CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS DEL INSTITUTO POLITÉCNICO NACIONAL

UNIDAD ZACATENCO

DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

SECCIÓN DE BIOELECTRÓNICA

Sistema de simulación del pulso arterial radial en tres modalidades para enseñanza de acuerdo con la técnica de la medicina tradicional china

Tesis que presenta

Janette Arminda Magaña Cortes

Para obtener el Grado de

Maestra en Ciencias

en la Especialidad de

Ingeniería Eléctrica

Director de la Tesis de la Tesis: Dr. Pablo Rogelio Hernández Rodríguez

Ciudad de México Noviembre 2022

Agradecimientos

Al Consejo Nacional de Ciencia y Tecnología (CONACYT) por el apoyo económico brindado durante la realización del presente proyecto.

Al Dr. Pablo Rogelio Hernández Rodríguez quien me apoyo y de quien es la idea de este trabajo, por sus consejos y orientación, por facilitarme la información que dio pie al desarrollo de este trabajo y sobre todo por su paciencia.

Al Ing. Eladio Cardiel Pérez por el apoyo, el tiempo dedicado por sus consejos, enseñanzas y por el crecimiento tanto académico como personal que me brindo.

A mi madre Lucia Cortes Carrillo por su amor, dedicación y paciencia con la que cada día se preocupaba por mi avance y desarrollo como ser humano, por ser la principal promotora de mis sueños, por cada día confiar y creer en mí y en mis expectativas, por no rendirse nunca y enseñarme a siempre salir adelante.

A mis hermanos Rigel Magaña y Sergio Magaña por apoyarme, darme sus consejos, por motivarme y siempre creer en mí, por estar en los momentos más importantes de mi vida,

A mi mejor amigo Gustavo Bernabe por estar presente en este gran proyecto, por alentarme a iniciar este ciclo de mi vida y siempre apoyarme para que no me rindiera.

A mis compañeros de maestría que me apoyaron brindándome otras visiones del proyecto y estar presentes en lo necesario.

Al personal administrativo del departamento y la sección por su atención para realizar los trámites necesarios durante estos años.

Gracias a la vida por este triunfo, porque cada día me demuestra lo hermosa que es la vida y lo justa que puede llegar a ser.

Finalmente a mis profesores de las etapas académicas anteriores, porque gracias a ellos pude formar parte de este centro tan prestigioso, CINVESTAV.

Contenido

Índice de Figuras V
Índice de Tablas VI
Resumen VII
Abstract VIII
Capítulo 1. Introducción 1
1.1 Planteamiento Del Problema 1
1.2 Objetivo General 2
1.2.1 Objetivos Específicos 2
1.3 Estructura de la Tesis 2
Capítulo 2. Antecedentes y Estado del Arte 4
2.1 Antecedentes 4
2.1.1 La Presión Arterial 4
2.1.2 Onda de Presión Arterial en Tiempo 5
2.1.3 Onda de Presión Arterial en Frecuencia 6
2.1.4 Efecto Windkessel 7
2.1.5 La Onda de Presión Aórtica 8
2.1.6 Onda de Presión Radial 8
2.1.7 Pulso Radial como Señal de Presión 10
2.1.8 Medicina Tradicional China 10
2.1.9 Palpación de Pulso Radial 12
2.2 Estado del Arte 12
Capítulo 3. Desarrollo 14
3.1 Digitalización de la Señal de Pulso 14
3.1.1 Patrones de onda de la señal de presión del pulso 16
3.1.2 Criterios de cuantificación 16
3.2 Tarjeta de desarrollo STM32F407VGT6Discovery 18
3.2.1 *Timer* 19
3.2.2 PWM 19
3.2.3 DAC 20
3.2.4 SPI 20
3.3 Motor Lineal de Bobina de Voz (LVCM, por sus siglas en inglés) 21
3.4 Bocina 22

3.5 Display Gráfico 22
3.6 Programación 24
3.6.1 Entorno de Programación 24
3.6.2 Modalidad de Presión del pulso, utilizando el Motor Lineal 27
3.6.3 Modalidad audible del Pulso 28
3.6.4 Modalidad Visual en el Display Gráfico 29
3.7 Circuitos complementarios 31
3.7.1 Puente H 32
3.7.2 Amplificador de Potencia 32
3.7.3 Recepción de audio 33
3.7.4. Interfaz SPI a Pantalla 34
Capítulo 4. Pruebas 35
4.1 Señal de pulso Digitalizada 35
4.1.1 Análisis espectral de la cadena de datos 37
4.2 Simulador de la Señal de Presión 39
4.2.1 Determinación del ciclo de trabajo del PWM 39
4.2.2 El módulo puente H 43
4.2.3 Prototipo para palpación 43
4.3 Acondicionamiento de la señal de pulso como audio 45
4.3.1. Amplificador de potencia 45
4.3.2. Acoplamiento bocina-micrófono 45
4.3.3 Reproducción de señal de pulso en audífonos 45
4.4 Presentación Gráfica 46
Capítulo 5. Discusión 49
Capítulo 6. Conclusiones y Perspectivas 50
6.1 Conclusiones 50
6.2 Perspectivas 51
Referencias 52
Anexos 53
Anexo 1: Gráficas con la aplicación *WebPlotDigitizer* 53
Anexo 2: Cada de datos empleada para la programación de pulso radial 54
Anexo 3: Hoja de datos motor lineal de voz LVCM-010-013-01 58
Anexo 4: Código empleado en funcionamiento de PWM 58

Anexo 5: Código empleando en funcionamiento de DAC 63
Anexo 6: Librería ST7735.h 68
Anexo 7: Librería GFX_FUNCTIONS.h 71
Anexo 8: Código empleado en funcionamiento de Display 72

Índice de Figuras

Fig. 1: Proceso de regulación de la presión arterial [1]. 4
Fig. 2: Relación entre la presión arterial y el riesgo de mortalidad cardiovascular [1]. 5
Fig. 3: Onda de presión arterial típica, se debe poder analizar en orden de aparición el ascenso sistólico, la presión arterial sistólica, el descenso sistólico, punto dícroto, descenso diastólico y la presión arterial diastólica, previo a un nuevo ciclo cardiaco [3]. 6
Fig. 4: Efecto Windkessel, representando el efecto de las propiedades viscoelásticas [5]. 7
Fig. 5: La Onda de Presión Aórtica de una presión de pulso; se observan las fases de sístole y diástole [5]. 8
Fig. 6: Puntos de palpación de pulsos arteriales principales [8]. 9
Fig. 7: Onda de presión arterial radial correspondiente a una onda sana [5]. 10
Fig. 8: Se presentan la relación entre el pulso y el estado de los órganos internos correspondientes a los puntos de referencia chinos en la muñeca [12]. 11
Fig. 9: Diagrama a bloques de solución propuesta. Se presenta la señal por PWM para el tacto y la señal por DAC para el audio y display. 14
Fig. 10 Patrones de onda de pulso radial en hipertensos [18]. 16
Fig. 11: Resultado de la graficación de puntos obtenidos con "WebPlotDigitize", los valores fueron ajustados en amplitud y tiempo, para su manejo. 18
Fig. 12: Esquema simplificado del flujo de datos [23]. 21
Fig. 13: Motor lineal de Voz LVCM-010-013-01 de la marca Moticont Inc.,EE. UU empleado en el desarrollo del proyecto [24]. 22
Fig. 14: Sección transversal de altavoz dinámico con partes [25]. 22
Fig. 15: Pantalla ST7735 las conexiones se observan en la parte superior donde se deben cuidar las conexiones indicadas. 23
Fig. 16: Configuración básica de Microcontrolador y pantalla empleada. 24
Fig. 17: Interfaz gráfica de la plataforma de programación STM32CubeIDE. Se destacan en color verde los pines que fueron empleados para la programación del código fuente del proyecto. 25
Fig. 18: Apartado de la interfaz gráfica de la plataforma de programación STM32CubeIDE para la frecuencia de reloj. Como se observa, la tarjeta fue configurada en modo HSE a una frecuencia de reloj de 8 MHz. 26
Fig. 19: Pantalla de actualización de sistema de comunicación entre plataforma y tarjeta de desarrollo, se debe conectar la tarjeta previa a la búsqueda de actualización. 27
Fig. 20: Diagrama de flujo del algoritmo para la programación del PWM. 28
Fig. 21: Diagrama de flujo del algoritmo para la programación del DAC que permite la activación de la bocina. 29
Fig. 22: Diagrama de flujo del algoritmo para la programación de la comunicación con el display ST7735 para la visualización de la onda de pulso arterial radial. 31
Fig. 23: Integración del Puente H para el control de desplazamiento del pistón en motor lineal. 32
Fig. 24: Integración del amplificador de potencia para reproducción del pulso como audio. 32
Fig. 25: Circuito amplificador de potencia basado en el CI LM675. 33
Fig. 26: Circuito para recepción de audio por medio de un micrófono electret, pre-amplificación y un amplificador de audio. 33
Fig. 27: Integración de conexión entre Tarjeta de desarrollo con la pantalla a través del protocolo SPI. 34
Fig. 28: Resultado de la señal entre la duración del ciclo de 15ms en verde y 30ms en amarillo, la apertura de la señal en 30ms permite apreciar con mayor detalle las pendientes. 35

Fig. 29: Se aplicó el algoritmo con el ciclo for para la visualización de la señal; se comparan dos escalas de la base de tiempo del osciloscopio de 1000 y 500ms. En esta última se observan mejor los detalles de la forma de la onda. 37
Fig. 30: Espectros en frecuencia correspondientes a las señales de pulso: a) Normotenso; b) Hipertenso atípico; c) Hipertenso establecido y d) Hipertenso Reciente atípico 39
Fig. 31: Filtro pasa bajas de tercer orden para reconstrucción de la señal modulada. 40
Fig. 32: Ondas de pulso reconstruidas por medio del filtro con base en la modulación del 50%. 41
Fig. 33: Ondas de pulso reconstruidas con base en la modulación del 100%, se observa un aumento considerable en el voltaje y las ondas con mayor detalle en las pendientes. 42
Fig. 34: Modulo Puente H, con formato en conexiones para aplicación en motores. 43
Fig. 35: Ensamble del actuador electromecánico. 44
Fig. 36: Vista Frontal. 44
Fig. 37: Vista Frontal e izquierda con aplicación de presión. 44
Fig. 38: Adaptación de colimador a bocina hasta micrófono. 45
Fig. 39: La señal de audio con los lóbulos conformados por los armónicos y sub-amónicos analizando. 46
Fig. 40: Espectro de la señal de pulso en audífonos. Obsérvese la presencia de armónicos en el rango audible. 46
Fig. 41: Visualización de la onda de pulso en display. 48
Fig. 42: Representación de gráficas de ondas de pulso obtenidas por medio de la aplicación con valores originales. 54

Índice de Tablas

Tabla 1: Configuración seleccionada por pines del núcleo, presentada en la figura 17. 25
Tabla 2: Cadena de datos para programación. 54

Resumen

El presente trabajo consiste en el desarrollo de un simulador del pulso sanguíneo de la arteria radial en tres modalidades complementarias de presentación que permita a estudiantes y médicos, de acuerdo con la práctica de la Medicina Tradicional China, familiarizarse con el pulso, *sintiendo, viendo y escuchando* para asociar la información con diagnósticos médicos. Se propone en este trabajo reproducir la señal del pulso arterial radial mediante un actuador electromecánico, presentar de manera visual la gráfica de la señal del pulso y emplear los armónicos de la señal en conjunto con una bocina para ofrecer información del pulso de presión de forma audible. De esta manera, aquellos estudiantes y especialistas interesados en la medicina tradicional china podrán identificar los parámetros principales de pulso sanguíneo, asociando la información con diversos padecimientos de la salud. Debido a la falta de información de la señal del pulso radial en las bases de datos existentes, se cuantificaron patrones gráficos del pulso radial reportados, los cuales se concatenaron para generar y reproducir las señales de pulso de forma continua. El sistema propuesto en este proyecto se diseñó en torno a la tarjeta de desarrollo STM32F4 Discovery, que permite cuantificar, digitalizar y reproducir la señal del pulso a través de algoritmos implementados en la plataforma de programación del procesador de la tarjeta, y empleando circuitos de acondicionamiento se aplica a los dispositivos de salida: un motor lineal de bobina de voz (LVCM-010-013-01), un reproductor de audio y una pantalla gráfica.

Palabras clave: Pulso Arterial Radial, Flujo Sanguíneo, Medicina Tradicional China, Análisis de Pulso Arterial Radial, Simulador de Señal para Enseñanza.

Abstract

The present work consists of developing a simulator of the blood pulse of the radial artery in three complementary modes of presentation. The objective is to achieve a functional device that allows students and doctors, by the practice of Traditional Chinese Medicine, to become familiar with the pulse, feeling, seeing, and listening to associate the information with medical diagnoses. It is proposed in this work to reproduce the radial arterial pulse signal using an electromechanical actuator, present the pulse signal graph visually, and use the harmonics of the signal to offer information on the pressure pulse audibly. In this way, those students and specialists interested in traditional Chinese medicine will be able to identify the main parameters of the blood pulse, associating the information with various health conditions. Due to the lack of information in existing databases, reported radial pulse graphic patterns were quantified and stitched together to continuously generate and reproduce the pulse signals. The system proposed in this project comprises the STM32F4 Discovery development card to quantify, digitize and reproduce the pulse signal. It was applied to a linear voice coil motor (LVCM-010-013-01), an audio player, and a graphic display through algorithms developed in the development board programming platform and conditioning circuits.

Capítulo 1. Introducción

1.1 Planteamiento Del Problema

Desde épocas remotas se ha empleado la Medicina Tradicional China (MTC, por sus siglas en español) para el diagnóstico de enfermedades, para lo cual, los médicos orientales emplean la observación, la visualización, el olfato, la palpación del pulso y de ser posible la historia clínica del paciente. La palpación del pulso se puede realizar en diferentes partes del cuerpo, pero por las características del flujo sanguíneo que se presenta en la arterial radial, ubicada en la cara anterior y lateral de las muñecas, entre el tendón del músculo flexor radial del carpo y la apófisis estiloides del radio, se suele emplear esta para la toma del pulso para el establecimiento de diagnósticos según la técnica de la MTC.

Un estudiante que pretende emplear la pulsología debe aprender a sentir y medir la onda de pulso, esperando que pueda identificar del orden de 28 patrones de pulso y asociarlos con las condiciones de salud y enfermedad.

En contraste, en occidente la toma de pulso da prioridad a los aspectos medibles del pulso, como frecuencia, ritmo, intensidad, tensión, amplitud y forma. Con esta perspectiva, la medicina occidental considera al pulso como la manifestación de cambios en el volumen de los vasos sanguíneos, lo cual ofrece información general del estado cardiovascular de una persona

Los médicos que pretendan usar el pulso para el establecimiento de diagnósticos deben entrenarse en la práctica de la pulsología por mucho tiempo y aprovechar la experiencia de especialistas, con el conocimiento de que en esta técnica existe una relación estrecha entre lo percibido con el tacto digital y padecimientos corporales.

En este trabajo se desarrolló un dispositivo que ofrece la información del pulso radial en tres modalidades distintas y complementarias: sentir, escuchar y visualizar la señal del pulso radial, con el objetivo de reducir el tiempo de aprendizaje de los

estudiantes y especialistas en el conocimiento de la señal de pulso y su asociación con enfermedades.

1.2 Objetivo General

- Desarrollar un sistema de enseñanza mediante simulación de la señal de la presión del pulso sanguíneo en la arteria radial, mediante palpación, visualización de su forma y la reproducción como audio, de acuerdo con la Medicina Tradicional China.

1.2.1 Objetivos Específicos

1. Revisar los conceptos relacionados con la medición del pulso radial, así como los tipos de pulso que se conocen en la medicina tradicional china.
2. Digitalizar señales del pulso radial reportadas en la literatura con y sin patologías.
3. Desarrollar un programa para generar una señal de pulso de manera continua derivada de la concatenación de patrones digitales de la onda de pulso radial.
4. Construir un sistema generador de señales de presión del pulso, utilizando un actuador electromecánico del tipo de motor lineal de bobina de voz.
5. Implementar un reproductor de audio relacionado con la onda de presión del pulso radial.
6. Presentar la forma de la señal del pulso radial en un display gráfico para su visualización.

1.3 Estructura de la Tesis

La tesis presentada se divide en seis capítulos:

- Capítulo 1: Descripción del problema planteado, así como la propuesta sugerida para su solución. En este capítulo también se presentan los objetivos planteados para el desarrollo del proyecto.

- Capítulo 2: Información del tema como parte del trabajo de investigación. Se habla sobre la generación de presión arterial, así como el proceso de recorrido hasta llegar a la arteria radial. También se encuentra información en relación con la toma del pulso de acuerdo a la medicina tradicional china. Se presenta el estado del arte donde se encuentran trabajos similares a este proyecto que han sido realizados y analizados para que en su caso, sean aprovechadas las contribuciones.
- Capítulo 3: Descripción del diagrama a bloques de la propuesta para el desarrollo del sistema. Se exponen el material y equipo empleado así como los procedimientos y circuitos realizados.
- Capítulo 4: Pruebas realizadas y los resultados obtenidos.
- Capítulo 5: Discusión de los resultados obtenidos.
- Capítulo 6: Conclusiones y perspectivas

Referencias.

Anexos.

Capítulo 2. Antecedentes y Estado del Arte

En el presente capítulo se describen algunos conceptos relacionados con la circulación sanguínea y el proceso por medio del cual se deriva y obtiene el pulso radial. En el apartado 2.2, estado del arte, se presenta una recopilación de información para pronta referencia, que permitirá comprender el tema y trabajos relacionados.

2.1 Antecedentes

Las arterias del cuerpo humano conforman el sistema de la circulación sanguínea y son las encargadas de distribuir la sangre oxigenada desde el corazón hacia los tejidos y órganos. En este proceso se expulsa rítmicamente la sangre desde la cámara del ventrículo izquierdo del corazón y se lleva a los tejidos corporales.

2.1.1 La Presión Arterial

La presión arterial se produce por la fuerza de la sangre al ser empujada contra las paredes de las arterias y depende del gasto cardiaco y la resistencia periférica total (figura 1)

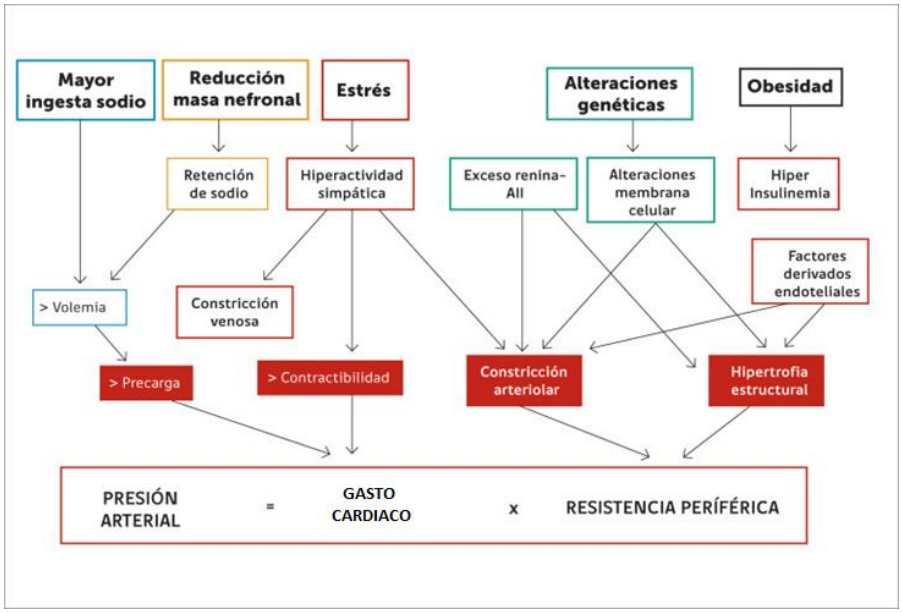


Fig. 1: Proceso de regulación de la presión arterial [1].

La lectura de la presión arterial corresponde a los datos de las presiones sistólica y diastólica. El primer dato, es producido por el movimiento del corazón cuando se contrae y empuja la sangre hacia las arterias y el segundo corresponde a la presión hidrostática presente en los vasos sanguíneos que ocurre cuando el corazón se relaja y se lleva a cabo el llenado de sangre en sus cavidades.

Diversos estudios establecen que la presión arterial es una variable biológica, es decir, una variable que depende de estímulos orgánicos intrínsecos, factores ambientales, físicos, emocionales, además de los períodos de actividad y reposo de cada sujeto, por lo tanto, no existe un punto de corte para definir los valores normales de la presión arterial. La Organización Mundial de la Salud (OMS) considera a los padecimientos relacionados con la presión arterial como el principal factor de riesgo para muertes por enfermedades cardiovasculares.

En la figura 2 se muestra una relación entre la presión arterial y el riesgo cardiovascular, se ha determinado el valor de 115/75mmHg como el de menor riesgo de mortalidad [1]

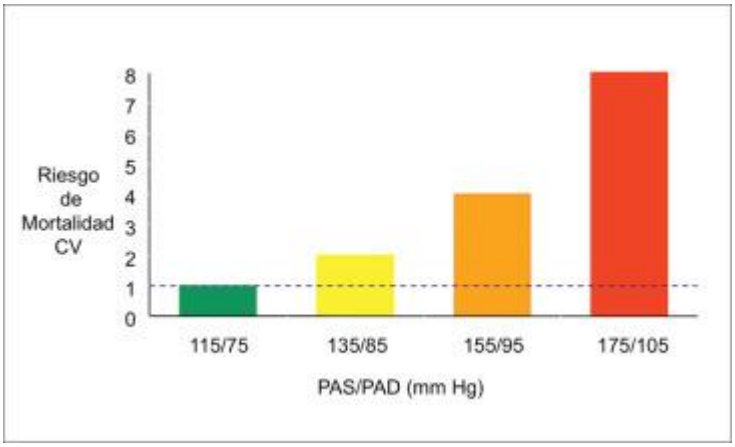


Fig. 2: Relación entre la presión arterial y el riesgo de mortalidad cardiovascular [1].

2.1.2 Onda de Presión Arterial en Tiempo

Los dominios del tiempo y frecuencia han sido empleados para estudiar la forma de la onda de la presión arterial, cada una cuenta con diferentes interpretaciones [2].

Debido a la falta de información para cuantificar el pulso en el dominio de la frecuencia se ha optado por trabajar en el dominio del tiempo.

Estudios de la presión arterial con respecto al tiempo han determinado tres principales parámetros a medir: la profundidad, la anchura y la fuerza [2].

Emplear el dominio del tiempo para la cuantificación del estado del pulso ha ganado relevancia ya que sus parámetros tienen un significado fisiológico. Se ha reportado la asociación entre niveles y formas de la onda de pulso en registros a niveles superficial, medio y profundo desde 1995. Sin embargo no existe un procedimiento estándar de registro por lo que la mayoría de los estudios se centran solamente en la condición del pulso [2].

La forma de una onda de presión arterial generalmente corresponde a la que se muestra en la figura 3. En ella se observan 6 partes a identificar, en mayor o menor medida, a menos que exista una patología presente.

El punto 4 corresponde al punto dícroto, el cual destaca porque se relaciona con el cierre de la válvula aortica, dividiendo así el ciclo cardiaco en sístole y diástole [3].

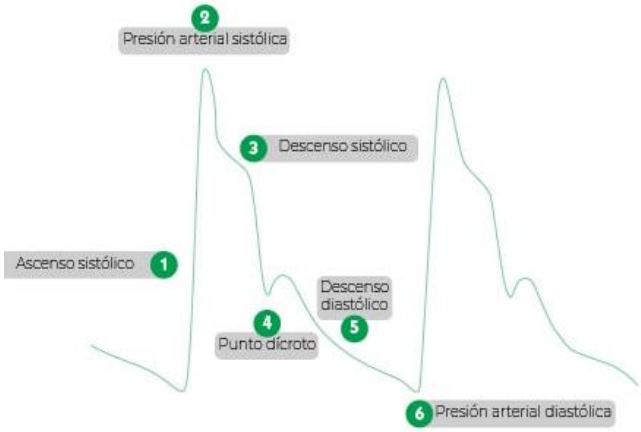


Fig. 3: Onda de presión arterial típica, se debe poder analizar en orden de aparición el ascenso sistólico, la presión arterial sistólica, el descenso sistólico, punto dícroto, descenso diastólico y la presión arterial diastólica, previo a un nuevo ciclo cardiaco [3].

2.1.3 Onda de Presión Arterial en Frecuencia

La frecuencia natural y resonancia de una onda de pulso arterial está compuesta por una onda fundamental entre 1-3Hz y varios armónicos entre 25-30Hz, los cuales son indicadores de información sistémica del estado fisiológico que guardan los órganos corporales [2, 4].

La frecuencia fundamental en una onda de presión arterial sana coincide con la frecuencia de pulso, es decir entre 60 y 120 latidos por minuto aproximadamente y es equivalente a 1Hz y 2Hz respectivamente.

Se ha reportado el análisis en el dominio de la frecuencia para la determinación de las características del pulso; para ello se obtiene la distribución de la energía de la onda de presión arterial. Analizar el espectro de potencia (amplitud y frecuencia) ayuda a determinar sus componentes en armónicos y ha servido para diferenciar enfermedades y relacionarlas con el síndrome y canales de meridianos; existen pocos estudios relacionados a diferenciar las características del pulso [2].

2.1.4 Efecto Windkessel

Cuando se produce una contracción cardiaca, la sangre es expulsada para su recorrido por el sistema arterial, donde las propiedades viscoelásticas de las arterias y el fenómeno de reflexión producen ajustes en flujo transversal y longitudinal que ayudan a mantener un flujo continuo y regular el volumen sistólico que llega hasta los capilares debido al efecto Windkessel figura. 4 [5-7]

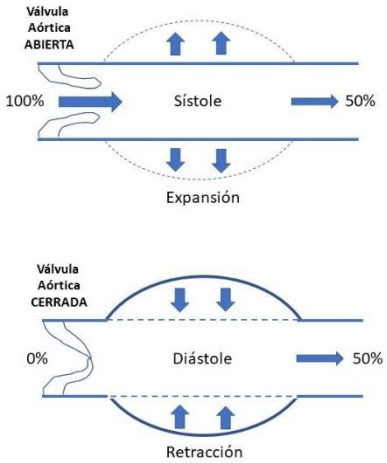


Fig. 4: Efecto Windkessel, representando el efecto de las propiedades viscoelásticas [5].

2.1.5 La Onda de Presión Aórtica

La onda de presión aórtica, se muestra en la figura 5. En esta se muestra como el fenómeno de reflexión sistólica producido por la resistencia periférica en las arterias y arteriolas terminales modifica la morfología en la rama dícrota [6]. También se puede observar una curva de ascenso (Rama Anacrótica) relacionada con el inicio de la presión arterial sistólica y una de descenso (Rama Dícrota) relacionada con la relajación ventricular, el cierre de la válvula aórtica y el establecimiento de la presión diastólica hasta el inicio de una nueva contracción cardíaca [5].

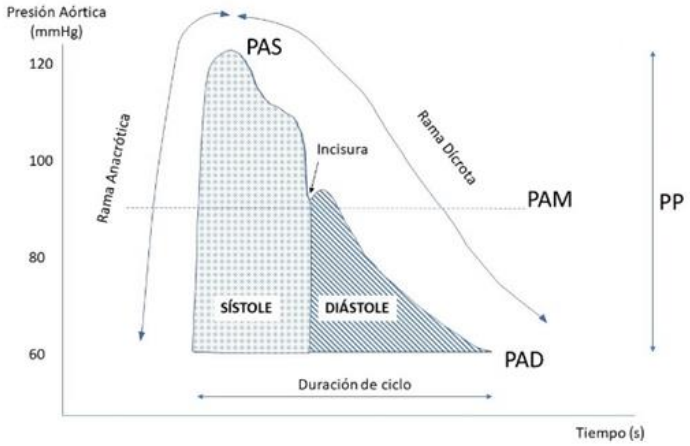


Fig. 5: La Onda de Presión Aórtica de una presión de pulso; se observan las fases de sístole y diástole [5].

2.1.6 Onda de Presión Radial

Durante la eyección sanguínea ventricular en la arteria aorta se genera una onda de presión, sanguínea la cual se propaga a una velocidad de aproximadamente 4 m/s [5] y que se presenta en las arterias de todo el sistema circulatorio. La palpación de esta onda de presión se puede realizar en diferentes puntos corporales, tal como se ilustra en la figura 6. En estudios clínicos, una de las arterias con mayor interés es la radial, ubicada en ambas muñecas, es superficial y de fácil acceso para los médicos; en ella se registra el pulso de manera no invasiva, mediante la palpación con los dedos.

La forma de la onda de presión en los lugares de palpación es diferente a la onda sistólica aórtica por las afectaciones que sufre en su recorrido por los vasos sanguíneos [5, 6]. La diferencia se asocia a los efectos de avance y retroceso del flujo

sanguíneo que originan la presencia de tres picos (figura 7). El pico 1 corresponde a la sístole, mientras que los picos 2 y 3 se refieren a sitios de reflexión en la región de las arterias renales y por debajo de la bifurcación de las arterias iliacas. El valle que se observa corresponde a residuos en la válvula aórtica, mientras que la T4 es el tiempo hasta la muesca dícrota presente en el inicio de la diástole [5].

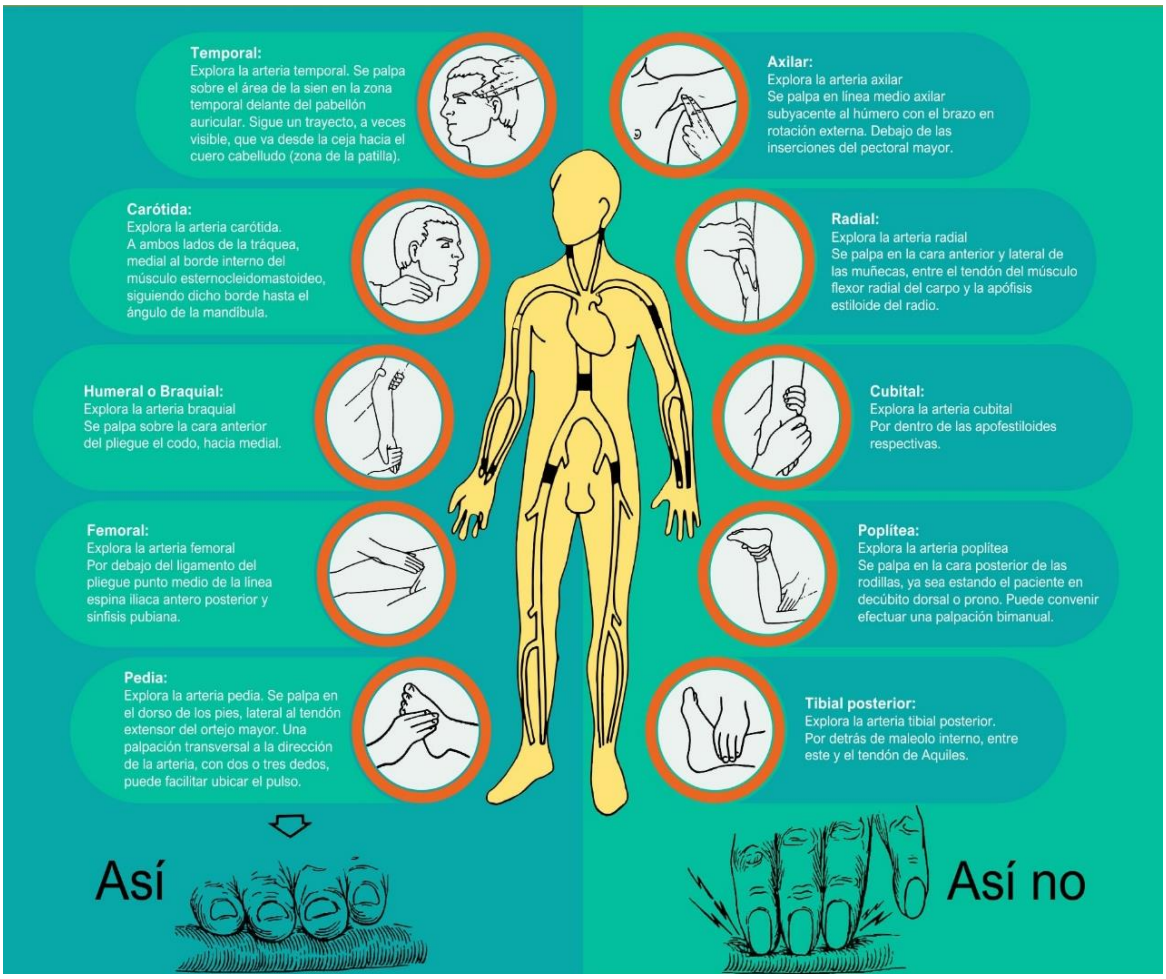


Fig. 6: Puntos de palpación de pulsos arteriales principales [8].

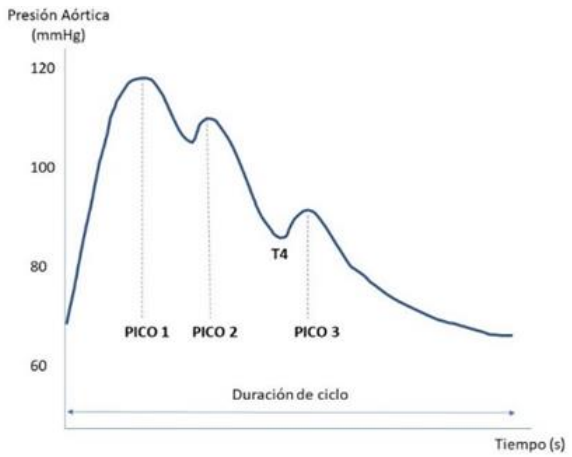


Fig. 7: Onda de presión arterial radial correspondiente a una onda sana [5].

2.1.7 Pulso Radial como Señal de Presión

La onda del pulso radial se puede estudiar directamente, o se puede obtener una aproximación de la onda de presión aórtica a través de un algoritmo matemático, para proporcionar las partes características de la señal del pulso de presión [9].

Debido a la poca disponibilidad de registros para estudiar la señal de la onda del pulso radial, se requiere el desarrollo de sistemas de medida y la conformación de una base de datos que permitan un análisis objetivo de la señal y su relación con la fisiología corporal humana.

2.1.8 Medicina Tradicional China

La Medicina Tradicional China (MTC) considera al ser humano como un micro-cosmos, en el cual se relacionan las leyes naturales con el flujo y la transformación de la energía. El médico de la MTC no diagnostica la enfermedad, sino al enfermo [10, 11].

2.1.8.1 El Pulso

El pulso en la MTC brinda un buen indicador de la naturaleza y el flujo de energía de una persona. Se puede tomar en diferentes lugares, pero comúnmente se toma en la muñeca, donde se divide en tres partes conocidas como: Cun (pulgada), Guan (conexión) y Chi (pie). Estos puntos se encuentran en las muñecas y se relacionan con diferentes órganos internos, tal como se puede ver en la figura 8. Al tomar el pulso, los

médicos tradicionales chinos son capaces de detectar pequeñas diferencias entre los pulsos de la muñeca izquierda y derecha, así como también en la misma mano, distinguiendo así los pulsos superficiales y profundos [12].

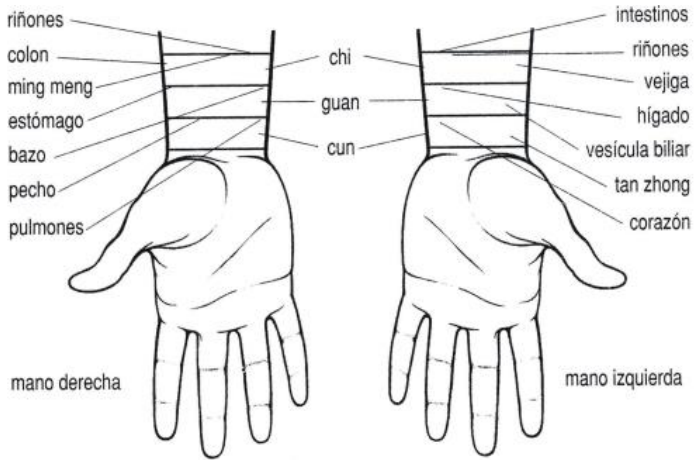


Fig. 8: Se presentan la relación entre el pulso y el estado de los órganos internos correspondientes a los puntos de referencia chinos en la muñeca [12].

Los grandes maestros de la MTC han clasificado los distintos pulsos en veintiocho tipos principales, de los cuales se mencionan seis principales [12, 13]; otras definiciones para la detección de los tipos de pulso puede encontrarse en [2, 14].

a. Pulso flotante: Se palpa fácilmente en la superficie de la piel, la onda es ancha y fuerte.
b. Pulso sumergido: Para palpar este pulso, se debe presionar sobre la muñeca ya que es lento y de onda baja.
c. Pulso lento: El pulso es bajo aproximadamente tres pulsaciones por respiración.
d. Pulso rápido: El pulso es rápido aproximadamente siete pulsaciones por respiración.
e. Pulso débil: Se puede palpar cualquier superficie, sin embargo las pulsaciones serán débiles.
f. Pulso fuerte: Se puede palpar en cualquier superficie, las pulsaciones son fuertes.

2.1.9 Palpación de Pulso Radial

La palpación del pulso es empleada para distinguir entre sus características físicas y morfológicas de la onda. De acuerdo con la MTC el diagnóstico a través del pulso refleja el estado del organismo como un todo por lo que brinda valiosa información de las condiciones de salud de una persona. Para la palpación del pulso radial en la MTC se emplean tres dedos, de manera que el dedo medio derecho (Guan), se coloca junto al esqueleto de la arteria radial, el dedo índice (Cun) y el dedo anular (Chi) se colocan seguidamente del Guan [15].

Durante la palpación del pulso se busca identificar las ondas ascendente y descendente, su amplitud y frecuencia de los latidos. Quien realiza el procedimiento debe identificar si son regulares (existe la misma distancia entre cada uno) o irregulares (Si existe alguna alteración entre cada pulso). También se debe identificar los picos mencionados en la figura 7 para una persona sana y conocer la diferencia que se genera por medio de la respiración.

2.2 Estado del Arte

Desde hace varias décadas la medicina tradicional china ha tratado al pulso como el centro de estudio, en donde se ha asociado la pulsología con la forma de la onda y la medición de los parámetros del pulso. Existen al menos 28 clasificaciones estudiadas para el pulso, las cuales se asocian por la forma o profundidad con algún órgano y la posibilidad de obtener un diagnóstico del estado de salud de un paciente [10, 12, 13]. Esta técnica de diagnóstico ha generado gran interés, sin embargo ya que la medición ha sido subjetiva, la única forma de aprenderla es a través de la experiencia transmitida por los especialistas [15]. Se ha estudiado el pulso tanto en oriente como en occidente, dando como resultado la generación de equipos capaces de medir el pulso como lo es el tonómetro [16] así como dispositivos simuladores de pulso [17].

Se ha reportado la necesidad de sistemas cuantitativos para clasificar los 11 tipos de pulsos trabajados en hospitales de MTC. Existe un trabajo para el diagnóstico cuantitativo basado en redes bayesianas, construyendo un mapeo entre las ondas de pulso y los tipos de pulso [14]. Los resultados muestran que el sistema obtiene

predicciones de tipo de pulso relativamente fiables; este sistema diseñado solo permite la clasificación de los tipos de pulsos sin embargo no permite una enseñanza a los médicos de la MTC.

La hipertensión crónica produce alteraciones en el sistema arterial por lo que el estudio de las ondas de pulso para determinación de enfermedades y evaluación del estado en el sistema circulatorio permite la posibilidad de hallar diferencias entre pacientes hipertensos recientes y crónicos en base al análisis de la onda de pulso radial; en un trabajo reportado en [18], se efectuó la determinación del índice de aumentación radial, que cuantifica la magnitud de las reflexiones de la onda de presión, en 61 pacientes hipertensos de edades comprendidas entre 40 y 60 años, sin patologías agregadas, y se lo comparó con otro estudio similar efectuado sobre un grupo de 68 pacientes normotensos sanos de la misma franja etaria; la toma del pulso fue realizada por medio de la tonometría

Debido a que la frecuencia cardiaca cambia según el entorno, es difícil diagnosticar objetivamente la onda de pulso de una persona [2, 4]. En la región Oriental, se implementó la mejora de un simulador que permite generar una onda de pulso completa y reproducirla por medio de tres motores lineales de bobina de voz [15] este trabajo logró simular una onda de pulso de un paciente sano y por medio de un programa deformar la señal según la necesidad del usuario, se realizaron pruebas con al menos cuatro tipos de ondas diferentes, siendo la onda sana el único patrón que se empleó para la comprobación. Cabe destacar que este simulador se realizó con motores lineales de bobina de voz, sin embargo, solamente cuenta con una onda de pulso de referencia. A diferencia de lo reportado, en esta propuesta de tesis cuenta con cuatro ondas de referencia y plantea ofrecer la información de la señal de pulso en tres modalidades complementarias de enseñanza para médicos y estudiantes.

Capítulo 3. Desarrollo

La realización del proyecto se llevó a cabo siguiendo el procedimiento descrito de manera gráfica en la figura 9. En esta destaca el proceso de adquisición y reproducción de la señal.

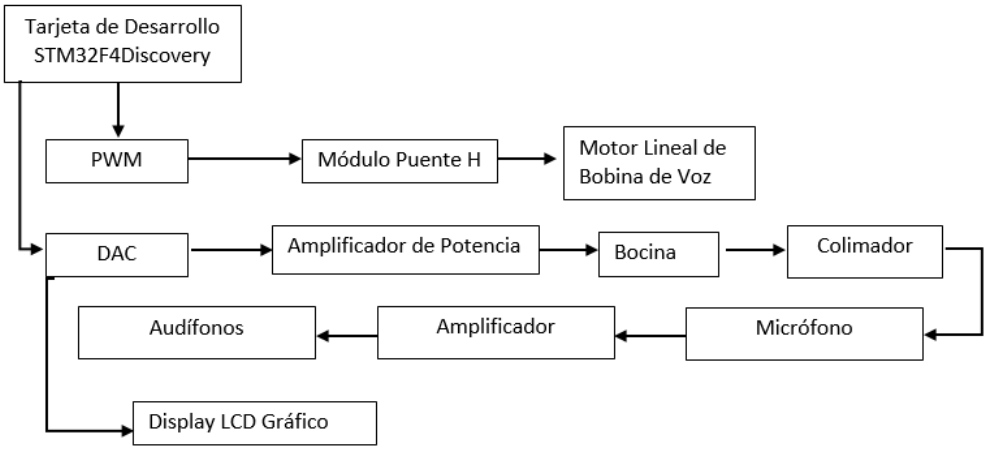
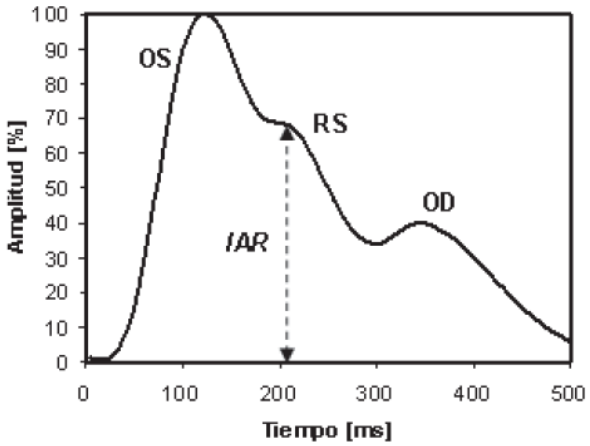


Fig. 9: Diagrama a bloques de solución propuesta. Se presenta la señal por PWM para el tacto y la señal por DAC para el audio y display.

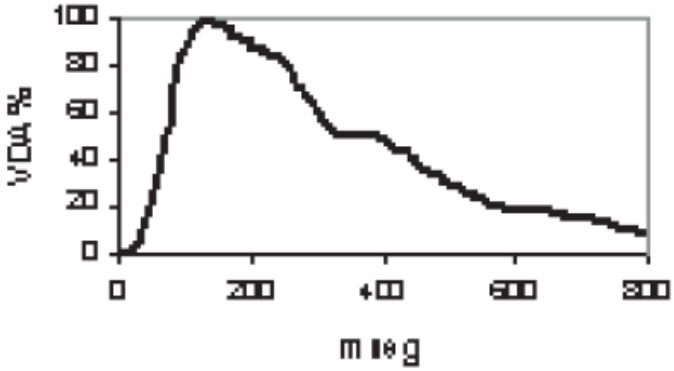
Para el desarrollo del trabajo se realizó la búsqueda de ondas de pulso radial en base de datos existentes. Se revisaron *Zenodo* y *Premilinary Pulse Wave Database* [19, 20] las cuales cuentan con datos de pulso arterial, pero no de la onda de pulso radial necesaria para el desarrollo del proyecto.

3.1 Digitalización de la Señal de Pulso

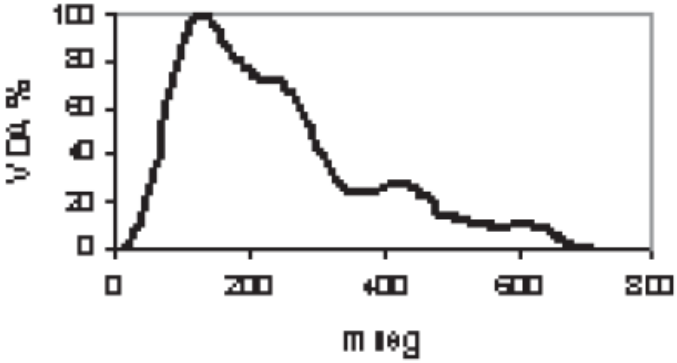
Debido a la falta de información en las bases de datos consultadas [19, 20], se trabajó en la digitalización de patrones de ondas de pulso arterial radial obtenidas en [18] las cuales se muestran en la figura 10 contando con información de 3 tipos de ondas de pulso de hipertensos y una onda de pulso normotenso.



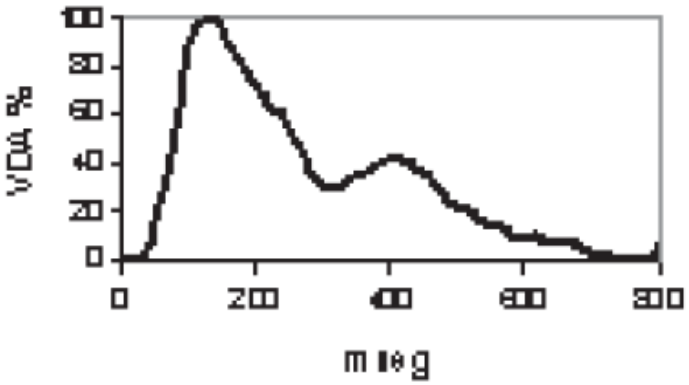
a) Onda de Pulso de un Normotenso, se observa la onda sistólica (OS), la onda diastólica (OD) y la reflexión sistólica (RS).



b) Onda de Pulso de un Hipertenso Establecido.



c) Onda de Pulso de un Hipertenso Atípico.



d) Onda de Pulso de un Hipertenso Reciente Atípico.

Fig. 10 Patrones de onda de pulso radial en hipertensos [18].

3.1.1 Patrones de onda de la señal de presión del pulso

Los patrones de pulso arterial radial fueron digitalizados a partir de imágenes reportadas en la literatura [18]. Los patrones corresponden a sujetos con hipertensión en un rango amplio de edades y sin patologías reportadas. La información de las imágenes fue adquirida por medio de la tonometría la cual fue promediada, filtrada y normalizada para relacionarla con la sístole y diástole.

En la figura 10a se muestra la onda de pulso de un adulto normotenso, la cual se encuentra normalizada en amplitud, donde el valor máximo del 100% equivale a la presión sistólica y el valor mínimo de 0% equivale a la presión diastólica. En la figura 10b la imagen corresponde al pulso promedio de 47 personas hipertensas establecidas y finalmente en la figura10d se presenta la imagen del pulso promedio de 14 personas con hipertensión recientes.

3.1.2 Criterios de cuantificación

La cuantificación de la imagen de la señal de pulso fue realizada por medio de la herramienta de libre acceso *WebPlotDigitize* [21]. Esta herramienta define el número de puntos necesarios para reconstruir la señal (anexo 1) y determina las coordenadas correspondientes a cada punto, como se menciona a continuación:

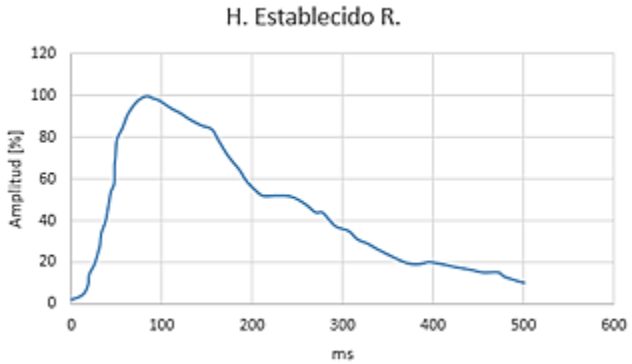
- Onda Típica (Onda de pulso Normotenso) con 74 puntos
- Onda Hipertenso Establecido con 61 puntos
- Onda Hipertenso Atípico con 54 puntos

- Onda Hipertenso Reciciente Atípico con 59 puntos

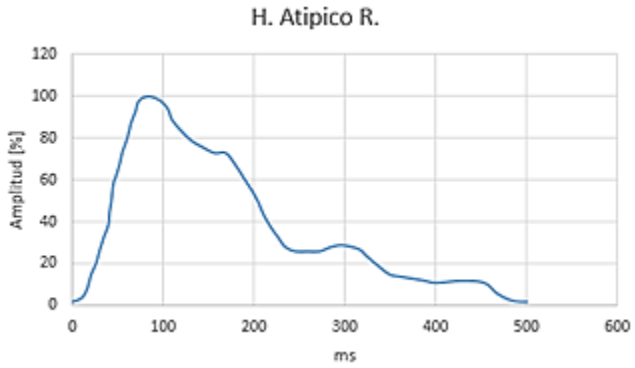
Empleando Excel se efectúa una normalización y ponderación para generar los valores de amplitud y tiempo adecuados al rango de trabajo del sistema digital. Debido a que la aplicación arroja los resultados en valores con decimales, fue necesario realizar una conversión apropiada a números enteros. Además, los valores de tiempo y amplitud fueron normalizados a 500ms y una amplitud máxima de 100%, como se puede observar en la figura 11, de manera que los datos registrados correspondan con 100 como máximo para la presión sistólica y 0 a la presión diastólica. Las cadenas de datos resultantes se presentan en el anexo 2.

Una vez obtenidos estos datos, se graficaron las ondas de pulso quedando como resultado las gráficas presentadas en la figura 11. Estas cadenas de datos fueron utilizadas en la programación de la onda de pulso radial para su aplicación en las tres distintas modalidades propuestas en este proyecto.

a) Onda de Pulso de un Normotenso



b) Onda de Pulso de un Hipertenso Establecido.



c) Onda de Pulso de un Hipertenso Atípico.

d) Onda de Pulso de un Hipertenso Reciente Atípico.

Fig. 11: Resultado de la graficación de puntos obtenidos con “WebPlotDigitize”, los valores fueron ajustados en amplitud y tiempo, para su manejo.

3.2 Tarjeta de desarrollo STM32F407VGT6Discovery

Debido a los requerimientos del proyecto se utilizó la tarjeta de desarrollo STM32F407VGT6Discovery que ofrece las siguientes características:

- Salida PWM
- Poseer salida de DAC
- Contar con protocolos SPI e I^2C

Debido a la necesidad de mostrar la señal de pulso como imagen, se empleó el display gráfico ST7735.

La tarjeta de desarrollo STM32F407VGT6Discovery cuenta con un procesador CortexM-4 FPUcore y puede trabajar hasta 180MHz. Cuenta con 12 salidas PWM con

propósitos de uso en general, dos salidas DAC, modalidades de trabajo de 3V y 5V y protocolos de comunicación I^2C y SPI.

3.2.1 *Timer*

El *timer* es una función interna y forma parte del conjunto esencial de periféricos integrados en los microcontroladores STM32, su aplicación se utiliza para control de motores, realizar retardos y contar eventos. Esta función se empleó en este proyecto para la activación del canal de salida en el modo PWM.

3.2.2 PWM

La modulación por ancho de pulso (*Pulse Width Modulation*, PWM) es una técnica de modulación que genera pulsos de ancho variable para representar la amplitud de una señal de entrada analógica. En este tipo de modulación, el periodo de los pulsos permanecerá fijo, pero la duración del ancho (ciclo útil) se modifica. Consiste en una señal cuadrada con una frecuencia determinada que puede variar su ciclo de trabajo para así poder modificar el valor nominal de la tensión. Para el uso en el modo de comparación de la salida en PWM, se empleó el *timer* activado por el registro TIMX_CR1 y definido por la frecuencia del oscilador externo (HSE).

Para la programación de la salida PWM, se eligió el funcionamiento del *timer* con un conteo ascendente de manera que se cuenta desde el valor 0 hasta el valor del registro de auto recarga TIM_ARR, al cual previamente se le asignó un valor acorde a la frecuencia que se desea generar. Una vez que el contador iguala el valor del registro de captura TIM_CCRx, se produce la señal de valor 0; al llegar a un nuevo valor de la auto recarga, el contador se reinicia y cambia la señal a 1, produciendo la señal de pulso. De esta manera cuando se modifica el TIM_CRR y TIM_ARR, se genera un cambio en el ciclo de trabajo generando una frecuencia de la señal en la salida del PWM.

3.2.3 DAC

El propósito del uso de los convertidores DAC de la tarjeta de desarrollo fue para realizar la conversión de la onda de pulso radial digitalizada a una representación analógica para su aplicación en una modalidad de las propuestas en este proyecto, convirtiendo así el dato digital a un voltaje analógico correspondiente.

El convertidor DAC de 12 bits, incluido en la tarjeta de desarrollo, cuenta con dos canales de salida, los cuales se activan a través de las salidas de actualización de temporizadores específicos. El voltaje de salida analógico está determinado por la Ec. 1 [22], para una resolución de 12 bits alineados a la derecha en cada canal de DAC, se suele alinear a la derecha para trabajar desde el bit LS (menos significativo).

$$DAC_{salida} = V_{REF}\ x\ DOR/4096 \qquad \text{Ec.1}$$

Donde VRef es de 3.3V máximo y DOR es el dato almacenado en el registro de salida.

Se utilizó el oscilador externo HSE para la operación del DAC y se seleccionó la salida DAC_OUT1. Cabe mencionar que se dispone de dos canales de salida que pueden usarse de manera simultánea.

3.2.4 SPI

La interfaz de periféricos en serie (SPI) es un protocolo de comunicación común en dispositivos integrados, permite un manejo transparente de la transferencia de datos entre periféricos y microcontrolador.

Se basa en un par de registros de desplazamiento internos físicamente separados, como se puede observar en la figura 12 el cual maneja un flujo de bits en serie de entrada y salida por medio de los pines MOSI y MISO sincronizados por la señal SCK. La configuración del SPI, dependiendo de si funciona como esclavo o maestro determina qué pin SPI actuara como entrada o salida y si la fuente de reloj SCK es externa o interna. Los datos se cronometran dentro y fuera a través de los pines MISO y MOSI por medio de registros de desplazamiento asociados al área de RAM que proporcionan los buffers Tx y Rx separados para guardas los datos según sea para recepción o transmisión, para una mejor comprensión se sugiere leer [23].

Cuando el periférico está configurado como esclavo, el reloj SCK siempre lo proporciona una fuente externa; cuando está configurado como maestro, el reloj SCK se obtiene internamente del núcleo [23].

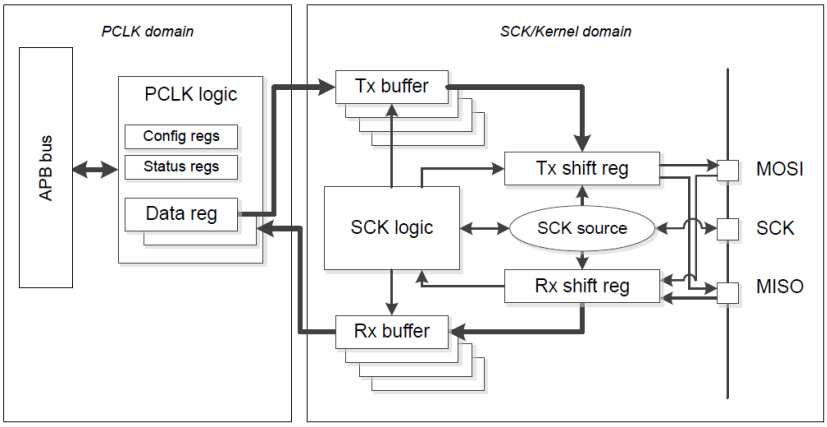


Fig. 12: Esquema simplificado del flujo de datos [23].

El SPI es capaz de funcionar con velocidades más altas que el I^2C con la desventaja de que requiere el uso de más pines. Sin embargo, su capacidad de respuesta lo convierte en una buena opción para el trabajo presentado en este proyecto además se definió el protocolo de comunicación SPI por ser el más apropiado para el modo de operación de la pantalla LCD TFT a color ST7735 de 1.8”.

3.3 Motor Lineal de Bobina de Voz (LVCM, por sus siglas en inglés)

El motor lineal de bobina de voz LVCM-010-013-01 (Moticont Inc.,EE. UU.), es un dispositivo electromagnético conformado por una bobina que actúa como pistón y un imán fijo, figura 13. Fue utilizado como actuador para simular y reproducir la señal de pulso radial. Se empleó la configuración puente H como refuerzo de corriente e inversión de sentido de desplazamiento para permitir al motor desarrollar una fuerza continua de hasta .28N y una potencia máxima de 1.8W, como se indica en la hoja de datos presentada en el anexo 3.

Al circular una corriente eléctrica a través de la bobina se genera un campo magnético el cual interactúa con el campo del imán, atrayendo o repeliendo la bobina en función de la fase del campo generado.

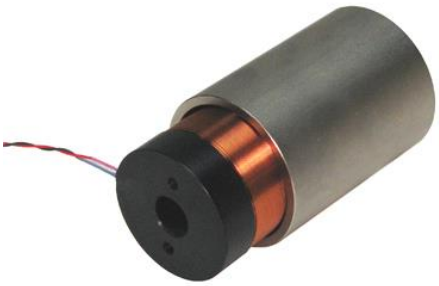


Fig. 13: Motor lineal de Voz LVCM-010-013-01 de la marca Moticont Inc.,EE. UU empleado en el desarrollo del proyecto *[24].*

3.4 Bocina

Con el objetivo de ofrecer la información del pulso radial de manera audible se aprovechó la presencia de los armónicos de la señal de pulso obtenidos mediante análisis espectral, siendo de 4Hz a 25 Hz para la cadena de datos y hasta 4500 Hz para la señal audible. Para estos fines se empleó una bocina del tipo *subwoofer* de 5" de diámetro y 8 ohms de impedancia, cuyo desempeño en bajas frecuencias la hizo adecuada para lograr el objetivo propuesto, figura 14.

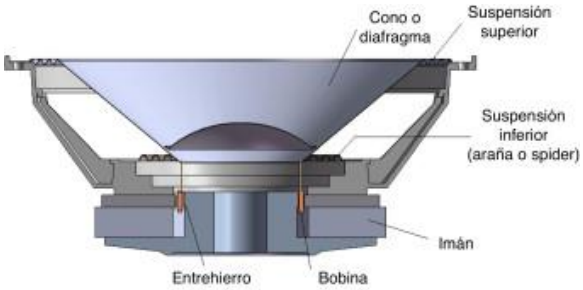


Fig. 14: Sección transversal de altavoz dinámico con partes [25].

3.5 Display Gráfico

Para cubrir la propuesta del proyecto en la modalidad visual, se empleó una pantalla gráfica LCD TFT a color ST7735 la cual cuenta con una resolución 128x128 pixeles figura 15.

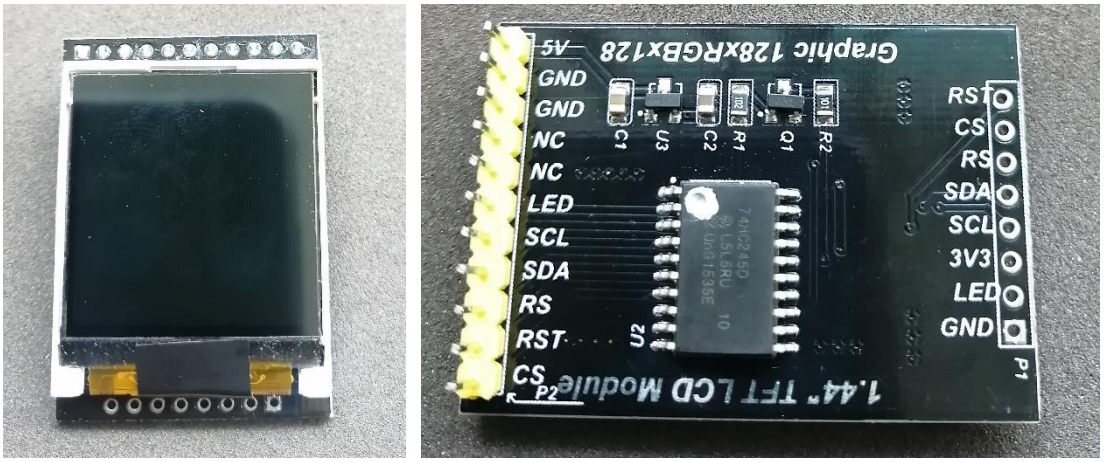


Fig. 15: Pantalla ST7735 las conexiones se observan en la parte superior donde se deben cuidar las conexiones indicadas.

Esta pantalla fue empleada debido a su compatibilidad con la tarjeta de desarrollo y el protocolo de comunicación SPI. Las principales líneas a conectar se pueden observar en la figura 16 y son:

- Reloj: reloj(CLK), reloj serie(SCK)
- Salida de datos del controlador: Salida de datos en serie(SDO), entrada de esclavo de salida maestra (MOSI), entrada de periférico de salida del controlador (COPI) donde el pin SDO del controlador debe conectarse al pin SDI del periférico
- Entrada de datos del controlador: entrada de datos en serie(SDI), salida de esclavo de entrada maestra (MISO) salida esclavo de entrada del controlador (CIPO) donde el pin SDI del controlador debe conectarse al SDO del periférico
- Selección: Selección de chip (CS), activas en bajo nivel, cuando se inicia la comunicación del dispositivo se bajan de manera individual, selección de esclavo (SS)

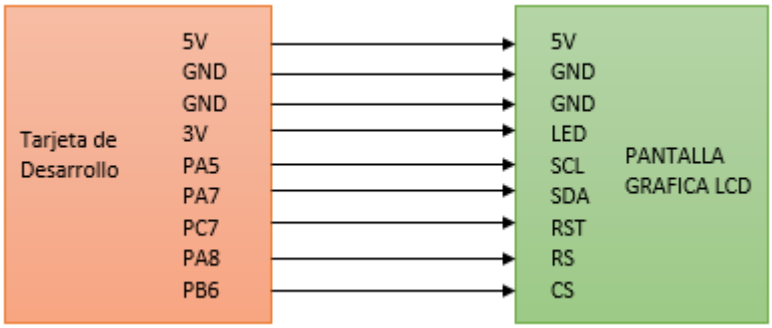


Fig. 16: Configuración básica de Microcontrolador y pantalla empleada.

Además se debe realizar las conexiones de:

- Reset: Permite el reinicio del módulo a la configuración original en caso de fallas.
- Led: Se conecta a 3v en la tarjeta de desarrollo.
- Voltaje: Se conecta a 5v en la tarjeta de desarrollo.
- Tierra: Ambos voltajes deben ir conectados a tierra, la gráfica indica dos conexiones a tierra.

3.6 Programación

A continuación se explica el funcionamiento básico del código de programación empleado en la tarjeta de desarrollo STM32, para ello se inicia con una breve explicación de la plataforma de programación del microcontrolador.

3.6.1 Entorno de Programación

Se empleó la plataforma de programación STM32CubeIDE que permite generar un código inicial de acuerdo con la necesidad del usuario. Cuenta con un conjunto de herramientas y es de implementación rápida, además de ser gratuita y descargable a través de la página de *st.com.*

La interfaz permite seleccionar el tipo de tarjeta a emplear, de manera que se puedan visualizar las características especiales que contiene; para este proyecto se seleccionó el núcleo STM32F407VGT6. Una vez ingresada esta información se presenta una lista de los periféricos especiales con que cuenta este núcleo (figura 17).

Además, ofrece la indicación de los pines específicos del periférico seleccionado para su operación. Finalmente, cuando se ha terminado la configuración la interfaz genera el código fuente correspondiente.

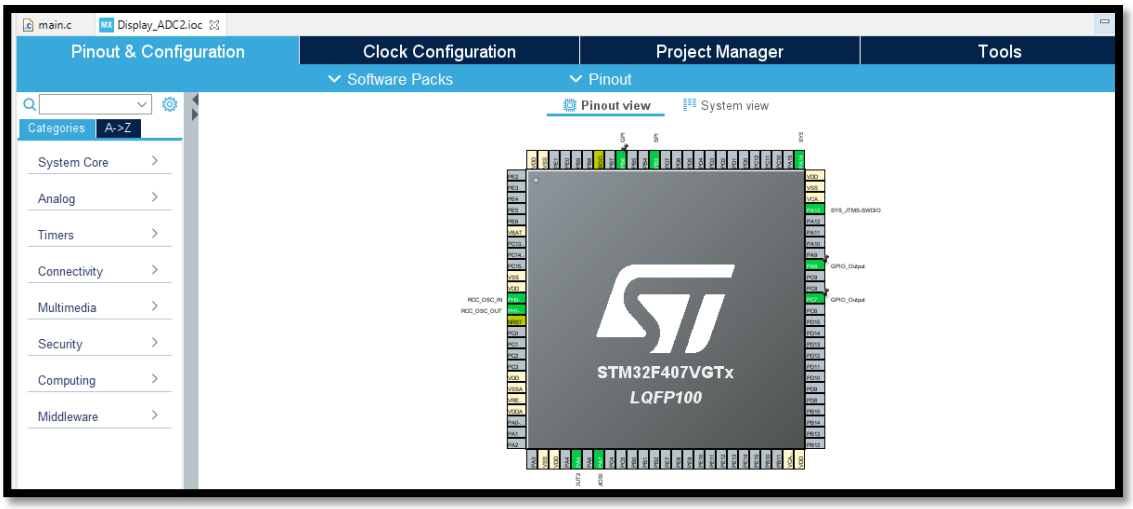


Fig. 17: Interfaz gráfica de la plataforma de programación STM32CubeIDE. Se destacan en color verde los pines que fueron empleados para la programación del código fuente del proyecto.

En la figura 17 se presenta la pantalla de la interfaz gráfica donde se indica el periférico seleccionado, destacando en color verde los pines utilizados en el desarrollo del código fuente y en la tabla 1 se explica su función, así como la configuración empleada.

Tabla 1: Configuración seleccionada por pines del núcleo, presentada en la figura 17.

Función	Tipo	Pin	Configuración
Comunicación serial	SYS	PA13 PA14	Serial Wire: Permite la comunicación entre el microcontrolador y el espacio de trabajo
Activa reloj externo	RCC	PH0 PH1	High Speed Clock (HSE) en crystal/ceramic resonator
PWM	Tim2	PA0	Reloj Interno, PWM canal 1
Convertidor Digital-Analógico	DAC	PA4	Alineado a la derecha. Salida a 100mv

Comunicación con pantalla ST7735	SPI1	PA5 PA7 PC7 PA8 PB6	SCL(SCK): Señal de reloj SDA(MOSI): Salida de datos, con dos cables activos RST: Reinicia de chip DC: Transferencia de datos CS: Selección de chip Conexión dúplex para emplear dos líneas de datos (MOSI, MISO)

La frecuencia del reloj se modifica en el apartado *clock configuration*, como se puede ver en la figura 18. En esta sección se selecciona si el reloj será interno o externo y se puede ajustar los valores de la frecuencia de reloj en el apartado *HCLK (MHz)* según los requerimientos, de ser necesario también se puede modificar los valores de cada periférico.

Como se puede ver en la figura 18 se ha colocado un valor de reloj de 72 MHz el cual se ajusta a la velocidad de funcionamiento con el DAC y el PWM. Otra ventaja de esta plataforma es que permite realizar los ajustes necesarios o bien de forma automática ajusta los valores dependiendo del *HCLK* seleccionado.

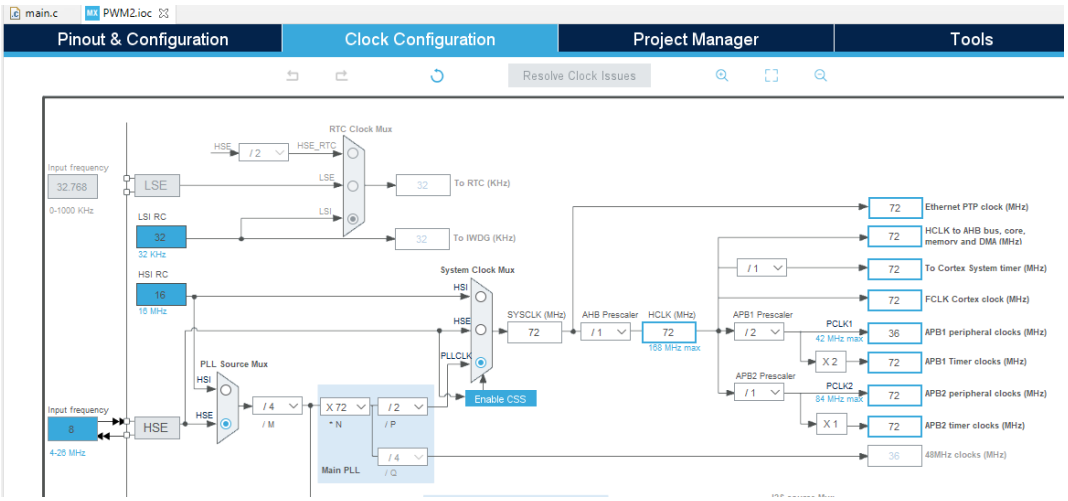


Fig. 18: Apartado de la interfaz gráfica de la plataforma de programación STM32CubeIDE para la frecuencia de reloj. Como se observa, la tarjeta fue configurada en modo HSE a una frecuencia de reloj de 8 MHz.

Finalmente se debió actualizar la versión del programa, ya que para la comunicación con la tarjeta se requería la herramienta de *ST-Link*, figura 19. Una forma de corroborarlo fue conectar la tarjeta, entrar en “Help” y actualizar el *ST-Link* a la versión que automáticamente indica el programa. Esta herramienta fue necesaria para realizar la comunicación entre la plataforma y la tarjeta de desarrollo empleada.

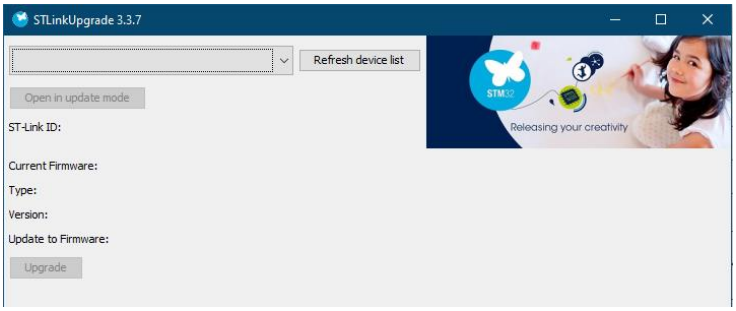


Fig. 19: Pantalla de actualización de sistema de comunicación entre plataforma y tarjeta de desarrollo, se debe conectar la tarjeta previa a la búsqueda de actualización.

3.6.2 Modalidad de Presión del pulso, utilizando el Motor Lineal

Para el funcionamiento de esta modalidad, pulsología, se implementó un algoritmo para generar y aplicar la señal de pulso radial al actuador electromecánico por medio de la salida PWM para el control y activación del actuador electromecánico del simulador de presión.

A través de la plataforma de programación STM32CubeIDE se seleccionó el *timer* 2 en modo de canal 1 a través del cual se configuró la salida *PWM*. Para esto se efectuó un ajuste al *prescaler* con valor de 72 y al contador con 100 cuentas de manera que la frecuencia de trabajo del PWM fuera de 10kHz, creando así el programa base para el funcionamiento del PWM.

Además, a este programa se le agrego la instrucción *HAL_TIM_PWM_START(&htim2,TIM_CHANNEL_1);* para inicializar el *PWM*, estos parámetros se relacionan con la cadena de datos del pulso radial, asociando así cada punto de la cadena a un tiempo determinado del ciclo de trabajo entre 0 y 100% de la señal de reloj. En la figura 20 se presenta el diagrama de flujo del algoritmo para la programación del PWM. El código fuente correspondiente a esta función se presenta en el anexo 4.

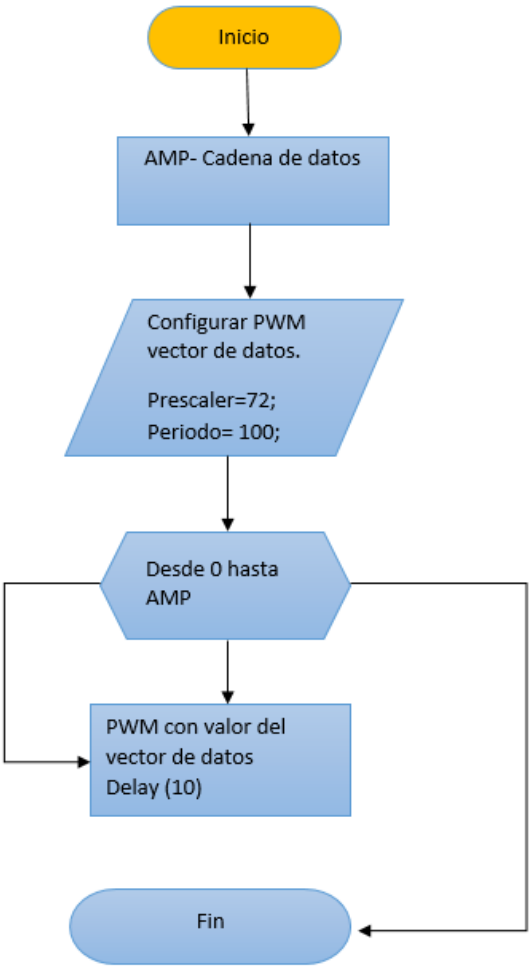


Fig. 20: Diagrama de flujo del algoritmo para la programación del PWM.

3.6.3 Modalidad audible del Pulso

La información de la señal de pulso de manera audible se generó aprovechando el fenómeno de transducción de la bocina subwoofer, de una señal mecánica a señal acústica. Para ello se adaptó un colimador cónico de plástico a la bocina *subwoofer* para dirigir la señal hacia un micrófono tipo electret, la cual fue amplificada y filtrada con el objetivo de reproducirla a través de audífonos.

Para llevar a cabo esta modalidad se desarrolló un algoritmo base a través de la plataforma de programación STM32CubeIDE seleccionando el DAC en modo OUT1. A este programa base se le agregó la instrucción Dac_channel_1 y mediante un ciclo *for* se recorrió la cadena de datos de la señal y cada punto fue enviado a la salida del DAC con el comando HAL_DAC_SetValue, con las definiciones: alinear a la derecha;

recorrer cadena de datos y multiplicar por 24 para amplificar la señal. Para cerrar el ciclo *for* se aplicó un *delay* de 30ms.

Esta señal de salida en la tarjeta de desarrollo fue enviada a un circuito de amplificación en potencia para ser enviada a la bocina. En la figura 21 se presenta el diagrama de flujo del algoritmo para la programación del DAC y el código fuente correspondiente a esta función en el anexo 5.

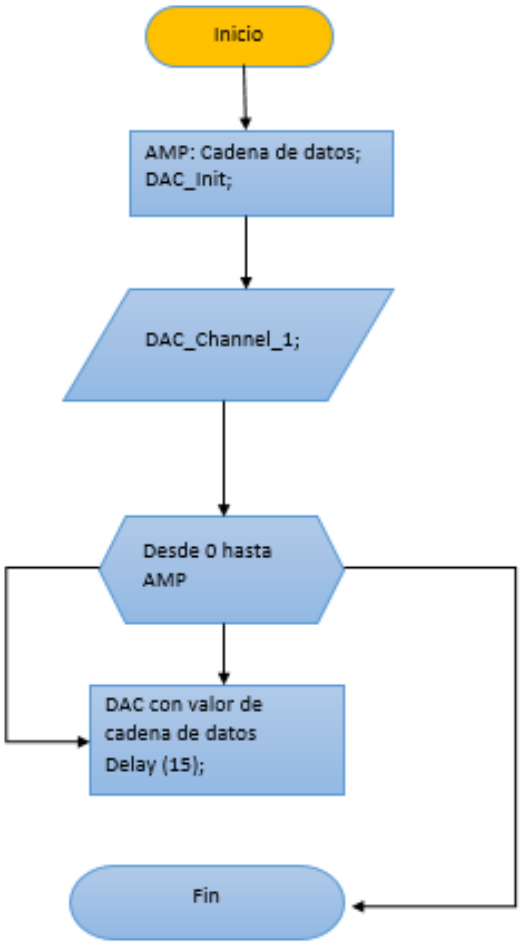


Fig. 21: Diagrama de flujo del algoritmo para la programación del DAC que permite la activación de la bocina.

3.6.4 Modalidad Visual en el Display Gráfico

En la modalidad visual se trabajó con la pantalla gráfica ST7735 como se presentó en la figura 15. Para su activación se empleó el protocolo SPI y en la plataforma de programación STM32CubeIDE se seleccionó la comunicación con la

opción SPI1, activando los puertos en modo *Full-Duplex Master* para el MOSI y el MISO. Los puertos siguientes fueron asignados de acuerdo con las librerías aplicadas.

Al iniciar el código base se agregaron las librerías *ST7735.h* y *GFX_FUNCTIONS.h.* En la primera se definieron los pines que se emplearon para el CS, DC, y RS (Tabla 1), así como la modificación del formato de la pantalla, ancho y altura, mientras que la segunda conservo las definiciones originales (se anexan las librerías principales en los anexos 6 y 7). Posteriormente se inicializaron la cadena de datos, una variable en 0 y la pantalla, esta última por medio del comando *ST7735_Init(0);* con el objetivo de lograr un mejor contraste de la señal se definió el fondo negro por medio de la instrucción *fillscreen(Black).* Se programó el código para enviar la señal de pulso de manera visual, para ello se ajusta la pantalla según la forma en que se desea visualizar la onda de pulso, empleando el comando *ST7735_SetRotation(2),* se inició un ciclo *for* para el conteo de las ondas visualizadas, dentro de este se colocó otro ciclo *for* para el recorrido de la cadena de datos del pulso radial empleando el comando *ST7735_DramPixel* se asoció la cadena de datos a la salida en la pantalla, se ajustó el conteo para la visualización de dos eventos del pulso, esto debido a las limitaciones por el tamaño de la pantalla, finalmente se indicó con el comando *fillScreen (White)* el color de la forma de onda de pulso visualizada junto con un *Delay*. En la figura 22 se presenta el diagrama de flujo del algoritmo para la programación de la comunicación con el display ST7735. El código fuente correspondiente a la programación del desplegado gráfico se presenta en el anexo 8.

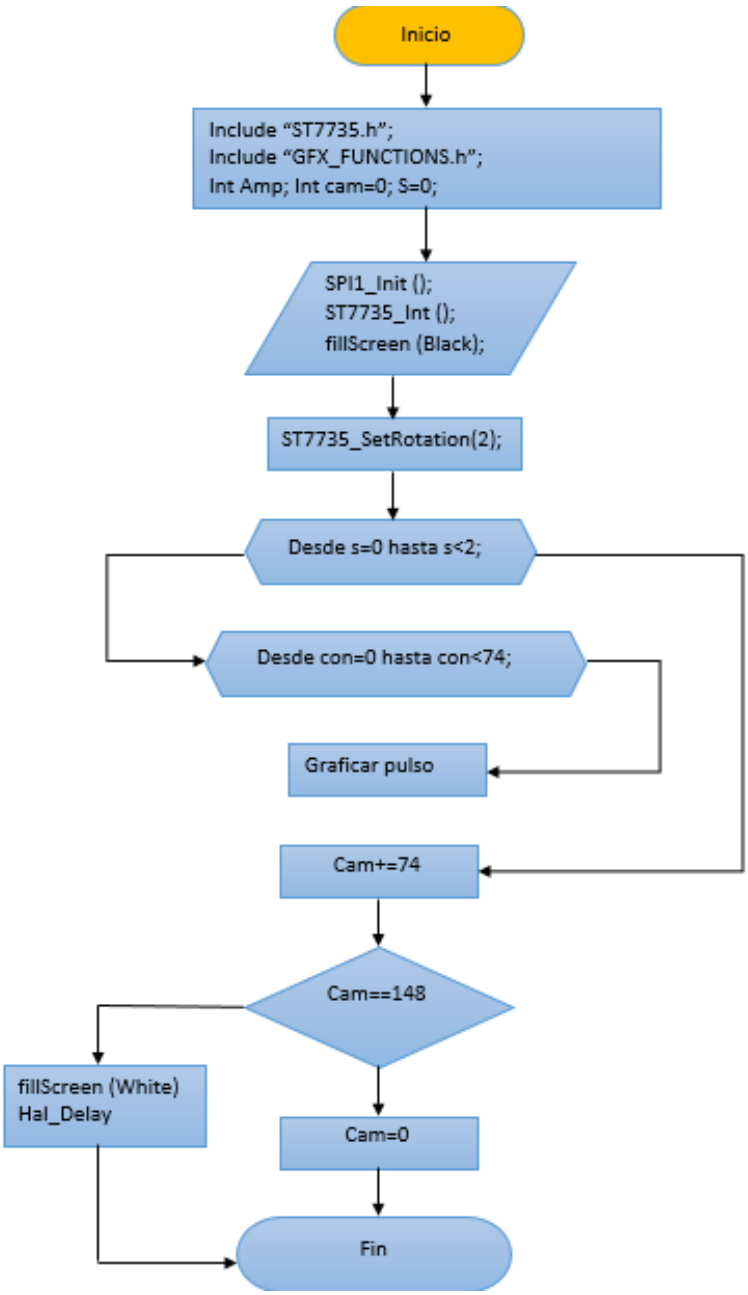


Fig. 22: Diagrama de flujo del algoritmo para la programación de la comunicación con el display ST7735 para la visualización de la onda de pulso arterial radial.

3.7 Circuitos complementarios

Se implementaron algunos circuitos complementarios a la tarjeta de desarrollo STM32F407VGT6Discovery que a continuación se presentan.

3.7.1 Puente H

Para la activación del motor lineal de bobina de voz LVCM-010-013-01 se empleó el módulo FC-120 basado en el puente H L298N, el cual se conectó a la salida PWM de la tarjeta de desarrollo. A su vez, la salida del módulo se conectó al motor y ajustando el voltaje de entrada del módulo se logró el desplazamiento del pistón siguiendo la señal de pulso radial (figura 23).

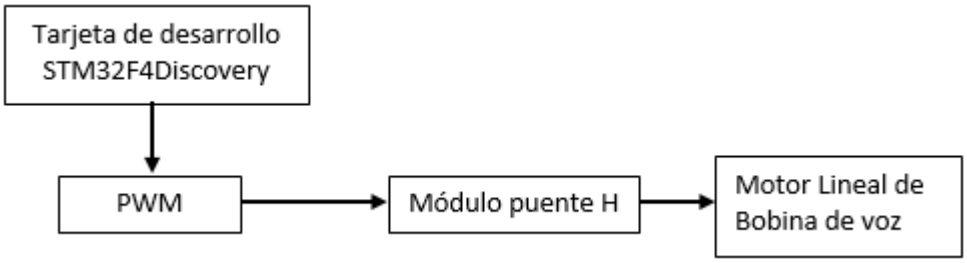


Fig. 23: Integración del Puente H para el control de desplazamiento del pistón en motor lineal.

3.7.2 Amplificador de Potencia

Para la reproducción de la señal de pulso como audio se empleó una bocina *subwoffer*, para lo cual fue necesario adaptar un circuito amplificador (figura 24). Se acondicionó el amplificador de potencia de propósito general LM675 (Texas Instruments, USA), conectado de acuerdo con la configuración recomendada en el *datasheet* (figura 25), con alimentación de 16V, un factor de amplificación de 16 y límite de corriente de 3A

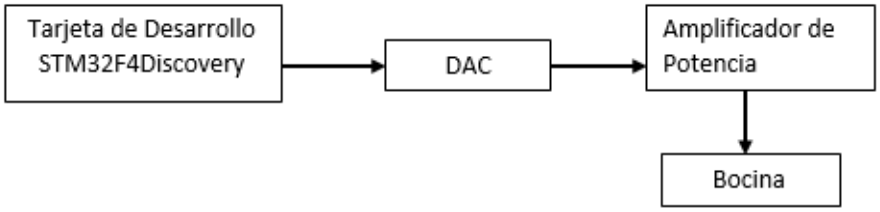


Fig. 24: Integración del amplificador de potencia para reproducción del pulso como audio.

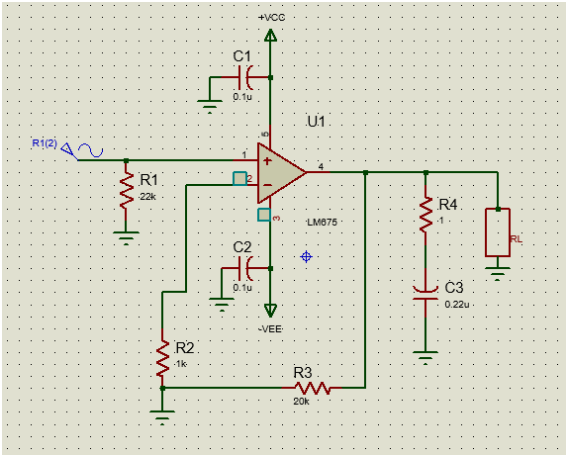


Fig. 25: Circuito amplificador de potencia basado en el CI LM675.

3.7.3 Recepción de audio

El siguiente circuito diseñado fue la adaptación de micrófono electret para la recepción del audio generado en la bocina. Para ello se amplificó la señal recibida por el micrófono por medio del C.I. UA741 al cual se le asignó una ganancia de 20, la salida de este circuito fue la entrada del LM386, circuito integrado especializado para audio, con un factor de ganancia de 20 agregando un filtro pasa altas para la eliminación de CD. El circuito utilizado se presenta en la figura 26.

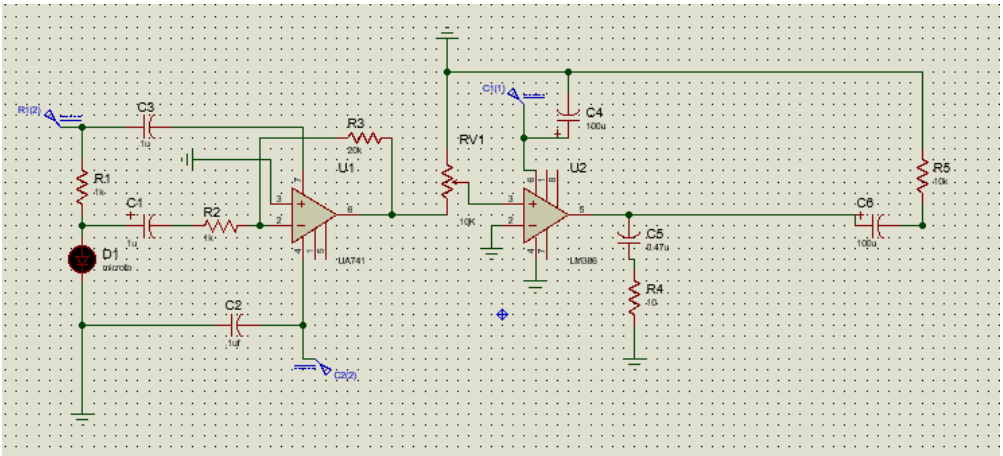


Fig. 26: Circuito para recepción de audio por medio de un micrófono electret, pre-amplificación y un amplificador de audio.

3.7.4. Interfaz SPI a Pantalla

Para este propósito se desarrolló un programa para la activación del protocolo de comunicación SPI en la tarjeta de desarrollo aplicado al *display* ST7732, de acuerdo con el diagrama a bloques de la figura 27.

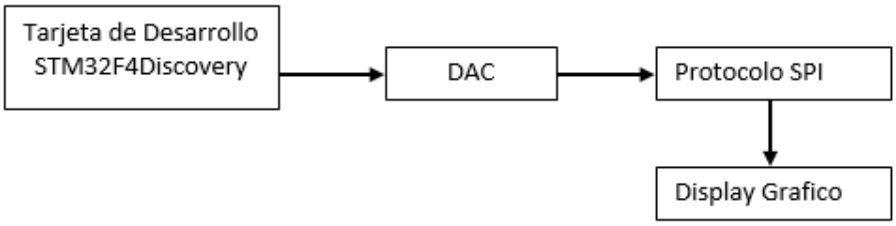


Fig. 27: Integración de conexión entre Tarjeta de desarrollo con la pantalla a través del protocolo SPI.

Capítulo 4. Pruebas

Se efectuaron pruebas de desempeño a cada una de las modalidades del simulador propuestas en este proyecto; se realizaron pruebas para ajustes y acondicionamiento del desempeño de las diversas etapas, empleando las facilidades del osciloscopio DSO1002A (*Agilent Technologies, USA*), para registrar, almacenar, analizar y procesar las señales de pulso.

4.1 Señal de pulso Digitalizada

Se realizaron pruebas, para observar y comprobar que la señal de pulso digitalizada mostrara la forma con los detalles más importantes, además se efectuó análisis espectral de la señal para determinar sus armónicos.

Para efectuar la comprobación de la información contenida en las cadenas de datos, se desarrolló un algoritmo empleando un ciclo *for*, con una duración de 30 ms, lo que permite mostrar con mayor claridad los detalles de la señal de pulso (figura 28). Los resultados visualizados en el osciloscopio de la figura 29, representan un simil entre la definición del tiempo para el ciclo *for* y el ajuste de la base de tiempo del osciloscopio para la apreciación de los detalles.

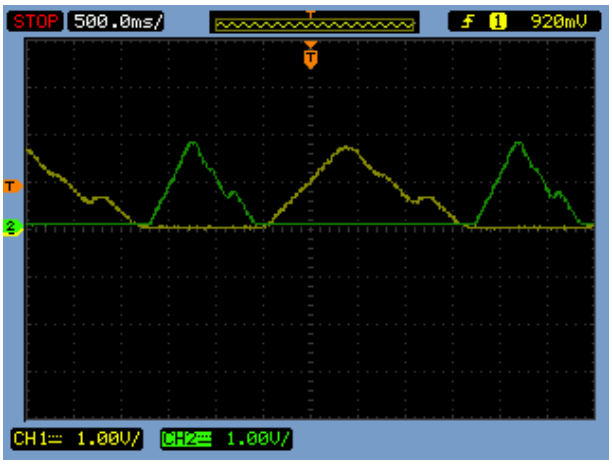
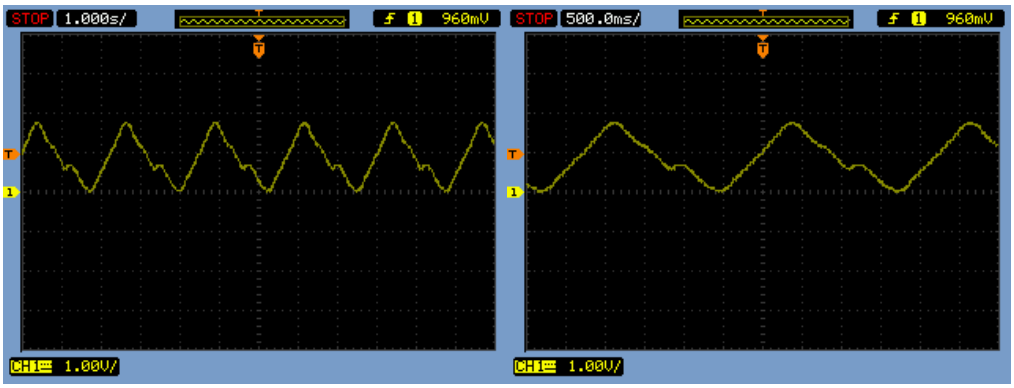
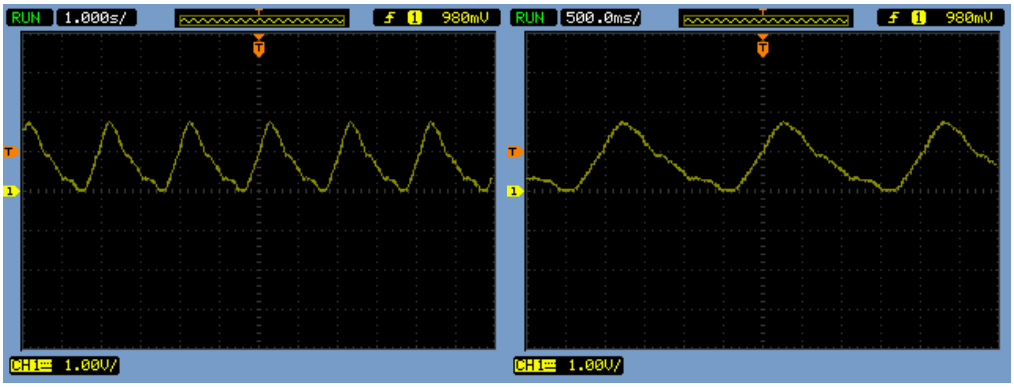


Fig. 28: Resultado de la señal entre la duración del ciclo de 15ms en verde y 30ms en amarillo, la apertura de la señal en 30ms permite apreciar con mayor detalle las pendientes.

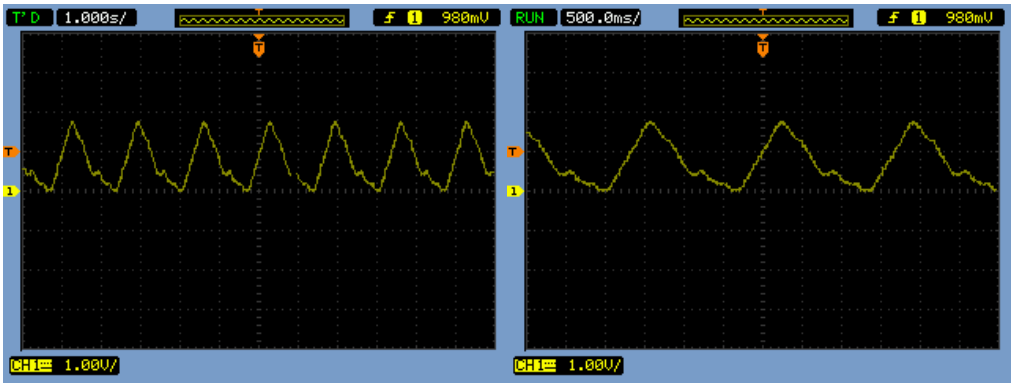
Se realizaron pruebas adicionales para evaluar el desempeño del algoritmo de presentación, empleando señales sinusoidales, triangulares y rectangulares con amplitudes y frecuencias diferentes, así como con señales moduladas.



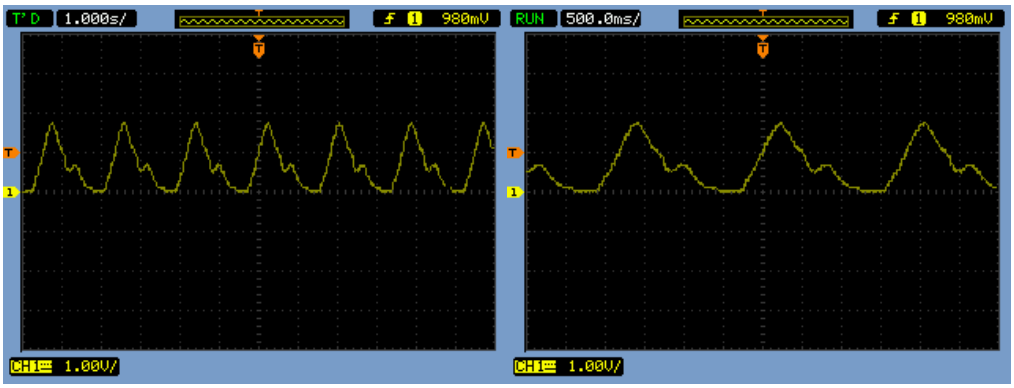
a) Onda Normotenso



b) Onda Hipertenso Establecido



c) Onda Hipertenso Atípico

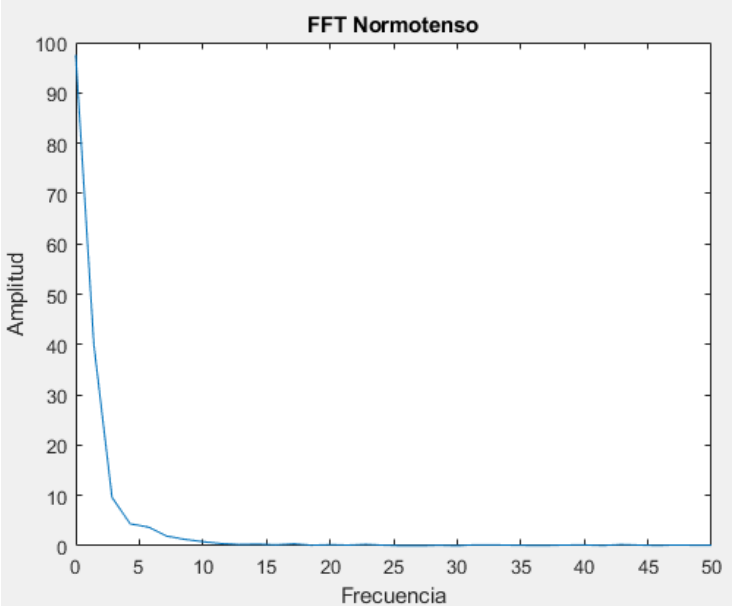


d) Onda Hipertenso Reciente Atípico

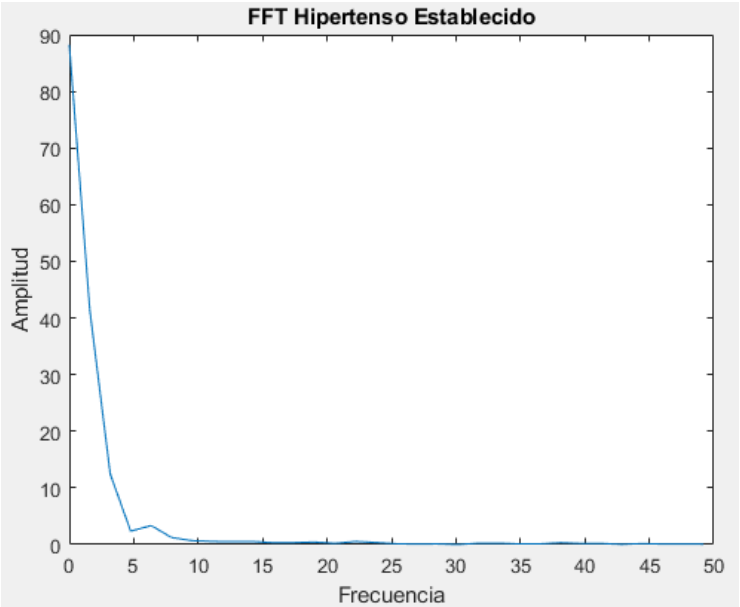
Fig. 29: Se aplicó el algoritmo con el ciclo *for* para la visualización de la señal; se comparan dos escalas de la base de tiempo del osciloscopio de 1000 y 500ms. En esta última se observan mejor los detalles de la forma de la onda.

4.1.1 Análisis espectral de la cadena de datos

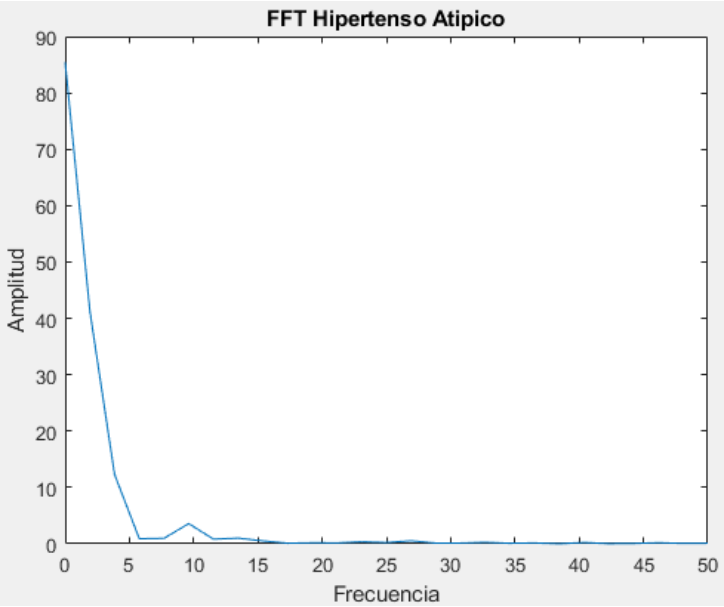
Con la ayuda de MATLAB R2020 se realizó un programa para efectuar un análisis espectral a las cadenas de datos para conocer los armónicos de las señales (figura 30).



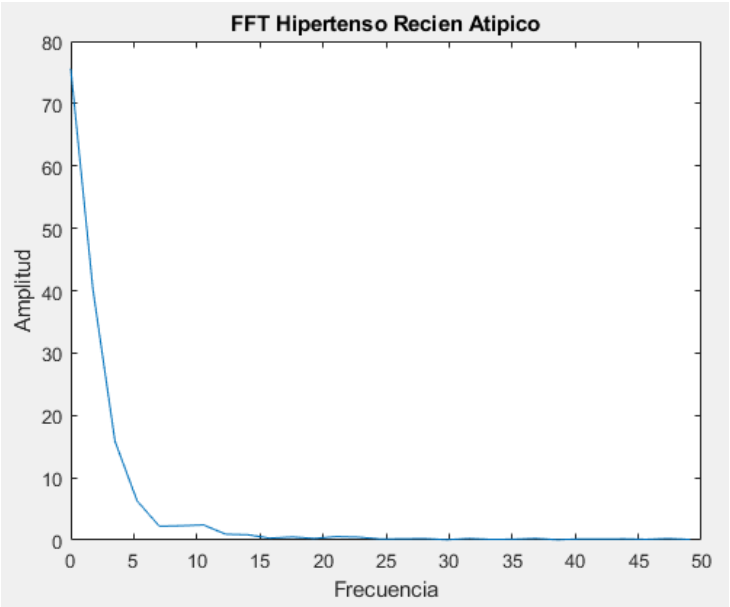
a) Análisis espectral onda normotenso.



b) Análisis espectral onda hipertenso establecido.



c) Análisis espectral onda hipertenso atípico



d) Análisis espectral onda hipertenso Reciente atípico

Fig. 30: Espectros en frecuencia correspondientes a las señales de pulso: a) Normotenso; b) Hipertenso atípico; c) Hipertenso establecido y d) Hipertenso Reciente atípico

4.2 Simulador de la Señal de Presión

4.2.1 Determinación del ciclo de trabajo del PWM

Para determinar el ciclo de trabajo apropiado de la modulación del PWM que permitiera reconstruir la señal en forma de onda analógica, se tomaron los datos obtenidos con el 50% y 100% del ciclo de trabajo y se aplicó un filtro pasa bajas de tercer orden (figura 31), con el objetivo de encontrar la modulación que permitiera conservar los detalles de la señal de pulso asociados a los fenómenos físicos del flujo sanguíneo para aplicarse al actuador electromecánico. De acuerdo con las pruebas efectuadas se determinó un ciclo de trabajo del 100% a una frecuencia de reloj de 1kHz. Las ondas analógicas obtenidas después del filtrado se presentan en las figuras 32 y 33.

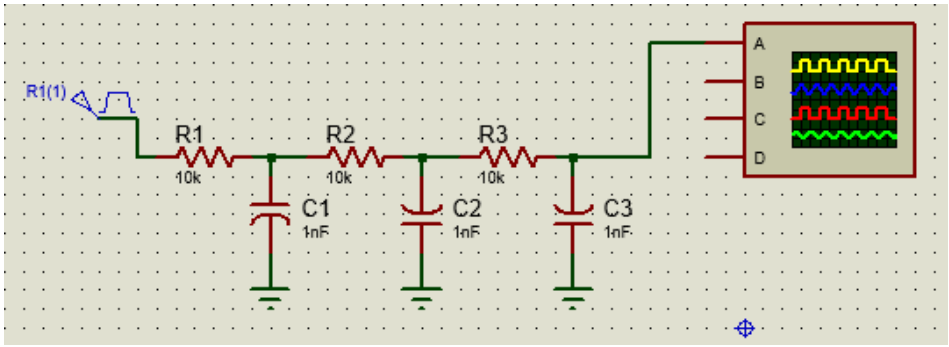
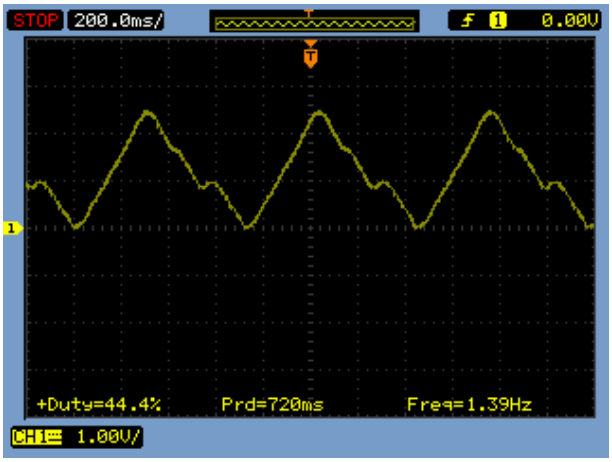
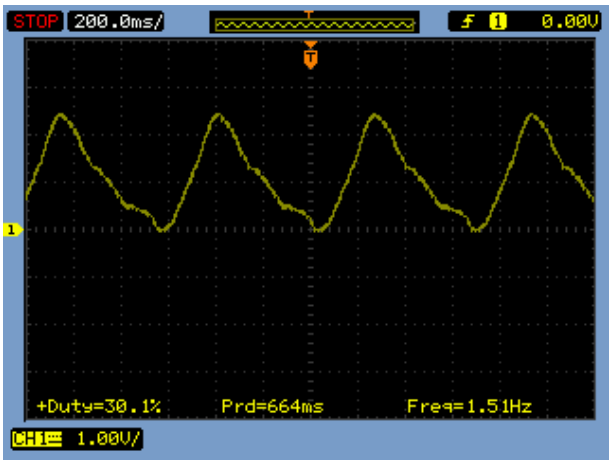


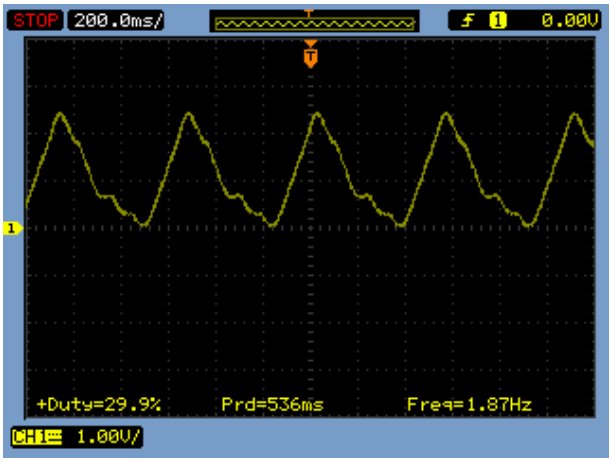
Fig. 31: Filtro pasa bajas de tercer orden para reconstrucción de la señal modulada.



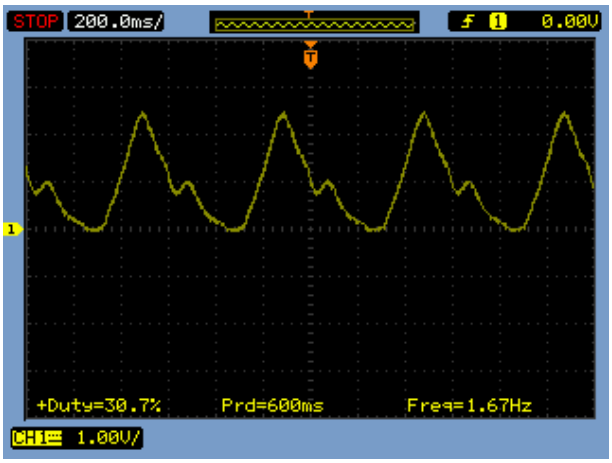
a) Onda Normotenso



b) Onda Hipertenso Establecido

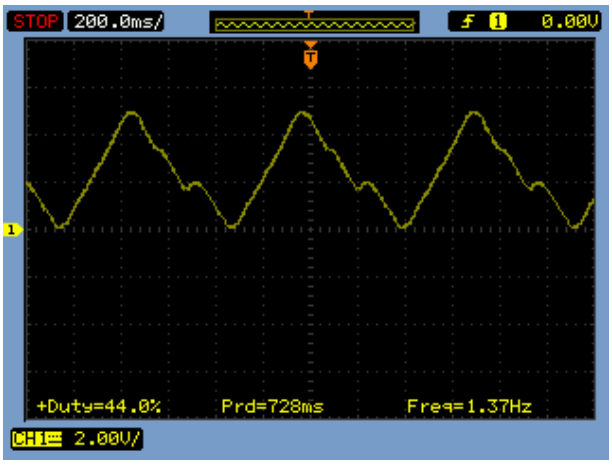


c) Onda Hipertenso Atípico

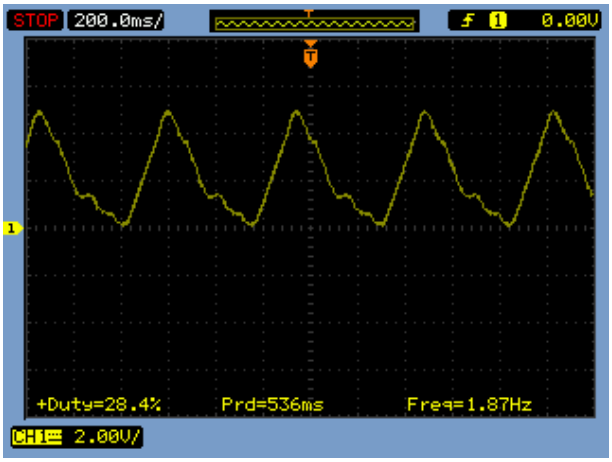


d) Onda Hipertenso Reciente Atípico

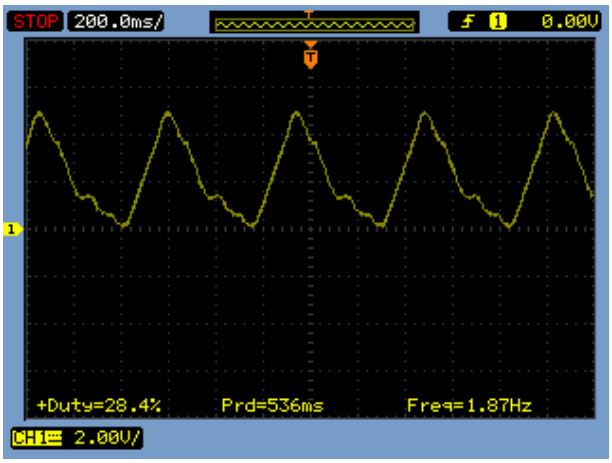
Fig. 32: Ondas de pulso reconstruidas por medio del filtro con base en la modulación del 50%.



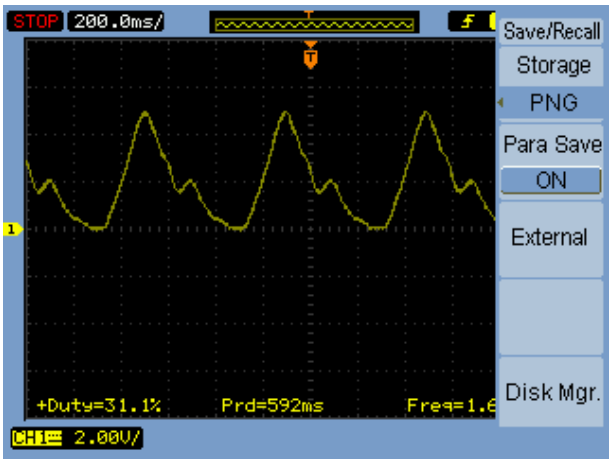
a) Onda Normotenso



b) Onda Hipertenso Establecido



c) Onda Hipertenso Atípico



d) Onda Hipertenso Reciente Atípico

Fig. 33: Ondas de pulso reconstruidas con base en la modulación del 100%, se observa un aumento considerable en el voltaje y las ondas con mayor detalle en las pendientes.

4.2.2 El módulo puente H

Se utilizó el módulo puente H L298N (figura 34), para refuerzo de corriente hasta 2A para la activación del motor lineal.

Se realizaron pruebas con voltajes de alimentación entre 5v y 8v para no sobreexcitar a la bobina del motor y evitar desplazamientos mayores a la carrera del pistón del motor. Como resultado de las pruebas se definieron las condiciones de 6V de alimentación y una amplitud de la señal de 2V del PWM para producir los desplazamientos del pistón del actuador acorde a la señal de pulso detectable en la yema de los dedos.

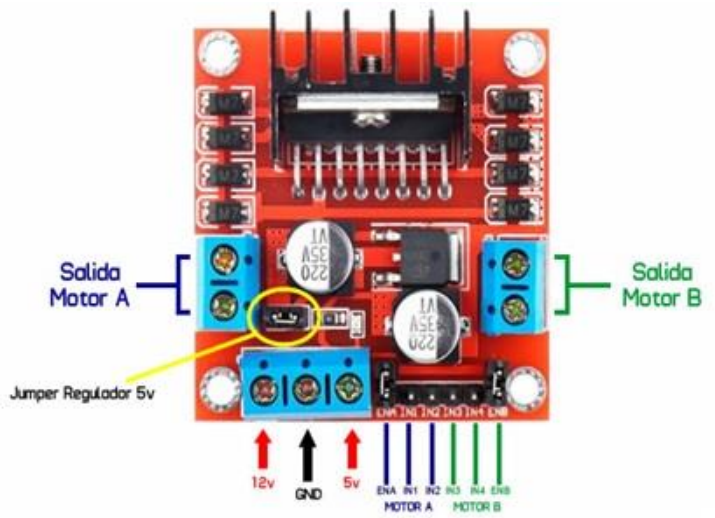


Fig. 34: Modulo Puente H, con formato en conexiones para aplicación en motores.

4.2.3 Prototipo para palpación

Para la construcción del prototipo táctil se empleó metacrilato (acrílico) para la base del montaje de los actuadores con tornillos de ajuste de la posición y soporte ergonómico para la colocación de los dedos sobre la membrana de látex elegida como interfaz para la palpación, aprovechando las características de resistencia y flexibilidad. En las figuras 35 a 37 se muestran las diferentes vistas del prototipo de palpación.

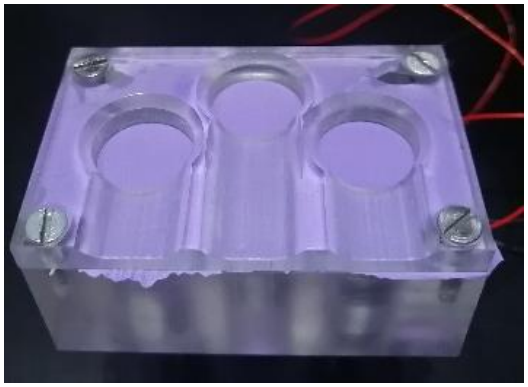


Fig. 35: Ensamble del actuador electromecánico.

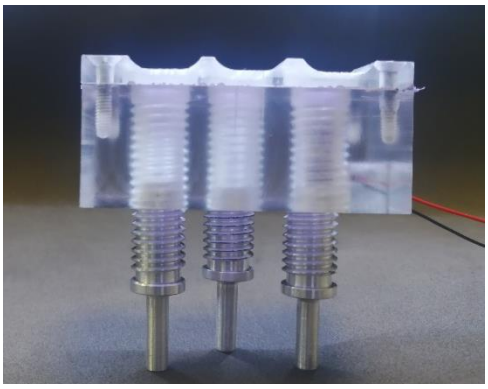


Fig. 36: Vista Frontal.

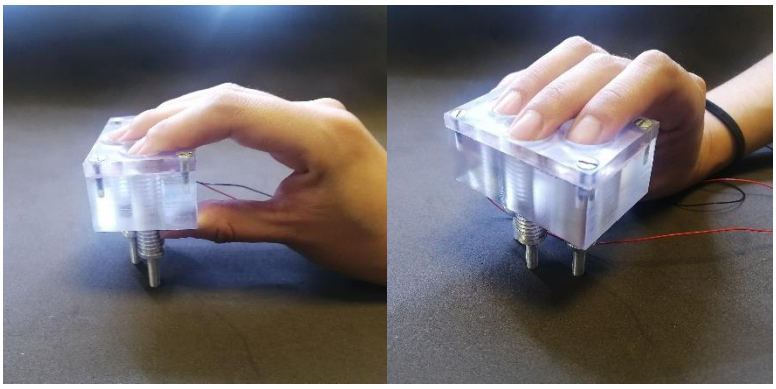


Fig. 37: Vista Frontal e izquierda con aplicación de presión.

4.3 Acondicionamiento de la señal de pulso como audio

4.3.1. Amplificador de potencia

Una vez realizado el programa por medio de la salida DAC, se realizaron pruebas de acondicionamiento al amplificador LM675 de propósito general con amplio ancho de banda. Para evitar saturación de la señal se definió una ganancia de 16 y un voltaje de alimentación de 12V.

4.3.2. Acoplamiento bocina-micrófono

Se realizaron pruebas para observar el desempeño del acoplamiento de la señal entre la bocina y micrófono, mediante el uso del colimador (figura 38), a través del registro de la amplitud de la señal recibida por el micrófono, definiéndose un voltaje de polarización de 12V.

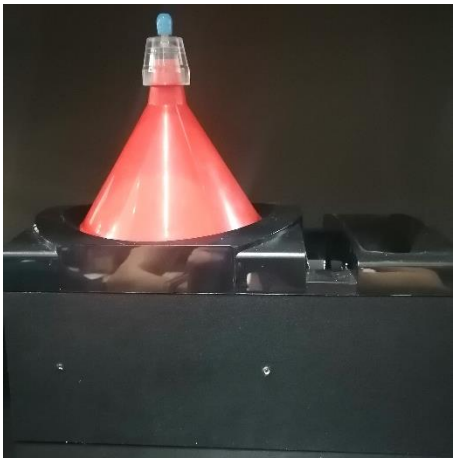


Fig. 38: Adaptación de colimador a bocina hasta micrófono.

4.3.3 Reproducción de señal de pulso en audífonos

Mediante el análisis espectral a la señal registrada en audífonos (figura 39), se comprobó el efecto de transducción que realiza la bocina subwoofer como generador de armónicos en el rango audible (figura 40).

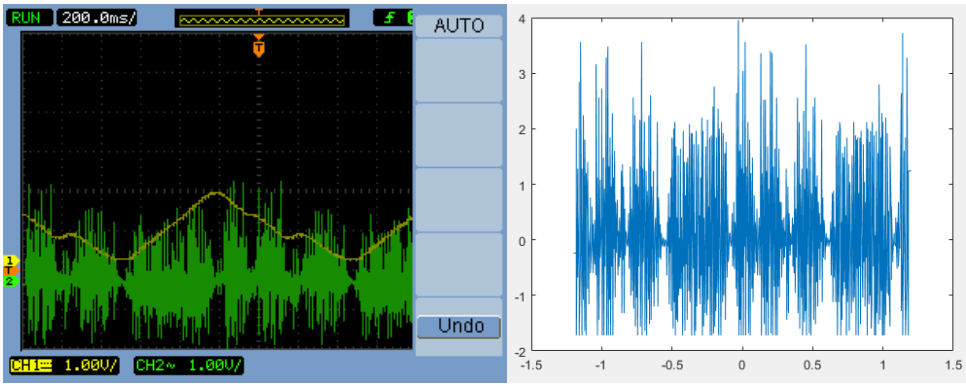


Fig. 39: La señal de audio con los lóbulos conformados por los armónicos y sub-amónicos analizando.

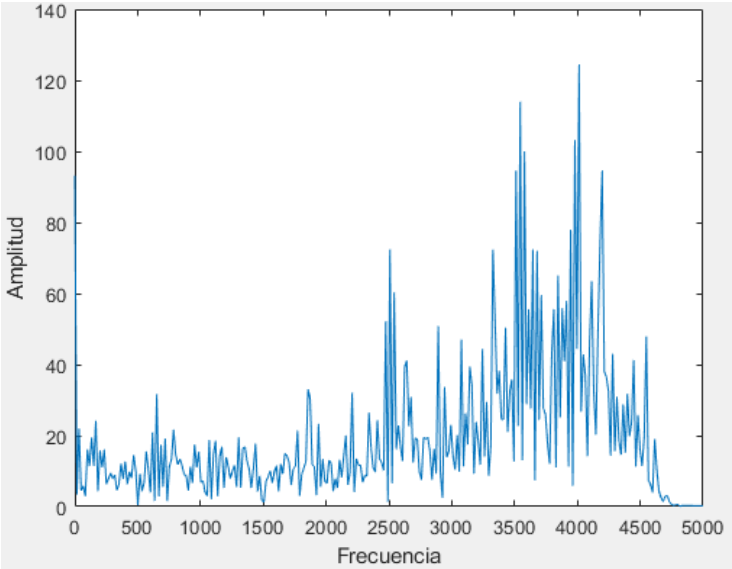


Fig. 40: Espectro de la señal de pulso en audífonos. Obsérvese la presencia de armónicos en el rango audible.

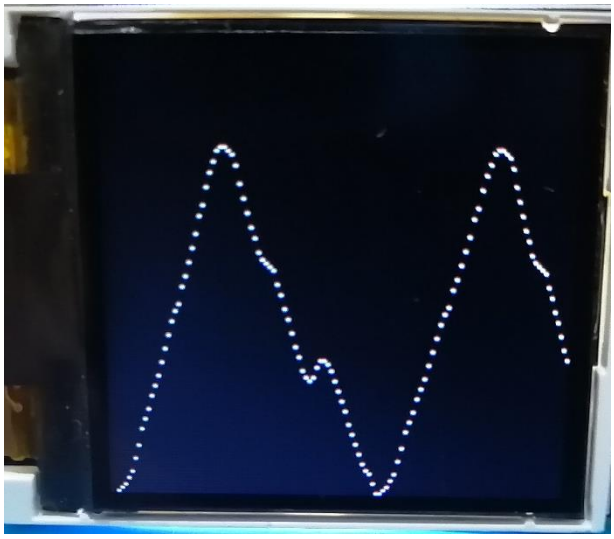
4.4 Presentación Gráfica

Para la prueba de la pantalla se realizó la conexión entre la tarjeta de desarrollo STM32F407VGT6Discovery y se aplicó voltaje para verificar el encendido de la pantalla

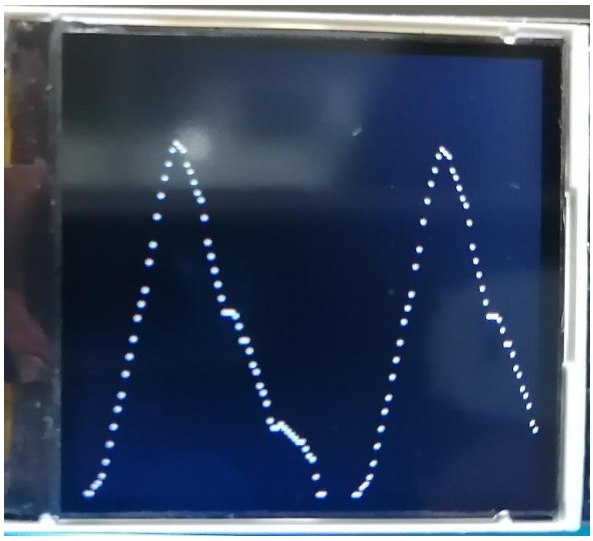
Se realizó la prueba de comunicación entre la tarjeta de desarrollo y el display gráfico. Para ello se implementó un algoritmo para comprobar la utilidad de las librerías *ST7735.h* y *GFX_FUNCTIONS.h,* disponibles para la pantalla ST7735 y definir el formato de pantalla.

Como parte de estas pruebas, se corrió un único código que incluye el comando *testAll ()*, el cual permite enviar diversos caracteres que iluminan cada parte de la pantalla LCD verificando así la comunicación entre la pantalla y la tarjeta, así como el funcionamiento adecuado de las librerías.

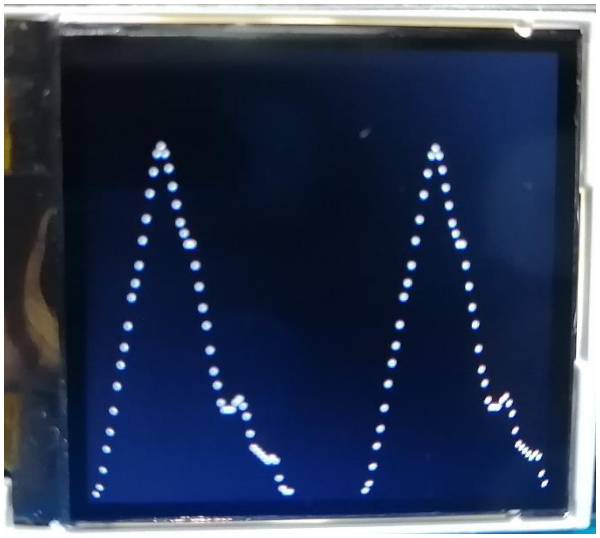
Finalmente se realizaron pruebas de visualización de la señal, enviando la cadena de datos de la señal de pulso. El resultado de la presentación gráfica de la señal de pulso se muestra en la figura 41.



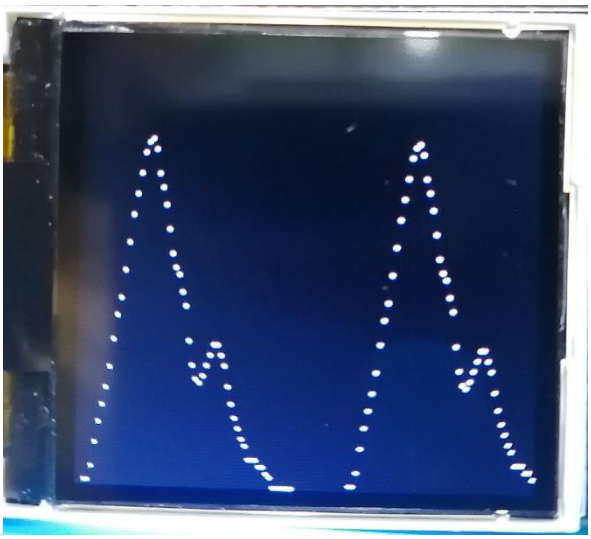
a) Onda Normotenso



b) Onda Hipertenso Establecido



c) Onda Hipertenso Atípico



d) Onda Hipertenso Reciente Atípico

Fig. 41: Visualización de la onda de pulso en display.

Capítulo 5. Discusión

Consideramos que la propuesta planteada de ofrecer la información de pulso radial mediante las modalidades de palpación, visual y auditiva, que involucran a tres sentidos de una persona podría representar una reducción en el tiempo de aprendizaje por la diferencia de la forma tradicional al presentar la información del pulso solamente por palpación como se propone en el trabajo realizado por Shin, Ki Young [15].

La modalidad de palpación que utilizó un actuador lineal electromecánico permitió dar seguimiento a la señal de presión que se registró hasta las yemas de los dedos. Sin embargo, se requiere efectuar pruebas de cuantificación de la presión utilizando un tonómetro en las membranas de registro para su validación.

En la modalidad de audio se utilizó la relación observada entre la señal de presión y la de audio. De esta forma, esta propuesta plantea un esquema experimental que busca reducir el tiempo de entrenamiento para el aprendizaje del conocimiento de la señal de pulso radial. Para ello se requiere una etapa de pruebas de validación de su utilidad.

En la modalidad visual se cuidó que la visualización fuera sin interrupciones y se reprodujera en una pantalla LCD de manera continua y simultánea con las otras modalidades de presentación. Como en el caso anterior, se requiere de una etapa de validación práctica.

Capítulo 6. Conclusiones y Perspectivas

6.1 Conclusiones

Se desarrolló un simulador de la señal del pulso arterial radial, basado en la Medicina Tradicional China, con propósitos de enseñanza que presenta la información del pulso en tres modalidades: por palpación de la presión en las yemas de los dedos, como audio, derivado de los armónicos y sub-armónicos que conforman la señal y de manera gráfica, la representación de la forma.

La propuesta mediante tres modalidades tiene como objetivo facilitar el aprendizaje de la información contenida en el pulso con diversos padecimientos orgánicos, así como una posible reducción del tiempo de enseñanza.

Se cuantificaron y se digitalizaron cuatro señales del pulso radial de gráficas reportadas en la literatura asociadas con diferentes estados de salud y se concatenaron para conformar secuencias de señales de pulso, las cuales, fueron utilizadas para su reproducción como presión, audio e imagen.

Se desarrolló un simulador de la señal del pulso arterial radial, basado en una tarjeta digital que incluye un microprocesador de alto desempeño, programada mediante un algoritmo diseñado a la medida para manejar al actuador electromecánico lineal de bobina móvil, la bocina *subwoofer* para reproducir bajas frecuencias y el display gráfico.

El sistema desarrollado involucra simultáneamente a tres sentidos de los estudiantes. De esta manera, se espera contribuir a la reducción del tiempo de aprendizaje de la pulsología.

6.2 Perspectivas

Hasta el momento, el sistema desarrollado cuenta con cuatro modelos gráficos de la señal del pulso arterial radial; esta condición limita su utilidad. Se sugiere trabajar con el desarrollo de un sistema de adquisición de señales de pulso, la creación de una base de datos con señales del pulso radial con su correspondiente información clínica, de manera que permita a los estudiantes familiarizarse y asociar diferentes padecimientos.

Se debe registrar la presión ejercida por el actuador electromecánico sobre la membrana para validar los valores.

Sería de utilidad incluir una interfaz gráfica para el manejo de una futura base de datos de señales de presión.

Referencias

[1] R. Tagle, "Diagnóstico de hipertensión arterial," *Revista Médica Clínica Las Condes,* vol. 29, no. 1, pp. 12-20, 2018.

[2] A. C. Y. Tang, "Review of traditional Chinese medicine pulse diagnosis quantification," in *Complementary therapies for the contemporary healthcare*, M. S. a. R. d. Medeiros, Ed., 2012, pp. 61-80.

[3] R. Peco. (2022, 22 de Enero). *¿Que es una Onda de Presión Arterial?* [Online]. Available: https://campusvygon.com/onda-de-presion-arterial-dr-alonso/

[4] D. P. Martín, "Análisis de la onda de presión arterial en Anestesiología y Cuidados Intensivos II," *Revista electrónica AnestesiaR,* vol. 12, no. 11, p. 2, 2020.

[5] D. P. Martín, "Análisis de la onda de presión arterial en Anestesiología y Cuidados Intensivos I," *Revista electrónica AnestesiaR,* vol. 12, no. 6, p. 4, 2020.

[6] F. M. Clara, A. Casarini, A. G. Scandurra, G. J. Meschino, and A. R. Introzzi, "Índice para la evaluación de las reflexiones mediante análisis de la onda de pulso radial," *Revista argentina de cardiología,* vol. 76, no. 4, pp. 272-277, 2008.

[7] J. J. S.-M. d. F. Sáez-Pérez, "Distensibilidad arterial: un parámetro más para valorar el riesgo cardiovascular," vol. 34, no. 6, pp. 284-290, 2008.

[8] F. Rozman. (2017, 04 de abril) 10 Puntos de palpación de los pulsos arteriales *Generación Elsevier* Available: https://www.elsevier.com/es-es/connect/medicina/10-puntos-de-palpacion-de-los-pulsos-arteriales

[9] F. Clara, G. Blanco, A. Casarini, P. Corral, G. Meschino, and A. Scandurra, "Estimación de la velocidad de propagación aórtica basada en el análisis de la onda de pulso radial," *MEDICINA,* vol. 71, no. 3, pp. 231-237, 2011.

[10] D. J. Sussmann, *Acupuntura: teoría y práctica*. Editorial Kier, 2011.

[11] G. Stux and B. Pomeranz, *Fundamentos de acupuntura*. Springer Science & Business Media, 1999.

[12] W. K. Kit, S. P. Cosmos Internet Sdn Bhd, Kedah, Malaysia, Ed. *El gran libro de la medicina china*, URANO, S.A. ed. Meredian Holding Group SA, 2003.

[13] M. A. S. Rodríguez, "Capítulo 25. Los pulsos chinos: La imagen y sus nombres (1ª parte)," en *Cruce de miradas, relaciones e intercambios*, 2010, pp. 401-418: Universidad de Granada.

[14] H. Wang and Y. Cheng, "A quantitative system for pulse diagnosis in traditional Chinese medicine," in *2005 IEEE Engineering in Medicine and Biology 27th Annual Conference*, 2006, pp. 5676-5679: IEEE.

[15] K. Y. Shin *et al.*, "A pulse wave simulator for palpation in the oriental medicine," in *2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, Boston, MA, USA, 2011, pp. 4163-4166: IEEE.

[16] S. E. C. Bastidas, H. E. E. C. Meza, A. J. C. Bastidas, and A. A. Cabrera, "Capítulo 2: Sensores y dispositivos utilizados para el cuidado de la salud," in *Redes de sensores Inalámbricas (WSN) caso de aplicación* Bogota D.C. : Davinci-UNAD 2018, pp. 91-127.

[17] "SPOT Light SpO2 Functional Tester " vol. 2, F. BIOMEDICAL, Ed., ed, 2012, p. 28.

[18] F. M. CLARA *et al.*, "Identificación de hipertensos recientes mediante análisis de onda de pulso radial," *Rev Fed Arg Cardiol,* vol. 39, no. 3, pp. 208-214, 2010.

[19] J. M. H. Peter H. Charlton, Samuel Vennin, Ye Li, Phil Chowienczyk & Jordi Alastruey. Consultado: (20 de octubre 2021). *Pulse Wave Database (PWDB): A database of arterial pulse waves representative of healthy adults*. Available: https://zenodo.org/record/3275625#.Yw0slHbMLIX

[20] P. H. Charlton, Mariscal Harana, Jorge, Vennin, Samuel Michel Laurent, Willemet, Marie Cecile, Chowienczyk, Philip Jan, Alastruey-Arimon, Jordi. Consultado: (20 de Octubre 2021). *Modelling arterial pulse wave propagation during healthy ageing* [Web]. Available: https://peterhcharlton.github.io/pwdb/ppwdb.html
[21] A. Rohatgi. (2021, 20 de Noviembre). *WebPlotDigitizer* [Web]. Available: https://automeris.io/WebPlotDigitizer
[22] STMicroelectronics, "Application note AN3126," 4 ed, 2020, pp. 1-23.
[23] STMicroelectronics, "Application note AN5543," 2 ed, 2021, pp. 1-24.
[24] Moticont. (2020, 20 de Noviembre). *Moticont* [Online]. Available: http://moticont.com/
[25] E. Segovia, J. Carbajo, A. Espí, and J. Ramis, "Análisis numérico del comportamiento vibratorio del conjunto móvil de un altavoz dinámico," *Revista Internacional de Métodos Numéricos para Cálculo y Diseñoñ en ingeniería,* vol. 31, no. 3, pp. 154-160, 2015.

Anexos

Anexo 1: Gráficas con la aplicación *WebPlotDigitizer*

Gráficas obtenidas con la aplicación *WebPlotDigitizer* previo a la manipulacion de datos.

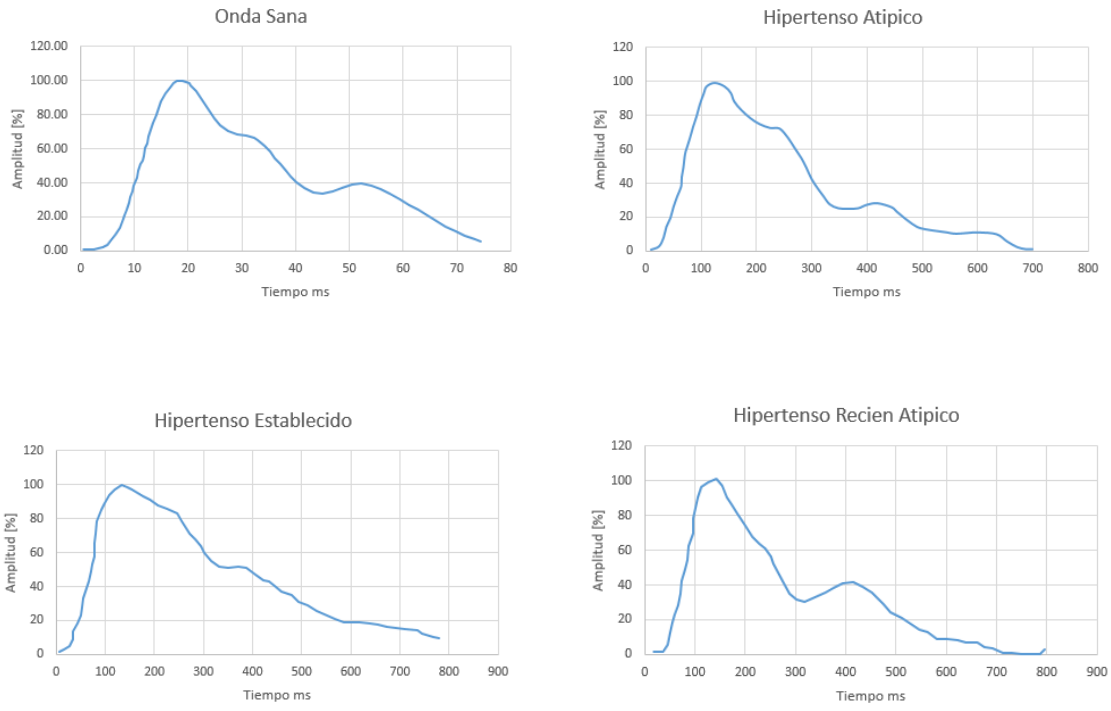


Fig. 42: Representación de gráficas de ondas de pulso obtenidas por medio de la aplicación con valores originales.

Anexo 2: Cada de datos empleada para la programación de pulso radial

Para anexar en programación se emplea únicamente el eje Y respetando el tiempo por medio de un *Delay* en ms, para una visualización de más ondas el *Delay* puede ser modificado.

Tabla 2: Cadena de datos para programación.

Datos Onda Normotenso		Datos Onda Atípica		Datos Onda H. Establecido		Datos Onda R. Atípica	
Eje X	Eje Y	Eje X	Eje Y	Eje X	Eje Y	Eje X	Eje Y
0	2	0	2	0	2	0	0
13	1	10	4	7	3	11	7
24	2	16	9	14	5	17	13
30	4	20	15	18	9	20	17
34	6	25	20	19	14	23	19
40	10	29	26	25	19	27	25
46	14	34	33	29	24	30	28
50	19	39	39	31	28	34	31
52	22	40	44	33	34	35	32
54	25	42	50	37	39	40	36
57	29	45	58	39	44	42	39
59	32	48	62	42	50	45	41
61	35	51	67	43	54	50	43
63	39	55	74	47	58	50	46
67	43	60	80	47	66	53	47
69	47	64	87	48	71	56	48
72	51	69	93	50	79	61	49
74	54	73	98	56	85	69	56
76	56	83	100	61	90	79	60
78	61	96	98	66	94	87	65
80	64	105	94	73	98	94	73
82	68	109	89	82	100	100	82
85	72	118	84	90	99	107	89
87	76	130	79	96	98	116	96
91	81	142	76	110	94	126	110
95	84	156	73	120	92	136	119
98	89	168	73	130	89	142	129
103	93	179	67	143	86	149	142
109	96	188	61	155	84	153	154
114	99	197	55	162	79	165	161
118	100	204	50	172	72	173	171
125	100	211	43	179	68	182	178
132	99	219	37	186	64	194	186

135	98
141	95
147	90
153	86
159	82
165	78
172	75
182	71
194	69
205	68
215	67
220	66
228	62
234	59
241	55
248	52
255	48
261	44
268	41
278	37
289	35
301	34
313	35
326	38
338	40
350	40
362	39
374	37
386	34
398	31
410	28
421	25
432	21
444	18
457	15

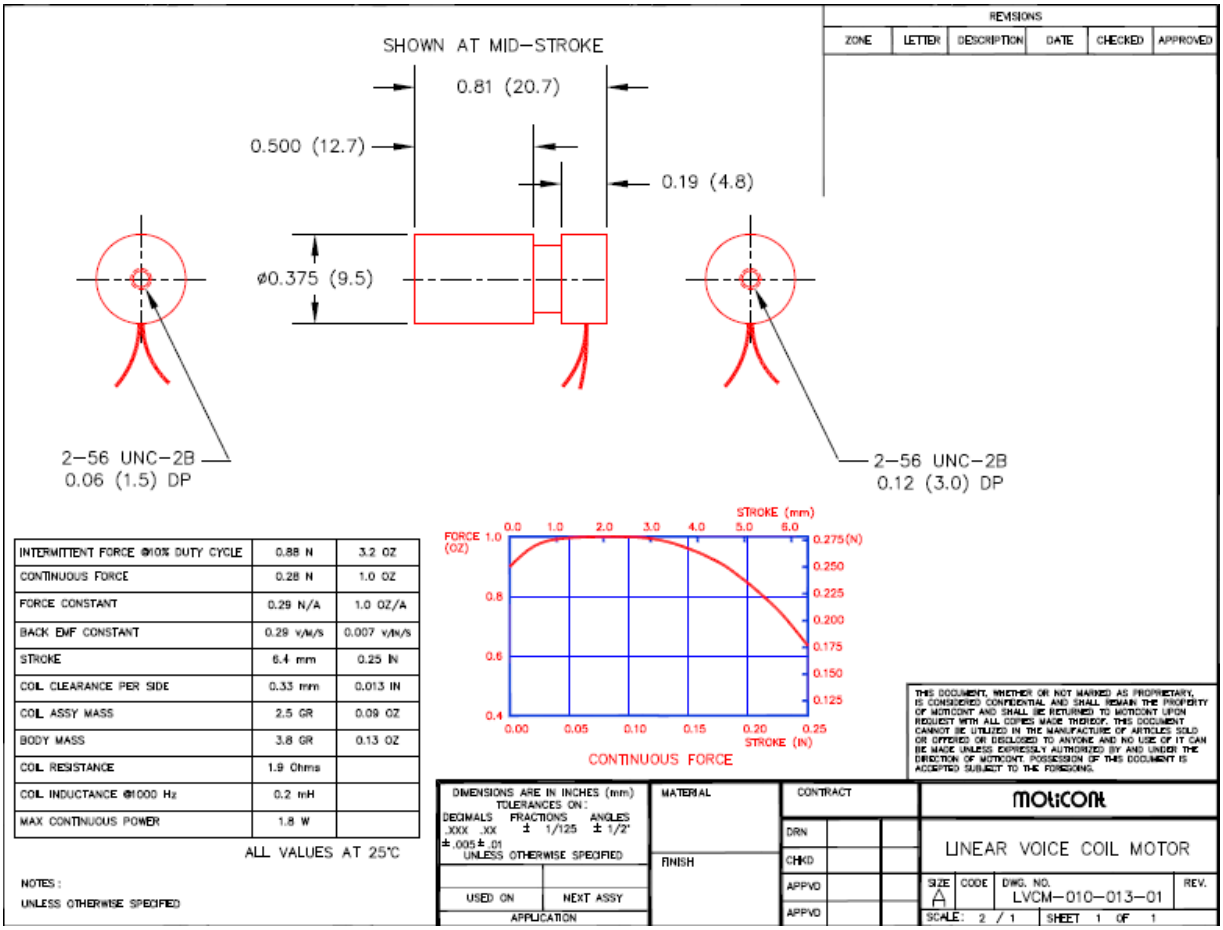
226	33
234	28
246	26
258	26
271	26
283	28
297	29
315	27
323	24
336	19
349	15
360	14
375	13
387	12
400	11
428	12
453	11
467	6
480	3
491	2
500	2

192	60
200	56
211	52
223	52
236	52
246	51
258	48
269	44
277	44
283	41
292	37
306	35
316	31
327	29
339	26
362	21
375	19
385	19
395	20
408	19
419	18
432	17
444	16
455	15
472	15
478	13
491	11
500	10

206	191
220	199
231	210
242	222
255	236
268	246
279	258
294	269
303	276
317	283
327	292
340	305
351	315
362	326
375	338
388	361
399	375
414	385
423	395
433	408
447	418
459	431
471	443
482	454
494	471
500	477

469	12
481	9
493	7
500	6

Anexo 3: Hoja de datos motor lineal de voz LVCM-010-013-01



INTERMITTENT FORCE @10% DUTY CYCLE	0.88 N	3.2 OZ
CONTINUOUS FORCE	0.28 N	1.0 OZ
FORCE CONSTANT	0.29 N/A	1.0 OZ/A
BACK EMF CONSTANT	0.29 V/M/S	0.007 V/IN/S
STROKE	6.4 mm	0.25 IN
COIL CLEARANCE PER SIDE	0.33 mm	0.013 IN
COIL ASSY MASS	2.5 GR	0.09 OZ
BODY MASS	3.8 GR	0.13 OZ
COIL RESISTANCE	1.9 Ohms	
COIL INDUCTANCE @1000 Hz	0.2 mH	
MAX CONTINUOUS POWER	1.8 W	

Anexo 4: Código empleado en funcionamiento de PWM

```
/* USER CODE BEGIN Header */
/**
```

```
******************************************************************************
        * @file           : main.c
        * @brief          : Main program body

******************************************************************************
        * @attention
        *
        * Copyright (c) 2022 STMicroelectronics.
        * All rights reserved.
        *
        * This software is licensed under terms that can be found in the LICENSE
file
        * in the root directory of this software component.
        * If no LICENSE file comes with this software, it is provided AS-IS.
        *

******************************************************************************
        */
      /* USER CODE END Header */
      /* Includes ------------------------------------------------------------------
--*/
      #include "main.h"
      /* Private includes ----------------------------------------------------------
--*/
      /* USER CODE BEGIN Includes */
      intNor[74]={0,1,2,4,6,10,14,19,22,25,29,32,35,39,43,47,51,54,56,61,64,68,72,
76,81,84,89,93,96,99,100,100,99,98,95,90,86,82,78,75,71,69,68,67,66,62,59,55,52,48
,44,41,37,35,34,35,38,40,40,39,37,34,31,28,25,21,18,15,12,9,7,6,3,0};
      intAti[54]={2,4,9,15,20,26,33,39,44,50,58,62,67,74,80,87,93,98,100,98,94,89,
84,79,76,73,73,67,61,55,50,43,37,33,28,26,26,26,28,29,27,24,19,15,14,13,12,11,12,1
1,6,3,2,2};
      intEst[66]={0,0,2,3,5,9,14,19,24,28,34,39,44,50,54,58,66,71,79,85,90,94,98,1
00,99,98,94,92,89,86,84,79,72,68,64,60,56,52,52,52,51,48,44,44,41,37,35,31,29,26,2
1,19,19,20,19,18,17,16,15,15,13,11,10,5,0,0};
      intRati[59]={1,1,5,12,18,23,28,34,42,49,54,62,70,78,84,90,96,99,100,97,90,86
,81,74,67,63,61,56,52,42,35,31,30,32,35,38,41,41,38,35,29,24,20,17,14,12,8,8,8,6,6
,4,3,0,0,0,0,0,0};
      /* USER CODE END Includes */
      /* Private typedef -----------------------------------------------------------
--*/
      /* USER CODE BEGIN PTD */
      /* USER CODE END PTD */
      /* Private define ------------------------------------------------------------
--*/
      /* USER CODE BEGIN PD */
      /* USER CODE END PD */
      /* Private macro -------------------------------------------------------------
--*/
      /* USER CODE BEGIN PM */
      /* USER CODE END PM */
      /* Private variables ---------------------------------------------------------
--*/
      TIM_HandleTypeDef htim2;
      /* USER CODE BEGIN PV */
```

```
/* USER CODE END PV */
/* Private function prototypes -----------------------------------------
--*/
void SystemClock_Config(void);
static void MX_GPIO_Init(void);
static void MX_TIM2_Init(void);
/* USER CODE BEGIN PFP */
/* USER CODE END PFP */
/* Private user code ---------------------------------------------------
--*/
/* USER CODE BEGIN 0 */
/* USER CODE END 0 */
/**
  * @brief  The application entry point.
  * @retval int
  */
int main(void)
{
  /* USER CODE BEGIN 1 */
        //uint16_t dutyCycle=10;
  /* USER CODE END 1 */
  /* MCU Configuration---------------------------------------------------
--*/
  /* Reset of all peripherals, Initializes the Flash interface and the Systick.
*/
  HAL_Init();
  /* USER CODE BEGIN Init */
  /* USER CODE END Init */
  /* Configure the system clock */
  SystemClock_Config();
  /* USER CODE BEGIN SysInit */
  /* USER CODE END SysInit */
  /* Initialize all configured peripherals */
  MX_GPIO_Init();
  MX_TIM2_Init();
  /* USER CODE BEGIN 2 */
  HAL_TIM_PWM_Start(&htim2, TIM_CHANNEL_1);
  /* USER CODE END 2 */
  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {

              for(int con=0;con<74;con++)
                     {
                   htim2.Instance->CCR1 = Nor[con];
                   HAL_Delay(10);

                     }
    /* USER CODE END WHILE */
    /* USER CODE BEGIN 3 */
        // htim2.Instance->CCR1=dutyCycle;
         //dutyCycle+=10;
         //if(dutyCycle>90) dutyCycle=10;
         //HAL_Delay(1000);
```

```
  }
  /* USER CODE END 3 */
}

/**
  * @brief System Clock Configuration
  * @retval None
  */
void SystemClock_Config(void)
{
  RCC_OscInitTypeDef RCC_OscInitStruct = {0};
  RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};
  /** Configure the main internal regulator output voltage
  */
  __HAL_RCC_PWR_CLK_ENABLE();
  __HAL_PWR_VOLTAGESCALING_CONFIG(PWR_REGULATOR_VOLTAGE_SCALE1);
  /** Initializes the RCC Oscillators according to the specified parameters
  * in the RCC_OscInitTypeDef structure.
  */
  RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSE;
  RCC_OscInitStruct.HSEState = RCC_HSE_ON;
  RCC_OscInitStruct.PLL.PLLState = RCC_PLL_ON;
  RCC_OscInitStruct.PLL.PLLSource = RCC_PLLSOURCE_HSE;
  RCC_OscInitStruct.PLL.PLLM = 4;
  RCC_OscInitStruct.PLL.PLLN = 72;
  RCC_OscInitStruct.PLL.PLLP = RCC_PLLP_DIV2;
  RCC_OscInitStruct.PLL.PLLQ = 4;
  if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
  {
    Error_Handler();
  }
  /** Initializes the CPU, AHB and APB buses clocks
  */
  RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK|RCC_CLOCKTYPE_SYSCLK
                              |RCC_CLOCKTYPE_PCLK1|RCC_CLOCKTYPE_PCLK2;
  RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_PLLCLK;
  RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
  RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV2;
  RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV1;
  if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_2) != HAL_OK)
  {
    Error_Handler();
  }
}
/**
  * @brief TIM2 Initialization Function
  * @param None
  * @retval None
  */
static void MX_TIM2_Init(void)
{
  /* USER CODE BEGIN TIM2_Init 0 */
  /* USER CODE END TIM2_Init 0 */
  TIM_ClockConfigTypeDef sClockSourceConfig = {0};
  TIM_MasterConfigTypeDef sMasterConfig = {0};
```

```
  TIM_OC_InitTypeDef sConfigOC = {0};
  /* USER CODE BEGIN TIM2_Init 1 */
  /* USER CODE END TIM2_Init 1 */
  htim2.Instance = TIM2;
  htim2.Init.Prescaler =72;
  htim2.Init.CounterMode = TIM_COUNTERMODE_UP;
  htim2.Init.Period = 100;
  htim2.Init.ClockDivision = TIM_CLOCKDIVISION_DIV1;
  htim2.Init.AutoReloadPreload = TIM_AUTORELOAD_PRELOAD_DISABLE;
  if (HAL_TIM_Base_Init(&htim2) != HAL_OK)
  {
    Error_Handler();
  }
  sClockSourceConfig.ClockSource = TIM_CLOCKSOURCE_INTERNAL;
  if (HAL_TIM_ConfigClockSource(&htim2, &sClockSourceConfig) != HAL_OK)
  {
    Error_Handler();
  }
  if (HAL_TIM_PWM_Init(&htim2) != HAL_OK)
  {
    Error_Handler();
  }
  sMasterConfig.MasterOutputTrigger = TIM_TRGO_RESET;
  sMasterConfig.MasterSlaveMode = TIM_MASTERSLAVEMODE_DISABLE;
  if   (HAL_TIMEx_MasterConfigSynchronization(&htim2,   &sMasterConfig)   !=
HAL_OK)
  {
    Error_Handler();
  }
  sConfigOC.OCMode = TIM_OCMODE_PWM1;
  sConfigOC.Pulse = 100;
  sConfigOC.OCPolarity = TIM_OCPOLARITY_HIGH;
  sConfigOC.OCFastMode = TIM_OCFAST_DISABLE;
  if (HAL_TIM_PWM_ConfigChannel(&htim2, &sConfigOC, TIM_CHANNEL_1) != HAL_OK)
  {
    Error_Handler();
  }
  /* USER CODE BEGIN TIM2_Init 2 */
  /* USER CODE END TIM2_Init 2 */
  HAL_TIM_MspPostInit(&htim2);
}
/**
  * @brief GPIO Initialization Function
  * @param None
  * @retval None
  */
static void MX_GPIO_Init(void)
{
  /* GPIO Ports Clock Enable */
  __HAL_RCC_GPIOH_CLK_ENABLE();
  __HAL_RCC_GPIOA_CLK_ENABLE();
}
/* USER CODE BEGIN 4 */
/* USER CODE END 4 */
/**
```

```
  * @brief  This function is executed in case of error occurrence.
  * @retval None
  */
void Error_Handler(void)
{
  /* USER CODE BEGIN Error_Handler_Debug */
  /* User can add his own implementation to report the HAL error return state
*/
  __disable_irq();
  while (1)
  {
  }
  /* USER CODE END Error_Handler_Debug */
}
#ifdef  USE_FULL_ASSERT
/**
  * @brief  Reports the name of the source file and the source line number
  *         where the assert_param error has occurred.
  * @param  file: pointer to the source file name
  * @param  line: assert_param error line source number
  * @retval None
  */
void assert_failed(uint8_t *file, uint32_t line)
{
  /* USER CODE BEGIN 6 */
  /* User can add his own implementation to report the file name and line
number,
     ex: printf("Wrong parameters value: file %s on line %d\r\n", file, line)
*/
  /* USER CODE END 6 */
}
#endif /* USE_FULL_ASSERT */
```

Anexo 5: Código empleando en funcionamiento de DAC

```
/* USER CODE BEGIN Header */
/**
*****************************************************************************
```

```
        * @file           : main.c
        * @brief          : Main program body

******************************************************************************
        * @attention
        *
        * Copyright (c) 2022 STMicroelectronics.
        * All rights reserved.
        *
        * This software is licensed under terms that can be found in the LICENSE
file
        * in the root directory of this software component.
        * If no LICENSE file comes with this software, it is provided AS-IS.
        *

******************************************************************************
        */
      /* USER CODE END Header */
      /* Includes ------------------------------------------------------------
--*/
      #include "main.h"
      /* Private includes ----------------------------------------------------
--*/
      /* USER CODE BEGIN Includes */
      intNor[74]={0,1,2,4,6,10,14,19,22,25,29,32,35,39,43,47,51,54,56,61,64,68,72,
76,81,84,89,93,96,99,100,100,99,98,95,90,86,82,78,75,71,69,68,67,66,62,59,55,52,48
,44,41,37,35,34,35,38,40,40,39,37,34,31,28,25,21,18,15,12,9,7,6,3,0};
      intAti[54]={2,4,9,15,20,26,33,39,44,50,58,62,67,74,80,87,93,98,100,98,94,89,
84,79,76,73,73,67,61,55,50,43,37,33,28,26,26,26,28,29,27,24,19,15,14,13,12,11,12,1
1,6,3,2,2};
      intEst[66]={0,0,2,3,5,9,14,19,24,28,34,39,44,50,54,58,66,71,79,85,90,94,98,1
00,99,98,94,92,89,86,84,79,72,68,64,60,56,52,52,52,51,48,44,44,41,37,35,31,29,26,2
1,19,19,20,19,18,17,16,15,15,13,11,10,5,0,0};
      intRati[59]={1,1,5,12,18,23,28,34,42,49,54,62,70,78,84,90,96,99,100,97,90,86
,81,74,67,63,61,56,52,42,35,31,30,32,35,38,41,41,38,35,29,24,20,17,14,12,8,8,8,6,6
,4,3,0,0,0,0,0,0};
      /* USER CODE END Includes */
      /* Private typedef -----------------------------------------------------
--*/
      /* USER CODE BEGIN PTD */
      /* USER CODE END PTD */
      /* Private define ------------------------------------------------------
--*/
      /* USER CODE BEGIN PD */
      /* USER CODE END PD */
      /* Private macro -------------------------------------------------------
--*/
      /* USER CODE BEGIN PM */
      /* USER CODE END PM */
      /* Private variables ---------------------------------------------------
--*/
      DAC_HandleTypeDef hdac;
      /* USER CODE BEGIN PV */
      /* USER CODE END PV */
```

```
      /* Private function prototypes -----------------------------------------
--*/
      void SystemClock_Config(void);
      static void MX_GPIO_Init(void);
      static void MX_DAC_Init(void);
      /* USER CODE BEGIN PFP */
      /* USER CODE END PFP */
      /* Private user code ---------------------------------------------------
--*/
      /* USER CODE BEGIN 0 */
      /* USER CODE END 0 */
      /**
        * @brief  The application entry point.
        * @retval int
        */
      int main(void)
      {
        /* USER CODE BEGIN 1 */
        /* USER CODE END 1 */
        /*  MCU Configuration--------------------------------------------------
--*/
        /* Reset of all peripherals, Initializes the Flash interface and the Systick.
*/
        HAL_Init();
        /* USER CODE BEGIN Init */
        /* USER CODE END Init */
        /* Configure the system clock */
        SystemClock_Config();
        /* USER CODE BEGIN SysInit */
        /* USER CODE END SysInit */
        /* Initialize all configured peripherals */
        MX_GPIO_Init();
        MX_DAC_Init();
        /* USER CODE BEGIN 2 */
        HAL_DAC_Start(&hdac,DAC_CHANNEL_1);
        /* USER CODE END 2 */
        /* Infinite loop */
        /* USER CODE BEGIN WHILE */
        while (1)
        {
          /* USER CODE END WHILE */
          /* USER CODE BEGIN 3 */
              for(int con=0;con<74;con++)
              {

HAL_DAC_SetValue(&hdac,DAC_CHANNEL_1,DAC_ALIGN_12B_R,Nor[con]*24);
                      HAL_Delay(30);
              }
            }
        /* USER CODE END 3 */
      }
      /**
        * @brief System Clock Configuration
        * @retval None
        */
```

```c
void SystemClock_Config(void)
{
  RCC_OscInitTypeDef RCC_OscInitStruct = {0};
  RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};
  /** Configure the main internal regulator output voltage
  */
  __HAL_RCC_PWR_CLK_ENABLE();
  __HAL_PWR_VOLTAGESCALING_CONFIG(PWR_REGULATOR_VOLTAGE_SCALE1);
  /** Initializes the RCC Oscillators according to the specified parameters
  * in the RCC_OscInitTypeDef structure.
  */
  RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSI;
  RCC_OscInitStruct.HSIState = RCC_HSI_ON;
  RCC_OscInitStruct.HSICalibrationValue = RCC_HSICALIBRATION_DEFAULT;
  RCC_OscInitStruct.PLL.PLLState = RCC_PLL_ON;
  RCC_OscInitStruct.PLL.PLLSource = RCC_PLLSOURCE_HSI;
  RCC_OscInitStruct.PLL.PLLM = 8;
  RCC_OscInitStruct.PLL.PLLN = 72;
  RCC_OscInitStruct.PLL.PLLP = RCC_PLLP_DIV2;
  RCC_OscInitStruct.PLL.PLLQ = 4;
  if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
  {
    Error_Handler();
  }
  /** Initializes the CPU, AHB and APB buses clocks
  */
  RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK|RCC_CLOCKTYPE_SYSCLK
                              |RCC_CLOCKTYPE_PCLK1|RCC_CLOCKTYPE_PCLK2;
  RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_PLLCLK;
  RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
  RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV2;
  RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV1;
  if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_2) != HAL_OK)
  {
    Error_Handler();
  }
}
/**
  * @brief DAC Initialization Function
  * @param None
  * @retval None
  */
static void MX_DAC_Init(void)
{
  /* USER CODE BEGIN DAC_Init 0 */
  /* USER CODE END DAC_Init 0 */
  DAC_ChannelConfTypeDef sConfig = {0};
  /* USER CODE BEGIN DAC_Init 1 */
  /* USER CODE END DAC_Init 1 */
  /** DAC Initialization
  */
  hdac.Instance = DAC;
  if (HAL_DAC_Init(&hdac) != HAL_OK)
  {
    Error_Handler();
```

```
  }
  /** DAC channel OUT1 config
  */
  sConfig.DAC_Trigger = DAC_TRIGGER_NONE;
  sConfig.DAC_OutputBuffer = DAC_OUTPUTBUFFER_ENABLE;
  if (HAL_DAC_ConfigChannel(&hdac, &sConfig, DAC_CHANNEL_1) != HAL_OK)
  {
    Error_Handler();
  }
  /* USER CODE BEGIN DAC_Init 2 */
  /* USER CODE END DAC_Init 2 */
}
/**
  * @brief GPIO Initialization Function
  * @param None
  * @retval None
  */
static void MX_GPIO_Init(void)
{
  /* GPIO Ports Clock Enable */
  __HAL_RCC_GPIOH_CLK_ENABLE();
  __HAL_RCC_GPIOA_CLK_ENABLE();
}
/* USER CODE BEGIN 4 */
/* USER CODE END 4 */
/**
  * @brief  This function is executed in case of error occurrence.
  * @retval None
  */
void Error_Handler(void)
{
  /* USER CODE BEGIN Error_Handler_Debug */
  /* User can add his own implementation to report the HAL error return state
*/
  __disable_irq();
  while (1)
  {
  }
  /* USER CODE END Error_Handler_Debug */
}
#ifdef  USE_FULL_ASSERT
/**
  * @brief  Reports the name of the source file and the source line number
  *         where the assert_param error has occurred.
  * @param  file: pointer to the source file name
  * @param  line: assert_param error line source number
  * @retval None
  */
void assert_failed(uint8_t *file, uint32_t line)
{
  /* USER CODE BEGIN 6 */
  /* User can add his own implementation to report the file name and line
number,
     ex: printf("Wrong parameters value: file %s on line %d\r\n", file, line)
*/
```

```
  /* USER CODE END 6 */
}
#endif /* USE_FULL_ASSERT */
```

Anexo 6: Librería ST7735.h

```
#ifndef __ST7735_H__
#define __ST7735_H__

#include "fonts.h"
```

```c
#include "stm32f4xx_hal.h"
#include <stdbool.h>

extern SPI_HandleTypeDef hspi1;
#define ST7735_SPI_PORT hspi1

/****** PIN DEFINES ******/
#define CS_PORT GPIOB
#define CS_PIN  GPIO_PIN_6
#define DC_PORT GPIOA
#define DC_PIN  GPIO_PIN_8
#define RST_PORT GPIOC
#define RST_PIN  GPIO_PIN_7

/****** TFT DEFINES ******/
//#define ST7735_IS_160X80 1
#define ST7735_IS_128X128 1
//#define ST7735_IS_160X128 1
#define ST7735_WIDTH  128
#define ST7735_HEIGHT 128

#define DELAY 0x80

#define ST7735_MADCTL_MY  0x80
#define ST7735_MADCTL_MX  0x40
#define ST7735_MADCTL_MV  0x20
#define ST7735_MADCTL_ML  0x10
#define ST7735_MADCTL_RGB 0x00
#define ST7735_MADCTL_BGR 0x08
#define ST7735_MADCTL_MH  0x04

#define ST7735_NOP     0x00
#define ST7735_SWRESET 0x01
#define ST7735_RDDID   0x04
#define ST7735_RDDST   0x09

#define ST7735_SLPIN   0x10
#define ST7735_SLPOUT  0x11
#define ST7735_PTLON   0x12
#define ST7735_NORON   0x13

#define ST7735_INVOFF  0x20
#define ST7735_INVON   0x21
#define ST7735_DISPOFF 0x28
#define ST7735_DISPON  0x29
#define ST7735_CASET   0x2A
#define ST7735_RASET   0x2B
#define ST7735_RAMWR   0x2C
#define ST7735_RAMRD   0x2E

#define ST7735_PTLAR   0x30
#define ST7735_COLMOD  0x3A
#define ST7735_MADCTL  0x36

#define ST7735_FRMCTR1 0xB1
```

```
#define ST7735_FRMCTR2 0xB2
#define ST7735_FRMCTR3 0xB3
#define ST7735_INVCTR  0xB4
#define ST7735_DISSET5 0xB6

#define ST7735_PWCTR1  0xC0
#define ST7735_PWCTR2  0xC1
#define ST7735_PWCTR3  0xC2
#define ST7735_PWCTR4  0xC3
#define ST7735_PWCTR5  0xC4
#define ST7735_VMCTR1  0xC5

#define ST7735_RDID1   0xDA
#define ST7735_RDID2   0xDB
#define ST7735_RDID3   0xDC
#define ST7735_RDID4   0xDD

#define ST7735_PWCTR6  0xFC

#define ST7735_GMCTRP1 0xE0
#define ST7735_GMCTRN1 0xE1

// Color definitions
#define      BLACK   0x0000
#define      BLUE    0x001F
#define      RED     0xF800
#define      GREEN   0x07E0
#define CYAN    0x07FF
#define MAGENTA 0xF81F
#define YELLOW  0xFFE0
#define WHITE   0xFFFF
#define color565(r, g, b) (((r & 0xF8) << 8) | ((g & 0xFC) << 3) | ((b & 0xF8)
>> 3))

// call before initializing any SPI devices
void ST7735_Unselect();

void ST7735_Init(uint8_t rotation);
void ST7735_SetRotation(uint8_t m);
void ST7735_DrawPixel(uint16_t x, uint16_t y, uint16_t color);
void ST7735_WriteString(uint16_t x, uint16_t y, const char* str, FontDef font,
uint16_t color, uint16_t bgcolor);
void ST7735_FillRectangle(uint16_t x, uint16_t y, uint16_t w, uint16_t h,
uint16_t color);
void ST7735_FillScreen(uint16_t color);
void ST7735_DrawImage(uint16_t x, uint16_t y, uint16_t w, uint16_t h, const
uint16_t* data);
void ST7735_InvertColors(bool invert);

int16_t _width;       ///< Display width as modified by current rotation
int16_t _height;      ///< Display height as modified by current rotation
int16_t cursor_x;     ///< x location to start print()ing text
int16_t cursor_y;     ///< y location to start print()ing text
uint8_t rotation;     ///< Display rotation (0 thru 3)
uint8_t _colstart;   ///< Some displays need this changed to offset
```

```
uint8_t _rowstart;       ///< Some displays need this changed to offset
uint8_t _xstart;
uint8_t _ystart;
#endif // __ST7735_H__
```

Además se debe tener en cuenta agregar la siguiente librería complementaria

```
/* vim: set ai et ts=4 sw=4: */
#ifndef __FONTS_H__
#define __FONTS_H__

#include <stdint.h>

typedef struct {
    const uint8_t width;
    uint8_t height;
    const uint16_t *data;
} FontDef;

extern FontDef Font_7x10;
extern FontDef Font_11x18;
extern FontDef Font_16x26;

#endif // __FONTS_H__
```

Anexo 7: Librería GFX_FUNCTIONS.h

```
/*
 * GFX_FUNCTIONS.h
 *  Created on: 30-Oct-2020
 *      Author: meh
 */
```

```
#ifndef INC_GFX_FUNCTIONS_H_
#define INC_GFX_FUNCTIONS_H_

void drawPixel(int16_t x, int16_t y, uint16_t color);
void writeLine(int16_t x0, int16_t y0, int16_t x1, int16_t y1, uint16_t
color);
void drawFastVLine(int16_t x, int16_t y, int16_t h, uint16_t color);
void drawFastHLine(int16_t x, int16_t y, int16_t w, uint16_t color);
void fillRect(int16_t x, int16_t y, int16_t w, int16_t h, uint16_t color);
void drawLine(int16_t x0, int16_t y0, int16_t x1, int16_t y1, uint16_t color);
void drawCircle(int16_t x0, int16_t y0, int16_t r, uint16_t color);
void drawCircleHelper( int16_t x0, int16_t y0, int16_t r, uint8_t cornername,
uint16_t color);
void fillCircleHelper(int16_t x0, int16_t y0, int16_t r, uint8_t corners,
int16_t delta, uint16_t color);
void fillCircle(int16_t x0, int16_t y0, int16_t r, uint16_t color);
void drawRect(int16_t x, int16_t y, int16_t w, int16_t h, uint16_t color);
void drawRoundRect(int16_t x, int16_t y, int16_t w, int16_t h, int16_t r,
uint16_t color);
void fillRoundRect(int16_t x, int16_t y, int16_t w, int16_t h, int16_t r,
uint16_t color);
void drawTriangle(int16_t x0, int16_t y0, int16_t x1, int16_t y1, int16_t x2,
int16_t y2, uint16_t color);
void fillTriangle(int16_t x0, int16_t y0, int16_t x1, int16_t y1, int16_t x2,
int16_t y2, uint16_t color);
void fillScreen(uint16_t color);
void testLines(uint16_t color);
void testFastLines(uint16_t color1, uint16_t color2);
void testRects(uint16_t color) ;
void testFilledRects(uint16_t color1, uint16_t color2);
void testFilledCircles(uint8_t radius, uint16_t color);
void testCircles(uint8_t radius, uint16_t color);
void testTriangles();
void testFilledTriangles();
void testRoundRects();
void testFilledRoundRects();
void testFillScreen();
void testAll (void);

#endif /* INC_GFX_FUNCTIONS_H_ */
```

Anexo 8: Código empleado en funcionamiento de Display

```
/* USER CODE BEGIN Header */
/**

*****************************************************************************
  * @file           : main.c
  * @brief          : Main program body
```

```
******************************************************************************
        * @attention
        *
        * Copyright (c) 2022 STMicroelectronics.
        * All rights reserved.
        *
        * This software is licensed under terms that can be found in the LICENSE
file
        * in the root directory of this software component.
        * If no LICENSE file comes with this software, it is provided AS-IS.
        *

******************************************************************************
        */
      /* USER CODE END Header */
      /* Includes ------------------------------------------------------------
--*/
      #include "main.h"
      /* Private includes ----------------------------------------------------
--*/
      /* USER CODE BEGIN Includes */
      #include "ST7735.h"
      #include "GFX_FUNCTIONS.h"
      /* USER CODE END Includes */
      /* Private typedef -----------------------------------------------------
--*/
      /* USER CODE BEGIN PTD */
      //Cadena de datos
      intNor[74]={0,1,2,4,6,10,14,19,22,25,29,32,35,39,43,47,51,54,56,61,64,68,72,
76,81,84,89,93,96,99,100,100,99,98,95,90,86,82,78,75,71,69,68,67,66,62,59,55,52,48
,44,41,37,35,34,35,38,40,40,39,37,34,31,28,25,21,18,15,12,9,7,6,3,0};
      int cam=0;
      intAti[54]={2,4,9,15,20,26,33,39,44,50,58,62,67,74,80,87,93,98,100,98,94,89,
84,79,76,73,73,67,61,55,50,43,37,33,28,26,26,26,28,29,27,24,19,15,14,13,12,11,12,1
1,6,3,2,2;
      intEst[66]={0,0,2,3,5,9,14,19,24,28,34,39,44,50,54,58,66,71,79,85,90,94,98,1
00,99,98,94,92,89,86,84,79,72,68,64,60,56,52,52,52,51,48,44,44,41,37,35,31,29,26,2
1,19,19,20,19,18,17,16,15,15,13,11,10,5,0,0};
      intRati[59]={1,1,5,12,18,23,28,34,42,49,54,62,70,78,84,90,96,99,100,97,90,86
,81,74,67,63,61,56,52,42,35,31,30,32,35,38,41,41,38,35,29,24,20,17,14,12,8,8,8,6,6
,4,3,0,0,0,0,0,0};
      /* USER CODE END PTD */
      /* Private define ------------------------------------------------------
--*/
      /* USER CODE BEGIN PD */
      /* USER CODE END PD */
      /* Private macro -------------------------------------------------------
--*/
      /* USER CODE BEGIN PM */
      /* USER CODE END PM */
      /* Private variables ---------------------------------------------------
--*/
      SPI_HandleTypeDef hspi1;
      /* USER CODE BEGIN PV */
```

```
      /* USER CODE END PV */
      /* Private function prototypes -----------------------------------------
--*/
      void SystemClock_Config(void);
      static void MX_GPIO_Init(void);
      static void MX_SPI1_Init(void);
      /* USER CODE BEGIN PFP */
      /* USER CODE END PFP */
      /* Private user code ---------------------------------------------------
--*/
      /* USER CODE BEGIN 0 */
      /* USER CODE END 0 */
      /**
        * @brief  The application entry point.
        * @retval int
        */
      int main(void)
      {
        /* USER CODE BEGIN 1 */
        /* USER CODE END 1 */
        /* MCU Configuration--------------------------------------------------
--*/
        /* Reset of all peripherals, Initializes the Flash interface and the Systick.
*/
        HAL_Init();
        /* USER CODE BEGIN Init */
        /* USER CODE END Init */
        /* Configure the system clock */
        SystemClock_Config();

        /* USER CODE BEGIN SysInit */
        /* USER CODE END SysInit */
        /* Initialize all configured peripherals */
        MX_GPIO_Init();
        MX_SPI1_Init();
        /* USER CODE BEGIN 2 */
          ST7735_Init(0);
          fillScreen(BLACK);
          //testAll();
        /* USER CODE END 2 */
        /* Infinite loop */
        /* USER CODE BEGIN WHILE */
        while (1)
        {
              ST7735_SetRotation(2);
              for (int s=0;s<2;s++){
                for(int  con=0;con<74;con++)    //Modificar  Con<xx  dependiendo  la
cadena de datos
                {
                  ST7735_DrawPixel(Nor[con],con+cam,WHITE);   //Modificar Nor[con]
segun la cadena de datos
                }
              cam+=74;
              if(cam==148){
                    cam=0;
```

```
        }
            /* USER CODE END WHILE */

            /* USER CODE BEGIN 3 */
              //fillScreen(BLACK);
              //HAL_Delay(5);
        }
    /* USER CODE END WHILE */
    /* USER CODE BEGIN 3 */
  }
  /* USER CODE END 3 */
}
/**
  * @brief System Clock Configuration
  * @retval None
  */
void SystemClock_Config(void)
{
  RCC_OscInitTypeDef RCC_OscInitStruct = {0};
  RCC_ClkInitTypeDef RCC_ClkInitStruct = {0};
  /** Configure the main internal regulator output voltage
  */
  __HAL_RCC_PWR_CLK_ENABLE();
  __HAL_PWR_VOLTAGESCALING_CONFIG(PWR_REGULATOR_VOLTAGE_SCALE1);
  /** Initializes the RCC Oscillators according to the specified parameters
  * in the RCC_OscInitTypeDef structure.
  */
  RCC_OscInitStruct.OscillatorType = RCC_OSCILLATORTYPE_HSE;
  RCC_OscInitStruct.HSEState = RCC_HSE_ON;
  RCC_OscInitStruct.PLL.PLLState = RCC_PLL_ON;
  RCC_OscInitStruct.PLL.PLLSource = RCC_PLLSOURCE_HSE;
  RCC_OscInitStruct.PLL.PLLM = 4;
  RCC_OscInitStruct.PLL.PLLN = 168;
  RCC_OscInitStruct.PLL.PLLP = RCC_PLLP_DIV2;
  RCC_OscInitStruct.PLL.PLLQ = 4;
  if (HAL_RCC_OscConfig(&RCC_OscInitStruct) != HAL_OK)
  {
    Error_Handler();
  }
  /** Initializes the CPU, AHB and APB buses clocks
  */
  RCC_ClkInitStruct.ClockType = RCC_CLOCKTYPE_HCLK|RCC_CLOCKTYPE_SYSCLK
                              |RCC_CLOCKTYPE_PCLK1|RCC_CLOCKTYPE_PCLK2;
  RCC_ClkInitStruct.SYSCLKSource = RCC_SYSCLKSOURCE_PLLCLK;
  RCC_ClkInitStruct.AHBCLKDivider = RCC_SYSCLK_DIV1;
  RCC_ClkInitStruct.APB1CLKDivider = RCC_HCLK_DIV4;
  RCC_ClkInitStruct.APB2CLKDivider = RCC_HCLK_DIV2;
  if (HAL_RCC_ClockConfig(&RCC_ClkInitStruct, FLASH_LATENCY_5) != HAL_OK)
  {
    Error_Handler();
  }
}
/**
  * @brief SPI1 Initialization Function
  * @param None
```

```
  * @retval None
  */
static void MX_SPI1_Init(void)
{
  /* USER CODE BEGIN SPI1_Init 0 */

  /* USER CODE END SPI1_Init 0 */

  /* USER CODE BEGIN SPI1_Init 1 */
  /* USER CODE END SPI1_Init 1 */
  /* SPI1 parameter configuration*/
  hspi1.Instance = SPI1;
  hspi1.Init.Mode = SPI_MODE_MASTER;
  hspi1.Init.Direction = SPI_DIRECTION_1LINE;
  hspi1.Init.DataSize = SPI_DATASIZE_8BIT;
  hspi1.Init.CLKPolarity = SPI_POLARITY_LOW;
  hspi1.Init.CLKPhase = SPI_PHASE_1EDGE;
  hspi1.Init.NSS = SPI_NSS_SOFT;
  hspi1.Init.BaudRatePrescaler = SPI_BAUDRATEPRESCALER_32;
  hspi1.Init.FirstBit = SPI_FIRSTBIT_MSB;
  hspi1.Init.TIMode = SPI_TIMODE_DISABLE;
  hspi1.Init.CRCCalculation = SPI_CRCCALCULATION_DISABLE;
  hspi1.Init.CRCPolynomial = 10;
  if (HAL_SPI_Init(&hspi1) != HAL_OK)
  {
    Error_Handler();
  }
  /* USER CODE BEGIN SPI1_Init 2 */

  /* USER CODE END SPI1_Init 2 */
}
/**
  * @brief GPIO Initialization Function
  * @param None
  * @retval None
  */
static void MX_GPIO_Init(void)
{
  GPIO_InitTypeDef GPIO_InitStruct = {0};
  /* GPIO Ports Clock Enable */
  __HAL_RCC_GPIOH_CLK_ENABLE();
  __HAL_RCC_GPIOA_CLK_ENABLE();
  __HAL_RCC_GPIOC_CLK_ENABLE();
  __HAL_RCC_GPIOB_CLK_ENABLE();
  /*Configure GPIO pin Output Level */
  HAL_GPIO_WritePin(GPIOC, GPIO_PIN_7, GPIO_PIN_RESET);
  /*Configure GPIO pin Output Level */
  HAL_GPIO_WritePin(GPIOA, GPIO_PIN_8, GPIO_PIN_RESET);
  /*Configure GPIO pin Output Level */
  HAL_GPIO_WritePin(GPIOB, GPIO_PIN_6, GPIO_PIN_RESET);
  /*Configure GPIO pin : PC7 */
  GPIO_InitStruct.Pin = GPIO_PIN_7;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
```

```
  HAL_GPIO_Init(GPIOC, &GPIO_InitStruct);
  /*Configure GPIO pin : PA8 */
  GPIO_InitStruct.Pin = GPIO_PIN_8;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
  HAL_GPIO_Init(GPIOA, &GPIO_InitStruct);
  /*Configure GPIO pin : PB6 */
  GPIO_InitStruct.Pin = GPIO_PIN_6;
  GPIO_InitStruct.Mode = GPIO_MODE_OUTPUT_PP;
  GPIO_InitStruct.Pull = GPIO_NOPULL;
  GPIO_InitStruct.Speed = GPIO_SPEED_FREQ_LOW;
  HAL_GPIO_Init(GPIOB, &GPIO_InitStruct);
}
/* USER CODE BEGIN 4 */
/* USER CODE END 4 */
/**
  * @brief  This function is executed in case of error occurrence.
  * @retval None
  */
void Error_Handler(void)
{
  /* USER CODE BEGIN Error_Handler_Debug */
  /* User can add his own implementation to report the HAL error return state
*/
  __disable_irq();
  while (1)
  {
  }
  /* USER CODE END Error_Handler_Debug */
}
#ifdef  USE_FULL_ASSERT
/**
  * @brief  Reports the name of the source file and the source line number
  *         where the assert_param error has occurred.
  * @param  file: pointer to the source file name
  * @param  line: assert_param error line source number
  * @retval None
  */
void assert_failed(uint8_t *file, uint32_t line)
{
  /* USER CODE BEGIN 6 */
  /* User can add his own implementation to report the file name and line
number,
     ex: printf("Wrong parameters value: file %s on line %d\r\n", file, line)
*/
  /* USER CODE END 6 */
}
#endif /* USE_FULL_ASSERT */
```