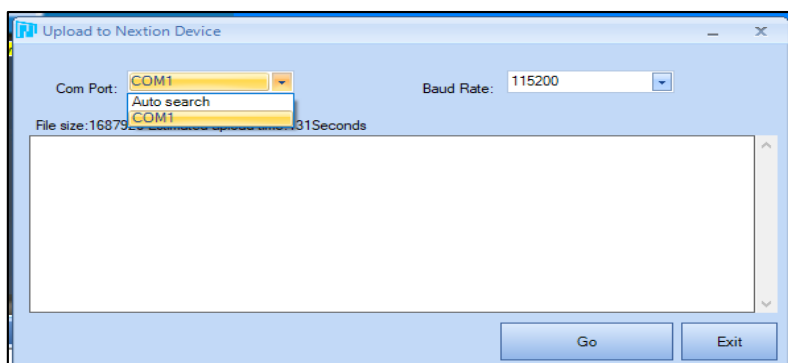


Figura 24. Pantalla de muestreo de carga de programa

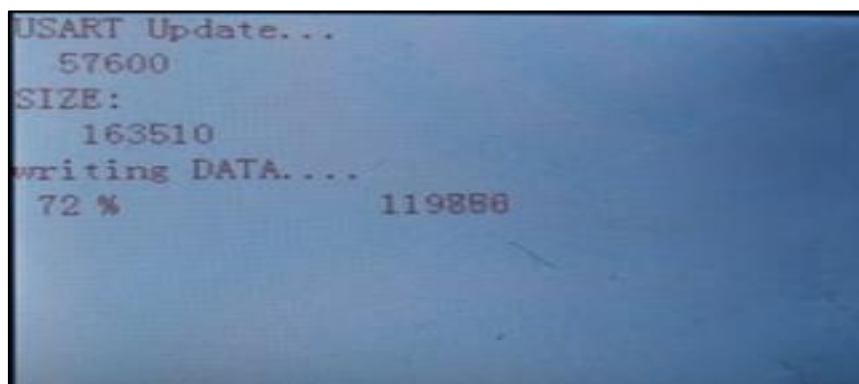


Figura 25. Pantalla Nextion Subiendo Programa

3.4 Estructura general del Proyecto

3.4.1 Librerías utilizadas

3.4.2 Pantalla nextion NX8048T070

- Librería Nextion: Esta librería nos permite básicamente la comunicación entre la pantalla táctil Nextion con la placa Arduino Mega.

3.5 Arduino uno

Para la programación, de la comunicación serial entre Shield mp3-Arduino Uno y Arduino Mega, se utilizó siguientes librerías

```
#include <SoftwareSerial.h>

#include <SPI.h>

#include <SdFat.h

#include <SFEMP3Shield.h>

#include <Bounce2.h>
```

Arduino Mega

Para la programación, de la comunicación serial entre Arduino mega y la pantalla táctil nextion, se utilizó siguientes librerías.

```
#include "Nextion.h"
```

Además, se realizó un programa principal en el cual se establecen los parámetros y condiciones para la realización el sintetizador de sonido. En la figura 27 se muestra un diagrama de bloques de la estructura general del software.

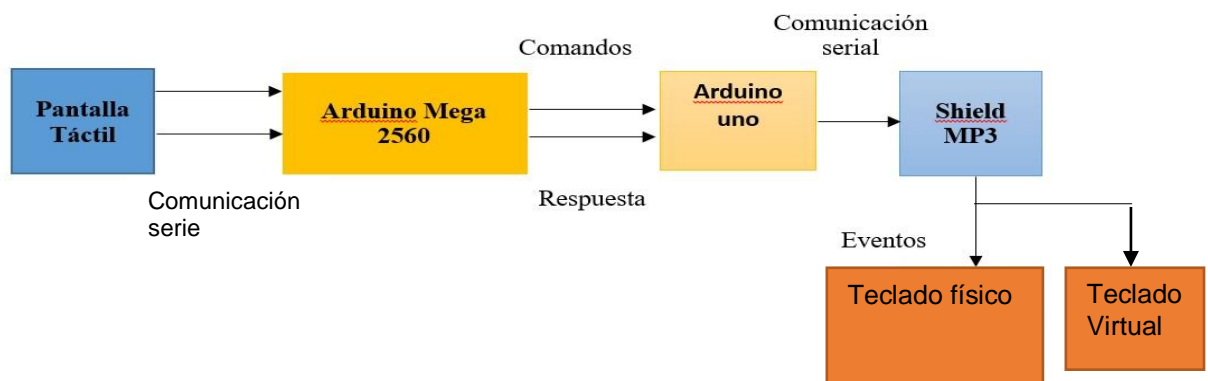


Figura 26. Diagrama de bloques de la estructura general del software²⁷

3.6 Diagrama de flujo de la programación Arduino uno

El diagrama de flujo del programa principal detalla la secuencia lógica de los pasos que realiza la tarjeta controladora Arduino uno-Shield MP3 .

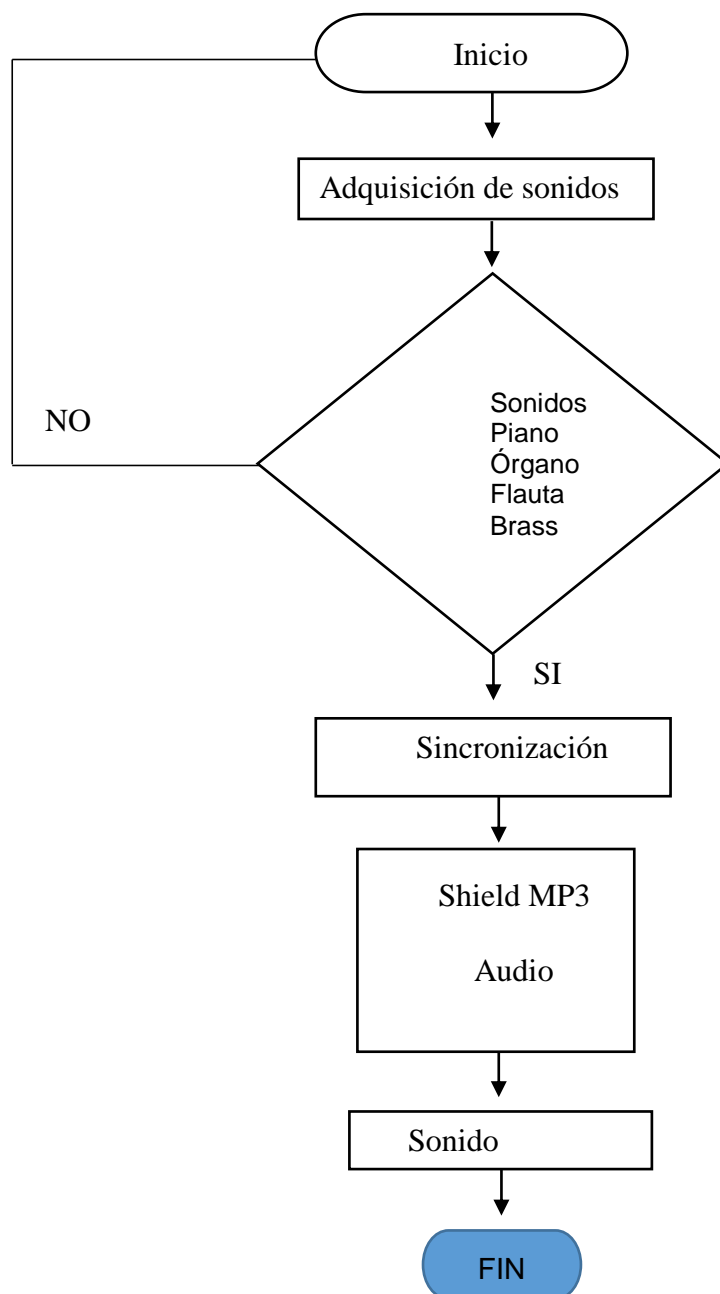


Figura 27. Diagrama de flujo del programa principal.

3.7 Programación de tarjeta Arduino uno

Para realizar el programa principal del sintetizador de sonidos se ejecutaron los siguientes pasos:

- Declarar puertos.
- Declarar variables.
- Void setup.
- Void loop.

3.7.1 Declarar puertos

En esta parte se declararon los puertos para establecer la comunicación serial con el Arduino UNO y Arduino MEGA la creación del puente de comunicación entre el Shield mp3 y Arduino UNO.

```
SoftwareSerial tSerial (3, 4); // RX, TX
/**
 * \breif Macro for the debounced NEXT pin, with pull-up
 */
#define B_NEXT A0

/**
 * \breif Macro for the debounced STOP pin, with pull-up
 */
#define B_STOP A1

/**
 * \breif Macro for the debounced PLAY pin, with pull-up
 */
#define B_PLAY A2

/**
 * \breif Macro for the Debounce Period [milliseconds]
 */
#define BUTTON_DEBOUNCE_PERIOD 20 //ms
```

3.7.2 Declarar variables

En este paso se declara las variables para la selección de los sonidos pregrabados.

```
SdFat sd;
```

```
/**
```

```
 * \brief Object instancing the SFEMP3Shield library.
```

```
 *
```

```
 * principal object for handling all the attributes, members and functions for the library.
```

```
 */
```

```
SFEMP3Shield MP3player;
```

```
/**
```

```
 * \brief Object instancing the Next Button.
```

```
 */
```

```
Bounce b_Next = Bounce ();
```

```
/**
```

```
 * \brief Object instancing the Stop Button library.
```

```
 */
```

```
Bounce b_Stop = Bounce ();
```

```
/**
```

```
 * \brief Object instancing the Play Button library.
```

```
 */
```

```
Bounce b_Play = Bounce ();
```

```
/**
```

```
 * \brief Index of the current track playing.
```

```
 *
```

```
 * Value indicates current playing track, used to populate "x" for playing the
```

```
 * filename of "track00x.mp3" for track000.mp3 through track254.mp3
```

```
 */
```

```
int8_t current_track = 0;
```

```
int tiempo = 150;
```

```
String nota = "";
```

```
String inputString = "";
```

```
String Recepcion="";
```

```
boolean flagRequest=0;
```

```
int option;
```

```
bool banPiano = false;
```

```
bool banOrgano = false;
```

```
bool banBrass = false;
```

```
bool banFlauta = false;
```

3.7.3 Subrutina Principal

En la función void setup, se inicializa para la comunicación serial en donde se muestra la selección de los diferentes sonidos, ya sea seguir a parar, también se muestra el nivel de volumen de cada sonido.

```
void setup () {
```

```
    Serial.begin(38400);
```

```
    tSerial.begin(38400);
```

```
    pinMode (B_NEXT, INPUT_PULLUP);
```

```
    pinMode (B_STOP, INPUT_PULLUP);
```

```
    pinMode (B_PLAY, INPUT_PULLUP);
```

```
    b_Next.attach(B_NEXT);
```

```
    b_Next.interval(BUTTON_DEBOUNCE_PERIOD);
```

```
    b_Stop.attach(B_STOP);
```

```
    b_Stop.interval(BUTTON_DEBOUNCE_PERIOD);
```

```
    b_Play.attach(B_PLAY);
```

```

b_Play.interval(BUTTON_DEBOUNCE_PERIOD);

if(!sd.begin(9, SPI_HALF_SPEED)) sd.initErrorHalt();

if (!sd.chdir("/")) sd.errorHalt("sd.chdir");

MP3player.begin();

MP3player.setVolume(10,10);

Serial.println(F("Looking for Buttons to be depressed..."));

attachInterrupt(digitalPinToInterrupt(22), rep, LOW);

}

```

3.7.4 Subrutina para combinar sonidos

En esta parte de la programación, se configuro que suene un sonido a la vez y que cada tecla tenga su nota musical fija.

```

void loop() {

// Below is only needed if not interrupt driven. Safe to remove if not using.

#ifdef USE_MP3_REFILL_MEANS \

    && ( (USE_MP3_REFILL_MEANS == USE_MP3_SimpleTimer) \

        || (USE_MP3_REFILL_MEANS == USE_MP3_Polled)    )

MP3player.available();

#endif

```

```
if (tSerial.available(>0){  
  
  //leemos la opcion enviada  
  
  option=tSerial.read();  
  
  //Serial.println(option);
```

```
  if(option=='A') {  
    Serial.println("PIANO");  
    delay(1);  
    banPiano = true;  
  
    banOrgano = false;  
  
    banFlauta = false;  
  
    banBrass = false;  
  
  }
```

En esta parte de la programación se seleccionó el sonido que se desea que produzca el teclado, en este caso es el sonido piano.

```
  if(option=='B') {  
    Serial.println("ORGANO");  
  
    delay (1);  
  
    banPiano = false;  
  
    banOrgano = true;  
  
    banFlauta = false;  
  
    banBrass = false;
```

```
}
```

```
if(option=='C') {  
  Serial.println("BRASS");  
  delay (1);
```

```
  banPiano = false;  
  banOrgano = false;  
  banFlauta = false;  
  banBrass = true;  
}
```

```
if(option=='D') {  
  Serial.println("FLAUTA");  
  delay (1);
```

```
  banPiano = false;  
  banOrgano = false;  
  banBrass = false;  
  banFlauta = true;  
}
```

3.8 Programación de la tarjeta Arduino Mega

3.8.1 Diagrama de flujo

El diagrama de flujo del programa principal detalla la secuencia lógica de los pasos que realiza la tarjeta controladora Arduino Mega 2560.



Figura 28. Diagrama de flujo del programa principal.

pasos:

- Declarar puertos.
- Declarar variables.
- Void setup.
- Void loop.

3.8.2 Declarar puertos

En esta parte se declaró los puertos para establecer la comunicación entre el teclado físico y Arduino MEGA.

```
#define d6 22
```

```
#define d7 24
```

```
#define d8 26
```

```
#define d1 28
```

```
#define d2 30
```

```
#define d3 32
```

```
#define d4 34
```

```
#define d5 36
```

```
#define dA 23
```

```
#define dB 25
```

```
#define dD 27
```

```
#define dC 29
```

```
#define dE 31
```

```
#define dF 33
```



```
#define dG 35
```

```
#define dH 37
```

```
#define d9 38
```

```
#define d10 40
```

```
#define d11 42
```

```
#define d12 44
```

```
#define d13 46
```

```
#define d14 48
```

```
#define d15 50
```

```
#define d16 52
```

```
#define dI 39
```

```
#define dJ 41
```

```
#define dK 43
```

```
#define dL 45
```

```
#define dM 47
```

```
#define dN 49
```

```
#define dO 51
```

```
#define dP 53
```

3.8.3 Declarar variables

En este paso se declaró las variables para la selección de los sonidos pre grabados a cada una de las teclas y las notas musicales ya sea naturales sostenidos o bemoles.

```
// Declara los objetos a utilizar
```

```
// [página id:0,componente id:1, NombreComponente: "Objeto_ObjName"]
```

```
NexButton bPiano = NexButton(2,18,"bPiano");
```

```
NexButton bOrgano = NexButton(2,19,"bOrgano");
```

```
NexButton bBrass = NexButton(2,20,"bBrass"); //Guardar los datos de la página DATOS
```

```
NexButton bFlauta = NexButton(2,21,"bFlauta"); //Guardar los datos de la pagina
```

DATOS VEHICULO (selección de notas musicales)

```
//PAGINA 3 FISICO - BOTONES
```

```
NexButton bPianoF = NexButton(3,1,"bPianoF");
```

```
NexButton bOrganoF = NexButton(3,2,"bOrganoF");
```

```
NexButton bBrassF = NexButton(3,3,"bBrassF"); //Guardar los datos de la página DATOS
```

```
NexButton bFlautaF = NexButton(3,4,"bFlautaF"); //Guardar los datos de la página DATOS
```

VEHICULO

```
//TECLAS
```

```
NexButton bDO = NexButton(2,1,"DO");
```

```
NexButton bDOS = NexButton(2,11,"DOS");
```

```
NexButton bRE = NexButton(2,2,"RE"); //Guardar los datos de la página DATOS
```

```
NexButton bRES = NexButton(2,12,"RES"); //Guardar los datos de la página DATOS
```

VEHICULO

```
NexButton bMI = NexButton(2,3,"MI");
```

```
NexButton bFA = NexButton(2,4,"FA");
```

```
NexButton bFAS = NexButton(2,13,"FAS"); //Guardar los datos de la página DATOS
```

```
NexButton bSOL = NexButton(2,5,"SOL"); //Guardar los datos de la página DATOS
```

VEHICULO

```
NexButton bSOLS = NexButton(2,14,"SOLS");
```

```
NexButton bLA = NexButton(2,6,"LA");
```

```
NexButton bLAS = NexButton(2,15,"LAS"); //Guardar los datos de la página DATOS
```

```
NexButton bSI = NexButton(2,7,"SI"); //Guardar los datos de la página DATOS VEHICULO
```

```
NexButton bdo = NexButton(2,8,"do");
```

```
NexButton bdos = NexButton(2,16,"dos");
```

```
NexButton bre = NexButton(2,9,"re"); //Guardar los datos de la página DATOS
```

```
NexButton bres = NexButton(2,17,"res"); //Guardar los datos de la página DATOS
```

VEHICULO

```
NexButton bmi = NexButton(2,10,"mi"); //Guardar los datos de la página DATOS
```

VEHICULO

```
//
```

```
boolean banPiano = LOW;
```

```
boolean banOrgano = LOW;
```

```
boolean banBrass = LOW;
```

```
boolean banFlauta = LOW;
```

```
boolean banPianoAnterior = LOW;
```

```
boolean banOrganoAnterior = LOW;
```

```
boolean banBrassAnterior = LOW;
```

```
boolean banFlautaAnterior = LOW;
```

```
boolean banPianoF = LOW;
```

```
boolean banOrganoF = LOW;
```

```
boolean banBrassF = LOW;
```

```
boolean banFlautaF = LOW;
```

```
boolean banPianoAnteriorF = LOW;
```

```
boolean banOrganoAnteriorF = LOW;
```

```
boolean banBrassAnteriorF = LOW;
```

```
boolean banFlautaAnteriorF = LOW;
```

```
// Registro al objeto Dual State Button a la lista de eventosTouch.
```

```
NexTouch *nex_listen_list [] =
```

```
{
```

```
    &bPianoF,  
    &bOrganoF,  
    &bBrassF,  
    &bFlautaF,  
    &bPiano,  
    &bOrgano,  
    &bBrass,  
    &bFlauta,
```

```

&bDO,
&bDOS,
&bRE,
&bRES,
&bMI,
&bFA,
&bFAS,
&bSOL,
&bSOLS,
&bLA,
&bLAS,
&bSI,
&bdo,
&bdos,
&bre,
&bres,
&bmi,

```

3.8.4 Subrutina principal

En la función void setup, se inicializo para la comunicación serial en donde se muestra la selección de los diferentes sonidos en la pantalla Nextion.

```
void setup(void)
```

```
{
```

```
    nexInit(); //Inicializo la pantalla Nextion se comunica con el Serial2
```

```
    //nexSerial.begin(115200);
```

```
    bPiano.attachPop(funcionPiano, &bPiano);//
```

```
    bOrgano.attachPop(funcionOrgano, &bOrgano);//
```

```
    bBrass.attachPop(funcionBrass, &bBrass);//
```

```
    bFlauta.attachPop(funcionFlauta, &bFlauta);//
```

```
    bPianoF.attachPop(funcionPiano, &bPianoF);//
```

```
bOrganoF.attachPop(funcionOrgano, &bOrganoF);//
```

```
bBrassF.attachPop(funcionBrass, &bBrassF);//
```

```
bFlautaF.attachPop(funcionFlauta, &bFlautaF);//
```

```
bDO.attachPop(funcionDO, &bDO);//
```

```
bDOS.attachPop(funcionDOS, &bDOS);//
```

```
bRE.attachPop(funcionRE, &bRE);//
```

```
bRES.attachPop(funcionRES, &bRES);//
```

```
bMI.attachPop(funcionMI, &bMI);//
```

```
//bMIS.attachPop(funcionMIS, &bMIS);//
```

```
bFA.attachPop(funcionFA, &bFA);//
```

```
bFAS.attachPop(funcionFAS, &bFAS);//
```

```
bSOL.attachPop(funcionSOL, &bSOL);//
```

```
bSOLS.attachPop(funcionSOLS, &bSOLS);//
```

```
bLA.attachPop(funcionLA, &bLA);//
```

```
bLAS.attachPop(funcionLAS, &bLAS);//
```

```
bSI.attachPop(funcionSI, &bSI);//
```

```
bdo.attachPop(funciondo, &bdo);//
```

```
bdos.attachPop(funciondos, &bdos);//
```

```
bre.attachPop(funcionre, &bre);//
```

```
bres.attachPop(funcionres, &bres);//
```

```
bmi.attachPop(funcionmi, &bmi);//
```

```
// Serial.begin(9600);  
  
// Serial1.begin(19200); // Initialize SoftwareSerial  
  
Serial3.begin(38400);  
digitalWrite(13,LOW);  
  
//           TECLADO FISICO  
  
pinMode(d1,OUTPUT);  
pinMode(d2,OUTPUT);  
pinMode(d3,OUTPUT);  
pinMode(d4,OUTPUT);  
pinMode(d5,OUTPUT);  
pinMode(d6,OUTPUT);  
pinMode(d7,OUTPUT);  
pinMode(d8,OUTPUT);
```

3.8.5 Subrutina para los sonidos graves y agudos

En esta parte de la programación, configuramos que las teclas tengan el sonido adecuado y se va realizando escala por escala y nota por nota. Siguiendo la escala musical de forma aguda (HIGH) y de forma grave (LOW).

```
void loop(void)  
{  
    nexLoop(nex_listen_list);
```

```
unsigned long currentMillis = millis();
```

```
if (currentMillis - previousMillis >= interval)
```

```
{
```

```
    previousMillis = currentMillis;
```

```
    //Enviar dato a TFT
```

```
}
```

```
////////////////////////////////////
```

```
digitalWrite(d5,HIGH);
```

```
digitalWrite(d4,LOW);
```

```
digitalWrite(d3,LOW);
```

```
digitalWrite(d2,LOW);
```

```
digitalWrite(d1,LOW);
```

```
digitalWrite(d6,LOW);
```

```
digitalWrite(d7,LOW);
```

```
digitalWrite(d8,LOW);
```

}
d es la nota musical do, en este caso la quinta tela del sintetizador es la nota más grave y las demás son agudos

```
if(digitalRead(dH) == HIGH){
```

```
    Serial3.println("E"); //GS3
```

```
    delay(retardo);
```

```
}
```



```
digitalWrite(d5,LOW);  
digitalWrite(d4,HIGH);  
digitalWrite(d3,LOW);  
digitalWrite(d6,LOW);  
digitalWrite(d7,LOW);  
digitalWrite(d8,LOW);  
digitalWrite(d1,LOW);  
digitalWrite(d2,LOW);  
  
if(digitalRead(dH) == HIGH){  
  Serial3.println("F"); //A3  
  delay(retardo);  
  }  
digitalWrite(d8,LOW);  
digitalWrite(d7,LOW);  
digitalWrite(d6,LOW);  
digitalWrite(d1,LOW);  
digitalWrite(d2,LOW);  
digitalWrite(d5,LOW);  
digitalWrite(d4,LOW);  
digitalWrite(d3,HIGH);  
  
if(digitalRead(dH) == HIGH){
```

```
Serial3.println("G"); //AS3

delay(retardo);

}

////////////////////////////////////

digitalWrite(d2,HIGH);

digitalWrite(d1,LOW);

digitalWrite(d8,LOW);

digitalWrite(d7,LOW);

digitalWrite(d6,LOW);

digitalWrite(d5,LOW);

digitalWrite(d3,LOW);

digitalWrite(d4,LOW);

if(digitalRead(dH) == HIGH){

Serial3.println("H"); //B3

delay(retardo);

}

digitalWrite(d2,LOW);

digitalWrite(d1,HIGH);

digitalWrite(d8,LOW);

digitalWrite(d7,LOW);
```

```
digitalWrite(d6,LOW);  
digitalWrite(d3,LOW);  
digitalWrite(d4,LOW);  
digitalWrite(d5,LOW);  
  
if(digitalRead(dH) == HIGH){  
  Serial3.println("I"); //C2  
  delay(retardo);  
}  
  
//GRUPO 4  
if(digitalRead(dE) == HIGH){  
  Serial3.println("Q"); //GS2  
  delay(retardo);  
}  
  
//GRUPO 6  
if(digitalRead(dC) == HIGH){  
  Serial3.println("Y"); //E1  
  delay(retardo);  
}  
  
digitalWrite(d2,LOW);  
digitalWrite(d1,LOW);  
digitalWrite(d8,HIGH);  
digitalWrite(d7,LOW);
```

```
digitalWrite(d6,LOW);
```

```
digitalWrite(d3,LOW);
```

```
digitalWrite(d4,LOW);
```

```
digitalWrite(d5,LOW);
```

```
if(digitalRead(dH) == HIGH){
```

```
  Serial3.println("J"); //CS2
```

```
  delay(retardo);
```

```
}
```

```
//GRUPO 4
```

```
if(digitalRead(dE) == HIGH){
```

```
  Serial3.println("R"); //A2
```

```
  delay(retardo);
```

```
}
```

```
//GRUPO 6
```

```
if(digitalRead(dC) == HIGH){
```

```
  Serial3.println("Z"); //F1
```

```
  delay(retardo);
```

```
}
```

```
digitalWrite(d2,LOW);
```

```
digitalWrite(d1,LOW);
```

```
digitalWrite(d8,LOW);
```

```
digitalWrite(d7,HIGH);  
  
digitalWrite(d6,LOW);  
  
digitalWrite(d3,LOW);  
  
digitalWrite(d4,LOW);  
  
digitalWrite(d5,LOW);  
  
  
if(digitalRead(dH) == HIGH){  
  Serial3.println("K"); //D2  
  delay(retardo);  
}  
  
//GRUPO 4  
  
if(digitalRead(dE) == HIGH){  
  Serial3.println("S"); //AS2  
  delay(retardo);  
}  
  
//GRUPO 6  
  
if(digitalRead(dC) == HIGH){  
  Serial3.println("1"); //FS1  
  delay(retardo);  
}  
  
  
digitalWrite(d2,LOW);  
  
digitalWrite(d1,LOW);
```

```
digitalWrite(d8,LOW);  
digitalWrite(d7,LOW);  
digitalWrite(d6,HIGH);  
digitalWrite(d3,LOW);  
digitalWrite(d4,LOW);  
digitalWrite(d5,LOW);
```

```
if(digitalRead(dH) == HIGH){  
  Serial3.println("L"); //DS2  
  delay(retardo);  
}
```

```
//GRUPO 4
```

```
if(digitalRead(dE) == HIGH){  
  Serial3.println("T"); //B2  
  delay(retardo);  
}
```

```
//GRUPO 6
```

```
if(digitalRead(dC) == HIGH){  
  Serial3.println("2"); //G1  
  delay(retardo);  
}
```

```
////////////////////////////////////
```

```
digitalWrite(d5,HIGH);
```

```
digitalWrite(d4,LOW);
```

```
digitalWrite(d3,LOW);
```

```
digitalWrite(d2,LOW);
```

```
digitalWrite(d1,LOW);
```

```
digitalWrite(d6,LOW);
```

```
digitalWrite(d7,LOW);
```

```
digitalWrite(d8,LOW);
```

CAPÍTULO IV

CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

- La implementación del control de un sintetizador mediante Arduino pantalla táctil y un shield MP3 en software libre y la tecnología NEXTION; nos permite tener acceso a una gran cantidad de características en diferentes aplicaciones con diferentes tipos de programaciones que serán de utilidad para la formación de los estudiantes de la Unidad de Gestión de Tecnologías.
- El control del del sintetizador se realizó a través de la comunicación serie entre el Arduino Mega y la pantalla táctil Nextion, con lo cual se utilizó los puertos de transmisión T(x) y de recepción R(x) d los dos dispositivos mencionados.
- El almacenamiento de sonidos se realizó mediante el formato Mp3 utilizando un Arduino Uno y un Shield de audio, generando los tracks o sonidos guardados, en la cual cada tecla programada producirá un sonido ya sea grave o agudo de los diferentes instrumentos musicales pregrabados.
- Para la producción de sonidos se realizó mediante un teclado de 5 octavas, mezclando varios sonidos ya programados y cada tecla con su nota musical.

4.2 Recomendaciones

- Antes de la manipulación de la pantalla táctil, es muy importante revisar las características técnicas, para que de esa manera evitemos fallas en la programación de las diferentes páginas.
- Para la correcta carga de la pantalla táctil se debe conectar los pines de transmisión, de recepción y de alimentación a una tarjeta Arduino ya que la pantalla táctil tiene requiere voltios DC y requiere una comunicación serial entre el software y la pantalla.
- En caso de que los sonidos pregrabados no se reproduzcan al presiona una tecla se deberá revisar el formato y el nombre de los audios, ya que el Shield Mp3 solo reconoce formato MP3 y el nombre de track.

REFERENCIAS BIBLIOGRÁFICAS

- Anónimo. (s.f.). *Sparkfund*. Recuperado el 13 de Septiembre de 2019, de <http://www.techmake.com/00181.html>
- García , V. (21 de noviembre de 2010). *EPA*. Recuperado el 13 de Septiembre de 2019, de
ELECTRÓNICA PRÁCTICA APLICDA:
<https://www.diarioelectronicohoy.com/blog/sintetizador-de-musica>
- Gómez Enrique, & Gómez, E. (17 de Octubre de 2017). *Rinconingenieril*. Recuperado el 10 de Junio de 2019, de Pantalla Nextion de Itead y Arduino:
<https://www.rinconingenieril.es/pantalla-nextion-de-itead-y-arduino/>
- Hidalgo Guacho, W. P. (23 de Marzo de 2015). *DSPACE ESPOCH*. Recuperado el 12 de Junio de 2019, de <http://dspace.espoch.edu.ec/handle/123456789/3790>
- Hormazábal Azúa, R. A. (22 de Noviembre de 2017). Interpretación musical en piano Software de aprendizaje de escalas. (M. Muso, Recopilador) Valparaíso, Chile. Recuperado el 13 de Agosto de 2019, de Informe Final del proyecto para la obtención de título de Ingeniero de ejecución en informática.
- Ismael, T. (Diciembre de 2015). *UPS*. Recuperado el 15 de Septiembre de 14, de UNIVERSIDAD POLITÉCNICA SALESIANA:
http://repositorio.uta.edu.ec/bitstream/123456789/19381/1/Tesis_t1093ec.pdf
- Jadiaz. (21 de Enero de 2016). *Mi arduino*. Recuperado el 10 de Agosto de 2019, de ARDUINO:
<http://www.iescamp.es/miarduino/2016/01/21/placa-arduino-uno/>

Llumiquina Leonardo. (1 de Septiembre de 2017). *Repositorio ESPE*. Recuperado el 22 de Junio de 2019, de <http://repositorio.espe.edu.ec/jspui/bitstream/21000/13460/1/T-ESPE-057370.pdf>

Manzano Yupa, H. M., & Tapia Manzano, C. H. (Octubre de 2013). *Universidad Politécnica Salesiana sede Guayaquil*. Recuperado el 14 de Agosto de 2019, de <https://techmake.com/00181.html>

Martinez Zorrilla, D. (2008). *OSJ*. Recuperado el 12 de Agosto de 2019, de OPEN SCIENCE JOURNAL: https://www.researchgate.net/profile/David_Martinez-Zorrilla/publication/260403799_LOS_SINTETIZADORES_Una_breve_introduccion/links/54b8f3480cf2c27adc490a42.pdf

NextionHMI, E. H. (2017). *NEXTION*. Recuperado el 14 de Julio de 2019, de NEXTION: <http://www.nextionec.com/pantallas-nextion.html>

Oña Chipuxi, R. Y. (Abril de 2015). *dspaceUPS*. Recuperado el 12 de Septiembre de 2019, de <https://dspace.ups.edu.ec/bitstream/123456789/10180/1/UPS%20-%20ST001827.pdf>

Pillajo Guitiérrez, V. E. (Agosto de 2018). Recuperado el 13 de Septiembre de 2019, de <http://157.100.241.244/bitstream/47000/1649/1/UISRAEL-EC-ELDT-378.242-2018-057.pdf>

Torres Valverde, E. I. (Diciembre de 2015). *Repositorio UTA*. Recuperado el 14 de Septiembre de 2019, de <file:///C:/Users/MI%20EQUIPO/Downloads/3749-12081-1-PB.pdf>

Veloso Cristian. (19 de Junio de 2018). *Etools*. Recuperado el 23 de Agosto de 2019, de Etools: <https://www.electrontools.com/Home/WP/2018/06/19/arduino-mega-2560-caracteristicas/>

ANEXOS



ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA
CARRERA DE TECNOLOGÍA EN ELECTRÓNICA MENCIÓN
INSTRUMENTACIÓN Y AVIÓNICA

CERTIFICACIÓN

Se certifica que la presente monografía fue desarrollada por el señor **Muso Tandalla Angel Michael** bajo mi supervisión.


En la ciudad de Latacunga a los 3 días del mes de enero 2020

Aprobado por:





ING. PABLO XAVIER PILATASIG PANCHI
DIRECTOR DE LA CARRERA



ING. ALPÚSIG CUICHÁN, SILVIA EMPERATRIZ
DIRECTOR DEL PROYECTO





ABG. PLAZA CARRILLO SARITA JOHANA
SECRETARIA ACADEMICA