



EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA



TECNOLÓGICO
NACIONAL DE MÉXICO®

Instituto Tecnológico de Apizaco

TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE APIZACO

**TEST BENCH DEVELOPMENT FOR QUERETARO
ENGINEERING TECH CENTER**

TITULACIÓN INTEGRAL INFORME DE RESIDENCIA PROFESIONAL

**PARA OBTENER EL TITULO PROFESIONAL DE:
INGENIERO EN ELECTRÓNICA**

**PRESENTA:
AZUCENA TLACOMULCO ZARAGOZA**

**NOMBRE DEL ASESOR:
DR. CARLOS MANUEL ORTIZ LIMA**



APIZACO, TLAX., MAYO 2025



EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA



TECNOLÓGICO
NACIONAL DE MÉXICO

Instituto Tecnológico de Apizaco
Electrónica

Apizaco, Tlaxcala, **19/agosto/2024**
Departamento: Ingeniería Eléctrica y Electrónica
No. De oficio: I.E.E. /768/2024

Asunto: Liberación de Proyecto
para la Titulación Integral

CÉSAR REYNALDO RAMOS GÓMEZ
JEFE DE LA DIVISIÓN DE ESTUDIOS PROFESIONALES
PRESENTE

Por este medio le informo que ha sido liberado el siguiente proyecto para la Titulación Integral:

Nombre del egresado:	AZUCENA TLACOMULCO ZARAGOZA
Carrera:	INGENIERÍA ELECTRÓNICA
No. De Control:	19371413
Nombre del proyecto:	"TEST BENCH DEVELOPMENT FOR QUERETARO ENGINEERING TECH CENTER"
Producto	TITULACIÓN INTEGRAL INFORME DE RESIDENCIA PROFESIONAL

Agradezco de antemano su valioso apoyo en esta importante actividad para la formación profesional de nuestros egresados.

ATENTAMENTE

Excelencia en Educación Tecnológica®
Pensar para Servir, Servir para Triunfar®



SECRETARÍA DE EDUCACIÓN PÚBLICA
TECNOLÓGICO NACIONAL
DE MÉXICO
INSTITUTO TECNOLÓGICO DE APIZACO
DEPARTAMENTO DE
INGENIERÍA ELÉCTRICA
Y ELECTRÓNICA

JAIME ALBERTO ELIZALDE OLIN
JEFE DEL DEPTO. DE INGENIERÍA
ELÉCTRICA Y ELECTRÓNICA


CARLOS MANUEL ORTIZ LIMA
Nombre y firma del asesor

JAEO/Isa*
C.p. Archivo



Av. Instituto Tecnológico No. 418, San Andrés Ahuashuatepec, Municipio de Tzompantepec,
Tlaxcala, Mex. C.P. 90491 Tel. (241) 4172010 Ext. 145,
e-mail: electronica@apizaco.tecnm.mx | apizaco.tecnm.mx



2023
Francisco
VILLA



Tzompantepec, Tlaxcala, 13/mayo/2025

AZUCENA TLACOMULCO ZARAGOZA
EGRESADO(A) DE LA CARRERA DE
INGENIERÍA ELECTRÓNICA.
CON NÚMERO DE CONTROL 19371413.
PRESENTE.

Por este medio, me permito informar a usted que por aprobación de la comisión revisora asignada para valorar el trabajo que mediante la opción TITULACIÓN INTEGRAL INFORME DE RESIDENCIA PROFESIONAL y cuyo tema es: TEST BENCH DEVELOPMENT FOR QUERETARO ENGINEERING TECH CENTER, conforme a lo establecido en el procedimiento para la obtención del Título Profesional en el Instituto Tecnológico, la División de Estudios Profesionales a mi cargo le otorga la.

AUTORIZACIÓN DE IMPRESIÓN

Debiendo entregar 2 USB del mismo, apropiadamente empastados, a la brevedad, para presentar su Acto de Recepción Profesional.

Sin otro particular por el momento me despido.

ATENTAMENTE

Excelencia en Educación Tecnológica®
Pensar para Servir, Servir para Triunfar®

SECRETARÍA DE EDUCACIÓN PÚBLICA
TECNOLÓGICO NACIONAL
DE MÉXICO
INSTITUTO TECNOLÓGICO DE APIZACO

DIVISIÓN DE
ESTUDIOS PROFESIONALES

ISRAEL JORGE SÁNCHEZ SÁNCHEZ
JEFE DE LA DIVISIÓN DE ESTUDIOS PROFESIONALES

ccp. Archivo
IJSS/Gsc*



2025
Año de
La Mujer
Indígena

Av. Instituto Tecnológico No. 418, San Andrés Ahuashuatepec,
Municipio de Tzompantepec, Tlaxcala, Mex. C.P. 90491 Tel. (241) 4172010
Ext. 304, e-mail: dep_titulacion@apizaco.tecnm.mx | apizaco.tecnm.mx



AGRADECIMIENTOS

Es justo dejar una nota de agradecimiento a todas las personas y organizaciones que hicieron posible el desarrollo de este proyecto, quisiera comenzar agradeciendo...

Al Ingeniero Oscar Ramirez & al Ingeniero Gabriel Ivan Trejo por su labor como asesores durante este tiempo, quienes demostraron no solo un gran conocimiento y experiencia, sino también una comprensión y empatía sin las que el trabajo no hubiera sido posible, de quienes he aprendido mucho, sin su duda su guía fue indispensable para mí.

Al Dr. Carlos Manuel Ortiz Lima, quien fungió como mi asesor interno, su apoyo, guía y comprensión fueron indispensables para la culminación exitosa de este Proyecto.

Al departamento de R&D de BCS-Automotive interface solutions, quienes confiaron en mi potencial dándome la oportunidad de desarrollar este proyecto dentro de sus instalaciones y con todo el apoyo necesario.

Un especial reconocimiento al equipo de software del que fui parte, gracias por su ayuda incondicional y palabras de apoyo, compartir con ustedes el desarrollo de este proyecto ha sido el primer paso en mi desarrollo profesional.

Finalmente quisiera agradecer a mi familia y pareja, por su apoyo incondicional en los momentos más difíciles, por todas las palabras de apoyo y el amor con el que me impulsaron siempre. Gracias por confiar en mi y por consentirme siempre en mis periodos de estrés.

Este trabajo está dedicado a mi gato, que me acompañó durante todas las noches de desvelo, los días de estrés, los días de estudio & los momentos de alegría. Donde quiera que estes, tu recuerdo siempre me impulsa y da fuerza para seguir adelante.

ÍNDICE GENERAL

AGRADECIMIENTOS.....	4
RESUMEN.....	14
CAPITULO I	15
GENERALIDADES DEL PROYECTO.....	15
INTRODUCCIÓN	16
DATOS DE LA EMPRESA	17
BCS-AIS Ensamblés Mexico.....	17
BCS Automotive Interface Solutions planta 1 en Querétaro	19
BCS Automotive Interface Solutions planta 2 en Querétaro	19
Departamento de R&D.....	20
Asesores	21
Lugar de realización del proyecto	22
Descripción del puesto desarrollado: Software Intern	22
PLANTEAMIENTO DEL PROBLEMA.....	23
OBJETIVOS	24
Objetivo General	24
Objetivos Específicos.....	24
JUSTIFICACIÓN.....	25
DELIMITACIÓN.....	25
CAPITULO II	26
MARCO TEÓRICO.....	26
INDUSTRIA AUTOMOTRIZ EN MÉXICO.....	27
TECNOLOGÍA HÁPTICA.....	28
Antecedentes	28
TEORÍA DE VIBRACIONES	32
Vibración.....	32
Clasificaciones de las vibraciones.....	35
Vibración Libre Amortiguada	36
Unidades de medición de vibración	37

ACTUADORES HÁPTICOS	38
Opciones de retroalimentación háptica	38
El Actuador Resonante Lineal (LRA)	40
Resonancia y Frecuencia Natural	41
Funcionamiento	42
Componentes de los actuadores de resonancia lineal.....	43
Motor de masa giratoria excéntrica (ERM).....	45
Funcionamiento	46
Diferencias entre LRAS Y ERMS	48
TEXAS INSTRUMENTS	49
PROTOCOLOS DE COMUNICACIÓN.....	50
I2c Interface	50
Transferencias de un byte y de múltiples bytes.....	52
UART	52
PYTHON	53
QT DESIGNER.....	53
VISUAL STUDIO	54
CAPITULO III.....	55
DESARROLLO DEL PROYECTO.....	55
DESCRIPCIÓN GENERAL DEL PROYECTO	56
Propuesta de diagrama general del proyecto	56
Definición y selección de materiales (Hardware)	57
Actuadores Hápticos	57
Descripción del funcionamiento del driver	58
Driver	59
Microcontrolador	61
Alimentación	62
Pinout	62
Definición y selección de herramientas (Software).....	63
DESARROLLO DE CASOS DE PRUEBA PRELIMINARES	63
Validación del funcionamiento del LRA.....	63
Desarrollo de código para casos de prueba de los efectos hápticos	66
Modo Waveform Sequencer.....	66

DESARROLLO DE SOFTWARE	68
Desarrollo del código para el microcontrolador	68
Inicialización del dispositivo	69
Modo de espera STAND BY	70
Acceso al mapa de registros I2C	70
Tensión nominal y de sobremarcha	71
ERM – Tensión nominal y de sobremarcha	71
LRA - Tensiones nominales y de sobremarcha	72
Ajuste de los registros de control	73
Registros de control de ERM	73
Registros de Control LRA	73
Selección de la biblioteca de formas de onda	74
Propuesta de parámetros para la programación de la inicialización de los actuadores	75
Autocalibración	78
Programación de secuenciador de formas de onda	84
Resumen de la programación de driver	86
Código de programación para el driver	86
Desarrollo de código para la HMI e integración de funcionalidades en Visual Studio	87
CAPITULO IV RESULTADOS Y CONCLUSIONES	90
RESULTADOS	91
Sección I Agrupación De Formas De Onda	91
Sección II Diagramas De Flujo Para La Programación	92
Inicialización	92
Autocalibración	94
Secuenciador de Formas de Onda	98
Sección III Resultado de la Interfaz prototipo	100
CONCLUSIONES	101
CAPITULO V	102
COMPETENCIAS DESARROLLADAS	102
COMPETENCIAS OBTENIDAS	103
Referencias	105
ANEXOS	108
Anexo A	108
Código de prueba para LRA	108
Anexo B	117
Código para casos de prueba de los efectos hápticos	117

Anexo C	119
Código para el microcontrolador	119
Anexo D	125
Código en visual studio	125
Anexo1	162
Diagrama de conexiones	162
Anexo 2	163
Data sheet actuador LRA	163
Anexo 3	168
Data sheet actuador ERM.....	168
Anexo 4	172
Registros de memoria DRV26xx	172

INDICE DE IMÁGENES

Figura 1 Cifras clave. Propiedad Confidencial © BCS Automotive Interface Solutions 2021.	17
Figura 2 Oferta de soluciones Automotrices. Propiedad confidencial © BCS Automotive Interface Solutions 2021.....	17
Figura 3 Presencia mundial de ingeniería. Propiedad confidencial © BCS Automotive Interface Solutions 2021.....	18
Figura 4 Presencia mundial de fabricación de BCS. Propiedad confidencial © BCS Automotive Interface Solutions 2021.....	18
Figura 5 Operaciones en México - Fabricación . Propiedad confidencial © BCS Automotive Interface Solutions 2021.....	19
Figura 6 Nueva Planta Qro II. Propiedad confidencial © BCS Automotive Interface Solutions 2021. ...	20
Figura 7 Apertura Qro II. Propiedad confidencial © BCS Automotive Interface Solutions 2021.	20
Figura 8 Exhibición de un chaleco con retroalimentación de fuerza Aura Interactor, 1994 (International Data Group, 2016).....	29
Figura 9 Haptic Gaming Vest, el chaleco que crea más de 30 sensaciones físicas al jugar un videojuego (Sin embargo, 2020).....	29
Figura 10 Actualidad Telefonía (D. Valero,2014).	30
Figura 11 Guantes de VR: Gloveone (Kit Digital, 2018).	31
Figura 12 Robot de asistencia para tareas de rehabilitación de los miembros superiores, en el laboratorio del centro tecnológico Cartif (Rodriguez, 2011).....	32
Figura 13 Movimiento de vibración (Blogger,2012).....	32
Figura 14 (a) Variaciones de presión dentro del corazón de un gato. (b) Vibraciones de un diapasón. (A.P. French, 2012).	33

Figura 15 Sistema Masa-Resorte-Amortiguador (Tlacomulco A. , 2024).	34
Figura 16 Clasificación de las vibraciones (Tlacomulco A. , 2024).	35
Figura 17 Vibración resultante amortiguada como una onda tipo seno (Molero B. ,2016).	37
Figura 18 Los sistemas de pantalla táctil háptica (Boréas Technologies, 2020).	39
Figura 19 Los actuadores piezoeléctricos hápticos permiten un rango de vibraciones que pueden ser usadas para indicar diferentes resultados de la activación de un solo botón. (Bóreas Technologies, 2020)	39
Figura 20 Actuador Resonante Lineal (indeed motors, 2020).	40
Figura 21 Estructura de un Actuador Resonante Lineal (Microdrives, 2022).	40
Figura 22 Representación gráfica de una frecuencia (Gutierrez,2020)	41
Figura 23 Movimiento lineal en el eje Z de un actuador lineal resonante (Vibration motors, 2020)	42
Figura 24 Componentes internos de un LRA (Ineed Motors, 2022).	43
Figura 25 Dispersión de fuerza electromagnética sobre el háptico (NFP Motor, 2022).	44
Figura 26 El Motor ERM (E. Siegle, 2011).	45
Figura 27 Diagrama explosivo del interior de un ERM (Ineed motors, 2022)	47
Figura 28 Logotipo Texas Instruments (Texas Intruments, 2024).	49
Figura 29 Secuencia típica de i2c (Texas Instruments, 2018).	51
Figura 30 Transmisión de paquete de datos UART (El procus,2017)	53
Figura 31 Python (Medium, 2022).	53
Figura 32 Logo Qt designer (Qt designer, 2018)	54
Figura 33 Capacidades de desarrollo (Microsoft, 2024).	54
Figura 34 Esquema simplificado de integración de un háptico (Tlacomulco A. , 2024).	56
Figura 35 Diagrama a bloques del sistema general (Tlacomulco A. , 2024)	57
Figura 36 Diagrama a bloques simplificado del driver TI (Texas Instruments, 2018).	59
Figura 37 Pinout DR26xx (Texas Instruments, 2024).	59
Figura 38 Características que ofrece Texas Instrument, reducen tiempos de integración (TI, 2013).	60
Figura 39 PINOUT Arduino Nano (Rochester Electronic, 2024).	62
Figura 40 Diagrama de conexiones para código de prueba de LRA (Tlacomulco A. ,2024).	64
Figura 41 Seteo del microcontrolador en arduino, (Tlacomulco A. , 2024)	64
Figura 42 Código de ejemplo para prueba de LRA´s (Tlacomulco A. , 2024).	65
Figura 43 Salida en el monitor serial de los efectos reproducidos	65
Figura 44 Programación del secuenciador de forma de onda (Texas Instruments Incorporated, 2018).	66
Figura 45 Código para casos de prueba d eefectos (Tlacomulco A., 2024).	67
Figura 46 Modos de operación del driver (Tlacomulco A., 2024).	68
Figura 47 Modos de operación del driver (Tlacomulco A., 2024).	69
Figura 48 Librerías ROM DRV26XX (Texas Instruments, 2024)	74
Figura 49 Salida del monitor serial. Ejecución del código para el microcontrolador. (Tlacomulco A., 2024).	86
Figura 50 d=Diseño de Interfaz en qT designer (Tlacomulco A., 2024)	87
Figura 51 Posicionamiento en la ruta de la carpeta del proyecto (Tlacomulco A., 2024).	88

Figura 52 Conversión de archivo .UI a .PY (Tlacomulco A., 2024)	88
Figura 53 Obtención del código de Python.	88
Figura 54 Integración de las funcionalidades en visual studio.....	89
Figura 55 Diagrama de resultados de la agrupación de sensaciones.....	91
Figura 56 Diagrama de flujo para la programación de la inicialización partes A (Tlacomulco A .,2024)92	
Figura 57 Diagrama de flujo para la programación de la inicialización parte B (Tlacomulco A .,2024). 93	
Figura 58 Diagrama de flujo para la programación de la Auto Calibración parte a (Tlacomulco A .,2024).	94
Figura 59 Diagrama de flujo para la programación de la Auto Calibración parte b (Tlacomulco A .,2024).	95
Figura 60 Diagrama de flujo para la programación del subproceso de la Auto Calibración parte a (Tlacomulco A .,2024)	96
Figura 61 Diagrama de flujo para la programación del subproceso de la Auto Calibración parte b (Tlacomulco A .,2024)	97
Figura 62 Diagrama de flujo para la programación de la reproducción de forma de onda parte a (Tlacomulco A .,2024)	98
Figura 63 Diagrama de flujo para la programación de la reproducción de forma de onda parte b (Tlacomulco A .,2024)	99
Figura 64 48 Diagrama de flujo para la programación del subproceso del reproductor de formas de onda (Tlacomulco A .,2024)	99
Figura 65 Interfaz prototipo (Tlacomulco A., 2024).....	100
Figura 66 Montaje del sistema en prototipo (Tlacomulco A., 2024).....	100

INDICE DE TABLAS

Tabla 1 Unidades típicas de medición de vibración.....	38
Tabla 2 Descripción de los componentes de un LRA	44
Tabla 3 Tabla comparativa entre LRA Y ERM	48
Tabla 4 Selección de Driver de la familia Texas Instruments.....	60
Tabla 5 Selección de las herramientas para desarrollo de software	63
Tabla 6 Ajustes de control para modo STANDBY	70
Tabla 7 Ajustes de registros para I2C.	71
Tabla 8 Registros de configuración para ERM	73
Tabla 9 Registros de configuración para LRA.....	73
Tabla 10 Librerías ROM par DRV26xx y propiedades del actuador.....	74
Tabla 11 Parámetros de Inicialización para el actuador LRA.....	75
Tabla 12 Parámetros de Inicialización para el actuador ERM	76
Tabla 13 Parámetros de Auto calibración para el actuador LRA.....	80
Tabla 14 Parámetros de Auto calibración para el actuador ERM	82
Tabla 15 Secuencia de registros con el efecto 4, 7 y 5.....	85

INDICE DE GRÁFICAS

Gráfica 1 Desglose de Ingresos. Propiedad confidencial © BCS Automotive Interface Solutions 2021.	17
Gráfica 2 : Encuesta a ejecutivos de la industria automotriz 2030. P: ¿Qué tipos de plataformas digitales operará su organización o en cuáles participará?. (IBM, 2019.).....	27
Gráfica 3 Forma de onda típica en un LRA (Texas Instruments, 2013)	43
Gráfica 4 Forma de onda para un ERM típica (Texas Instruments, 2013).	47
Gráfica 5 Ingresos 2023 Texas Instruments (Texas Instruments, 2024).....	50

RESUMEN

La presente memoria de residencias abarca el desarrollo de investigación y diseño para un prototipo basado en tecnología háptica de un banco de pruebas que a través de un HMI permite la reprogramación de múltiples sensaciones de vibración con la finalidad de validar funcionalidades antes de una implementación final. Esta propuesta nace de la necesidad de reducir tiempos de reingeniería, rediseño y adaptación rápida a la demanda del cliente, que a su vez genera un impacto directo en la reducción de basura electrónica.

Se encuentra dividida en cinco capítulos, donde en el Capítulo 1 se presentan las generalidades del proyecto como lo son objetivos, la empresa donde se desarrolló el proyecto, planteamiento del problema y justificación.

Posteriormente en el Capítulo 2 se presenta el marco teórico donde se fundamenta toda la teoría del funcionamiento de la tecnología háptica, específicamente de sus actuadores principales, cuyo funcionamiento está basado en un sistema masa-resorte-amortiguador, así como sus antecedentes y demás fundamentos necesarios para comprender el desarrollo del proyecto.

Durante el Capítulo 3 se abarca el desarrollo de software así como la selección de componentes y herramientas. Se explica detalladamente el proceso necesario para la programación del driver que controla a los actuadores, el manejo de sus registros de memoria, las propuestas de valores, casos de prueba etc. Continuando, se presenta el desarrollo del código para el microcontrolador, y el desarrollo de código para la interfaz, así como la integración de ambos para el prototipo final.

En el Capítulo 4 se muestran los resultados y conclusiones obtenidos, como el diagrama de agrupación de efectos que facilita la customización para el usuario del HMI, los diagramas de flujo desarrollados para la programación de las funcionalidades de los códigos, y finalmente el resultado final del diseño de la interfaz prototipo obtenida.

Finalmente el Capítulo 5, abarca el resumen de las competencias y experiencia obtenida durante el desarrollo de esta estancia de prácticas, así como la formación que se alcanzó con los diferentes cursos tomados.

CAPITULO I

GENERALIDADES DEL PROYECTO

INTRODUCCIÓN

La tecnología háptica ha emergido como un componente revolucionario en la interacción entre humanos y máquinas, permitiendo una experiencia más inmersiva e intuitiva. Esta tecnología, que se basa en la creación de sensaciones táctiles a través de dispositivos, mediante la implementación de actuadores hápticos, ha encontrado aplicaciones significativas en diversas industrias, siendo el sector automotriz uno de los más prometedores.

En los vehículos modernos, la tecnología háptica se utiliza para mejorar la seguridad y la comodidad del conductor, proporcionando retroalimentación táctil en controles como pantallas táctiles y asientos. Además, se prevé que las tendencias futuras en esta área incluyan el desarrollo de sistemas más avanzados que integren inteligencia artificial y aprendizaje automático, permitiendo una personalización aún mayor a la experiencia del usuario. A medida que los vehículos se vuelven más autónomos, la tecnología háptica podría jugar un papel crucial en la comunicación entre el conductor y el vehículo, ofreciendo alertas e información a través de estímulos táctiles que complementen la información visual y auditiva.

Por otro lado, la importancia del desarrollo de bancos de prueba antes de llevar a cabo una implementación final, radica en su capacidad para validar conceptos y funcionalidades de un proyecto de manera temprana y efectiva. Estos demostradores, que son versiones preliminares del producto, permiten a los equipos de desarrollo y a los interesados visualizar y experimentar con el sistema propuesto, facilitando la identificación de problemas y la recolección de retroalimentación valiosa.

Al crear un banco de pruebas capaz de programar y controlar múltiples actuadores de vibración, se pueden realizar pruebas de usabilidad y funcionalidad para producir y validar múltiples sensaciones y curvas de aceleración que ayuden a asegurar que el producto final cumpla con las expectativas y necesidades de los usuarios.

Háptic. Actuador electromagnético que convierte la energía eléctrica en un pulso mecánico o de vibración para la interacción táctil.

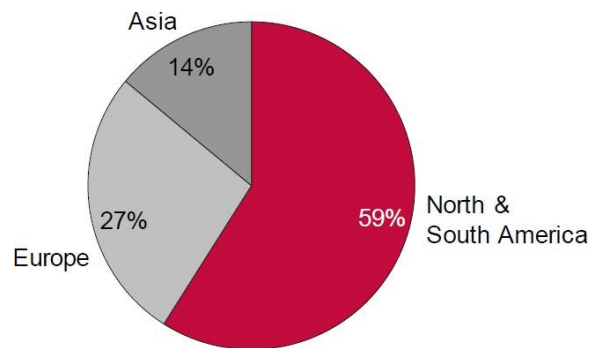
DATOS DE LA EMPRESA

BCS-AIS Ensamblajes Mexico

BCS Automotive Interface Solutions originalmente “Global Body Control Systems de ZF TRW” fue adquirida por Luxshare Consumer Electronics en abril de 2018. Con sede en Radolfzell, Alemania, cuenta con 70 años de historia y presta servicio a una amplia gama de líderes OEM.

Sales	~USD 1 bn
Product lines	6 major categories
Employees	4,000

Figure 1 Cifras clave. Propiedad Confidencial © BCS Automotive Interface Solutions 2021.



Gráfica 1 Desglose de Ingresos. Propiedad confidencial © BCS Automotive Interface Solutions 2021.

BCS junto con empresas del grupo Luxshare ofrecen una amplia gama de soluciones de automoción.



Figure 2 Oferta de soluciones Automotrices. Propiedad confidencial © BCS Automotive Interface Solutions 2021.

OEM. Original Equipment Manufacturer. Es un creador de componentes auténticos para otras empresas de fabricación.

BCS-AIS es un proveedor mundial automotriz especializado en soluciones electrónicas para el interior del vehículo que garanticen un control seguro y cómodo del automóvil por parte del conductor. Tiene alrededor de 3.600 empleados y está operando en distintas ubicaciones y centros de ingeniería en América del Norte, Europa y Asia Pacífico. Al estar presente en 10 países con 8 instalaciones de fabricación y 13 ubicaciones de ventas e Ingeniería, BCS Automotive Interface Solutions tiene una presencia en todas las regiones relevantes de la automoción y apoya a sus clientes a nivel mundial.

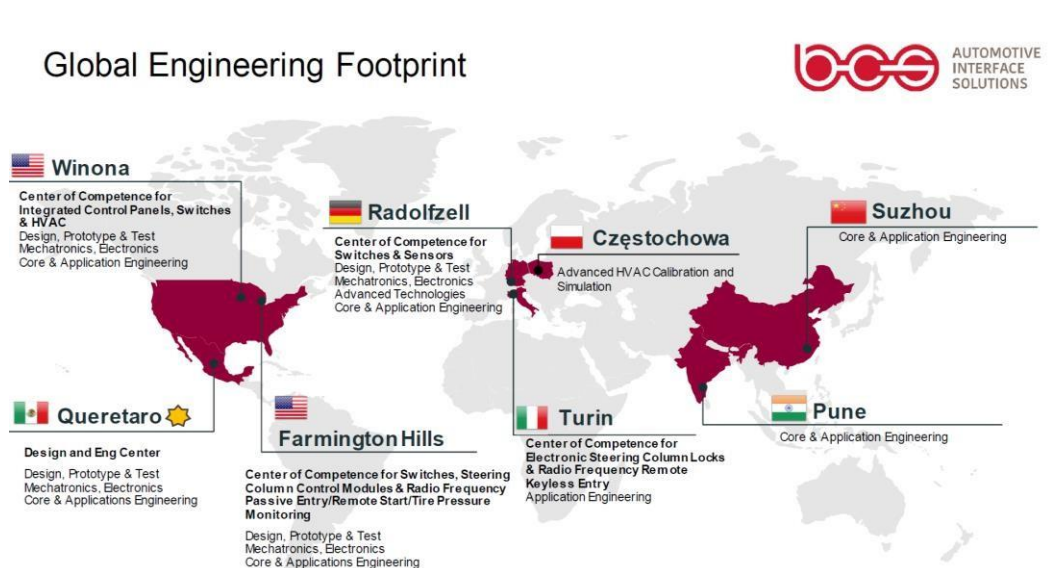


Figure 3 Presencia mundial de ingeniería. Propiedad confidencial © BCS Automotive Interface Solutions 2021



Figure 4 Presencia mundial de fabricación de BCS. Propiedad confidencial © BCS Automotive Interface Solutions 2021.

BCS Automotive Interface Solutions planta 1 en Querétaro

En 2022 con una inversión de 1,200 MDP, BCS Automotive Interface Solutions inauguró en Querétaro su primera planta de manufactura para la producción de módulos de control de dirección y otros componentes electrónicos de alta tecnología para la industria automotriz. El proyecto Wireless Charge Port , con el que arrancó esta planta, está destinado para el mayor el líder global de la industria automotriz.

De acuerdo con la CEO Yuan KK, el Cargador Eléctrico Inalámbrico es un puente tecnológico entre un producto de consumo y el sector automotriz, demostrando las sinergias entre la compañía BCS y su compañía hermana Luxshare-ICT₁.

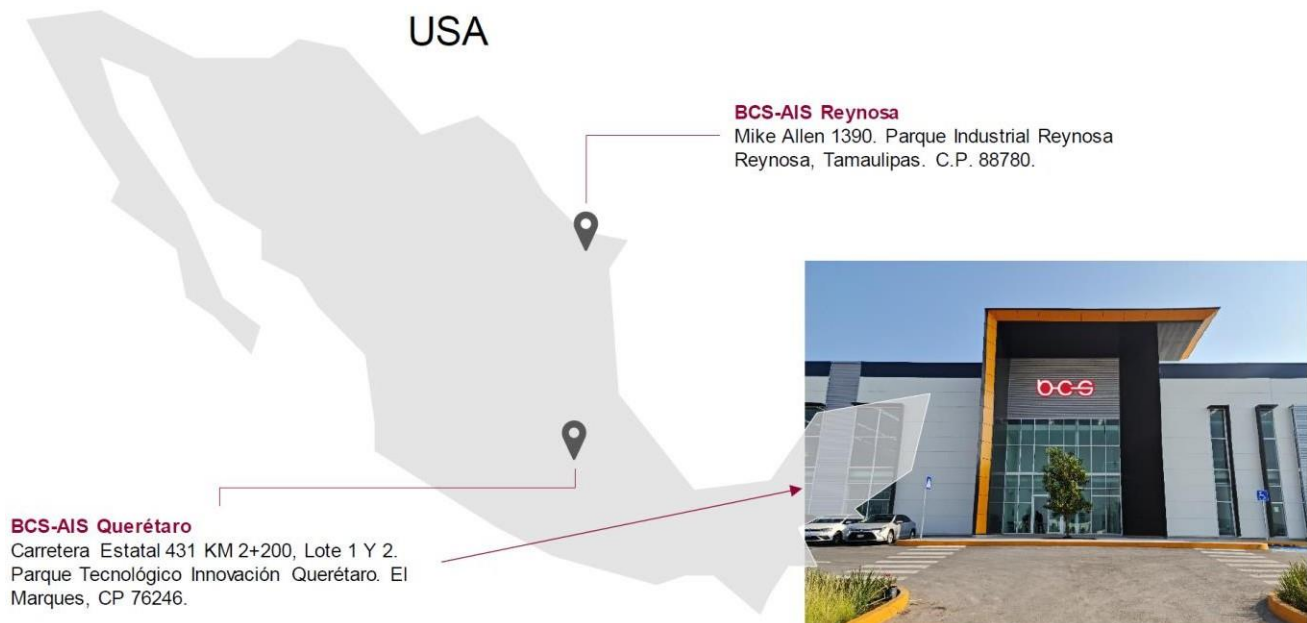


Figure 5 Operaciones en México - Fabricación . Propiedad confidencial © BCS Automotive Interface Solutions 2021.

BCS Automotive Interface Solutions planta 2 en Querétaro

Recientemente en marzo de 2023 BCS-AIS inauguró una segunda planta “BCS-AIS QUERETARO II”, el propósito fundamental de esta nueva sede es abrir un espacio destinado a consolidar un equipo de investigación y desarrollo que sea capaz de dar soporte a todos los sitios de manufactura a nivel global. Y a su vez proveer un sitio para pruebas de alto nivel.



Figure 6 Nueva Planta Qro II. Propiedad confidencial © BCS Automotive Interface Solutions 2021.

Departamento de R&D

El proyecto presentado tiene lugar en el departamento de investigación y desarrollo (Research & Development, R&D), el cual se conforma por el área de Hardware, Software, Mecánica, Sistemas y el Laboratorio de Pruebas, donde se trabaja en conjunto con la finalidad de desarrollar innovación tecnológica en sistemas embebidos para el sector automotriz. Desde el diseño, análisis, implementación, pruebas y validación de nuevos productos dirigidos a clientes finales.



Figure 7 Apertura Qro II. Propiedad confidencial © BCS Automotive Interface Solutions 2021.

Asesores

Oscar Ramirez

Educación

- Estudio una Ingeniería en mecatrónica en la Universidad Tecnológica de Querétaro en 2009-2011.
- TSU en Sistemas Informaticos especialidad en software

Experiencia

- Tecnico de análisis y reparación, Harman de México S. de R.L DE C.V 2012-2013
- CideC QTC 2016-2019
- Tecnico de pruebas
Tecnico de automatizacion
Ingeniero de automatizacion
Desarrollador de software integrador de software
Desarrollador de software general
- Kostal 2020-2022
Integrador de software
Desarrollador de software
- Actualmente de desenvuelve como Desarrollador de Software Sr. en BCS-Automotive Interface Solutions Queretaro II.

Gabriel Ivan Trejo

Educación

- Técnico Superior Universitario Tecnologías de Automatización Industrial
- Ingeniería en Electrónica, robótica y mecatrónica en la Universidad Tecnológica de Querétaro en 2020-2023.

Experiencia

- Desarrollador .NET y Soporte Técnico, DBxtra 2017.
- Auxiliar de Automatización y Control Remoto, Brose Group 2020-2021.
- Investigador Asistente de Instrumentación y Control, Centro Nacional de Metrología 2022-2023
- Control Systems Tools Programmer, GE Aerospace 2023.
- R&D Embedded SW Engineer, en BCS-Automotive Interface Solutions Queretaro II 2023-2024.
- Actualmente R&D Embedded SW Lead, en BCS-Automotive Interface Solutions Queretaro II 2024.

Lugar de realización del proyecto

- Nombre de la empresa: BCS-AIS Ensamblés México, S. de R.L. de C.V (QRO II)
- Giro, Ramo o sector: Automotriz
- Razón Social: TRW ELECTRONICA ENSAMBLES SA DE CV
- Actividades Generales: Ensamble De Paneles De Control Para Uso Automotriz
- Inicio Actividad: 11 de diciembre de 1985
- Domicilio: Parque Industrial PyMe, Calle Industria Energética Lote 1, Manzana 2. El colorado, El Márquez CP. 7626 Querétaro.
- RFC: TRW851211KD0
- Ciudad: El Marqués Querétaro CP 76246
- Teléfono: 8995266178

Descripción del puesto desarrollado: Software Intern

Dentro de las actividades que se deben desarrollar en este puesto, se pueden hacer notar:

- Soporte a las líneas de producción en el área de la manufactura para los diferentes productos, ejecutando códigos de monitoreo de piezas para la verificación de comunicación CAN y la obtención de datos para el análisis de un estudio R&R.
- Extracción de llaves para el desbloqueo de piezas con ciberseguridad.
- Desarrollo e implementación de código para los diferentes productos.
- Realización de requerimientos para funcionalidades de software en los diferentes productos.
- Participación en el desarrollo de sistemas embebidos para nuevos productos.
- Investigación y desarrollo de nuevas tecnologías.
- Diseño y montaje de circuitos electrónicos para placas de prueba para los distintas funcionalidades de software
- Reflasheo de microcontroladores para las diferentes placas de pruebas
- Debugeo de código
- Participación activa en la resolución de problemas presentados en las líneas de producción

PLANTEAMIENTO DEL PROBLEMA

El diseño, fabricación e implementación de un nuevo producto conlleva un largo proceso de investigación, y pruebas, que involucra el trabajo de un gran número de personas y un costo inicial elevado que solo es redituable si el producto cumple con la especificación y aceptación del cliente.

Durante este proceso suelen surgir cambios en cuanto a lo que se quiere lograr y con ello cambios en los diseños, que en casos graves donde algunos productos ya han sido manufacturados representan una alta pérdida en costos por el material, mano de obra, maquinaria y tiempo invertido.

Actualmente el proceso de reingeniería y rediseño representa una importante área de oportunidad para la implementación de mejoras en la prevención de errores y la optimización de tiempos. Estos procesos implican costos adicionales, como la contratación de expertos en el tema, la adquisición de nuevas herramientas o tecnologías, y la capacitación del personal.

Por otro lado existe el riesgo de cometer errores o introducir fallas en los sistemas teniendo consecuencias negativas, como la pérdida de datos, el mal funcionamiento de los sistemas o incluso la interrupción de los servicios que dependen de estos sistemas.

Además el tiempo de implementación es prolongado, lo que puede retrasar realización de mejoras o nuevas funcionalidades. Esto es problemático en un entorno empresarial en constante evolución, donde la rapidez en la implementación de cambios puede marcar la diferencia en términos de competitividad.

En un entorno donde el cambio es constante, es crucial que las empresas sean ágiles y capaces de adaptarse rápidamente a nuevas condiciones y demandas de los clientes. Al reducir los tiempos de rediseño y pruebas, las empresas pueden mejorar su eficiencia y productividad. Esto significa que pueden realizar los cambios necesarios en sus procesos de manera más rápida y efectiva, lo que les permite adaptarse rápidamente a los cambios del mercado.

OBJETIVOS

Objetivo General

Diseñar, programar e implementar un prototipo de un banco de pruebas capaz de controlar múltiples actuadores de vibración usando un microcontrolador a través de un HMI para producir y validar múltiples sensaciones y curvas de aceleración en dispositivos de gama alta para usuarios finales.

Objetivos Específicos

- Estudiar y comprender el funcionamiento del DRV26xx a través de su data sheet para poder programar sus diferentes modos de uso
- Analizar y comprender los registros de memoria comprendidos en DRV26xx para la customización de los parámetros necesarios programación de las diferentes sensaciones de vibración
- Diseñar los esquemáticos electrónicos para la comunicación maestro-esclavo con los circuitos integrados de control
- Diseñar los diagramas a bloques del funcionamiento y lógica entre los diferentes componentes de hardware
- Desarrollar códigos de prueba para la validación del funcionamiento de los actuadores.
- Desarrollar código de prueba para la categorización de efectos
- Agrupar los efectos de forma de onda según la sensación que produzcan para una mejor interacción del usuario con el HMI
- Diseñar diagramas de flujo del funcionamiento del driver que se usará en el microcontrolador
- Desarrollar el código para el control del driver de acuerdo con el diseño del diagrama de flujo
- Diseñar el software HMI para controlar el prototipo por medio de UART
- Desarrollar el código de programación para el HMI
- Integrar las funcionalidades del código para el control del driver junto con el del desarrollo del HMI

JUSTIFICACIÓN

El desarrollo de nuevas tecnologías en el sector automotriz representa una ventaja competitiva con respecto a la oferta de nuevos productos, especialmente si estos son innovadores.

Al implementar un banco de pruebas háptico con un microcontrolador que permita manejar diferentes tipos de actuadores con diferentes configuraciones se podría mostrar todas las diferentes sensaciones posibles de un actuador de vibración y así decidir cuál de ellos implementar en un producto de usuario final reduciendo tiempos de rediseño, pruebas, y costos en reingeniería.

Esto no solo reduce el riesgo de errores costosos en etapas posteriores, sino que también mejora la comunicación entre los desarrolladores y los clientes, asegurando que todos estén alineados en cuanto a los objetivos y requisitos del proyecto.

Sin mencionar que el desarrollo de un demostrador de este tipo reduciría el nivel de basura electrónica que se genera al fabricar n número de placas de prueba, que al final resultan inservibles y contribuyen al problema global de e-waste.

Finalmente, permite iterar sobre el diseño y las características del producto, lo que fomenta un enfoque más ágil y adaptable al desarrollo. En un entorno donde la tecnología y las necesidades del mercado cambian rápidamente, contar con un banco de pruebas efectivo puede ser la clave para el éxito de la implementación final, garantizando que el producto no solo sea funcional, sino también relevante y competitivo.

DELIMITACIÓN

En este proyecto se desarrollará la documentación de diseño de la programación, documentación de casos de prueba, esquemáticos, diagramas a bloques y diagramas de flujo para poder realizar la programación necesaria y así demostrar las diferentes configuraciones de vibraciones que se pueden obtener. Se presentan los componentes a utilizar, la metodología y los resultados de diseño de software.

CAPITULO II

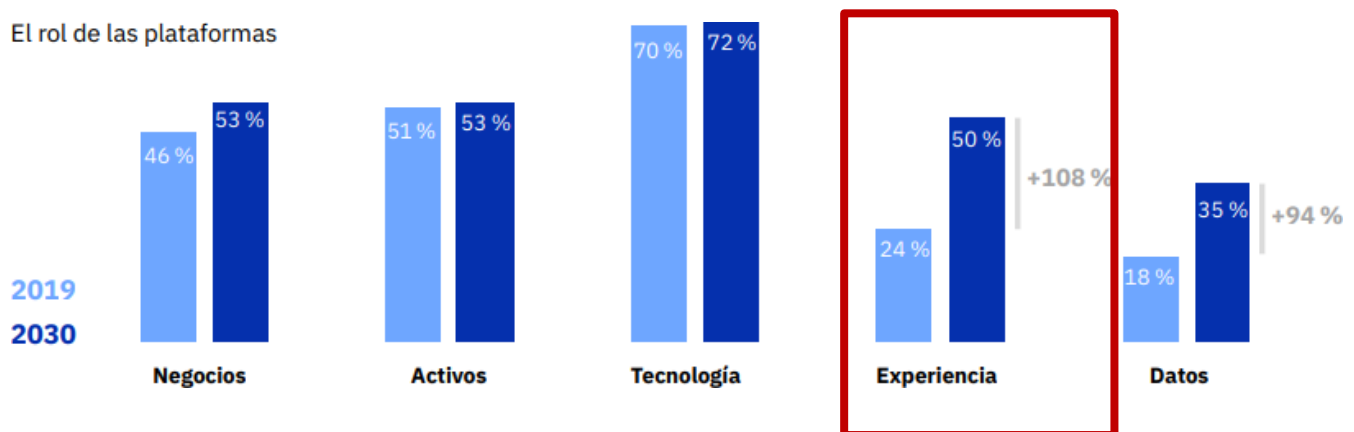
MARCO TEÓRICO

INDUSTRIA AUTOMOTRIZ EN MÉXICO

La industria automotriz es uno de los sectores que ha evolucionado con la incursión de las nuevas tecnologías. La inversión de estas tecnologías estructura un modelo de negocios de última generación. La tendencia hacia los próximos años requiere de una implementación de las tecnologías digitales y una reinversión de los procesos de producción para sobrevivir, de acuerdo con el Instituto for Business o IBM².

En el sector de autopartes, México se posiciona como el cuarto mayor productor a nivel mundial, quedando por detrás únicamente de China, Estados Unidos y Japón, y superando a países líderes como Alemania e India. Durante el año 2021, México alcanzó un valor de producción de 94,778 millones de dólares en autopartes, la cual fue exportada en su gran mayoría a Estados Unidos. Esto ha llevado a la consolidación de México como el principal proveedor de autopartes para su vecino del norte³.

En este sentido para que las compañías automotrices logren destacar sus marcas, la innovación continúa es crucial. La generación de nuevo crecimiento se reduce en cinco tipos de plataformas digitales, que se reducen a las observadas en la gráfica 2.



Gráfica 2 : Encuesta a ejecutivos de la industria automotriz 2030. P: ¿Qué tipos de plataformas digitales operará su organización o en cuáles participará?. (IBM, 2019.)

La grafica anterior muestra los resultados obtenidos de un estudio que realizo IBM a 1500 ejecutivos, donde se observa que el 24% de ellos considera que el desarrollo de innovación en la experiencias del usuario final son claves para asegurar la competitividad de la empresa y se proyecta que para 2030 alcance el 50%.

TECNOLOGÍA HÁPTICA

El término háptica designa la ciencia del tacto, por analogía con la acústica y la óptica. La palabra proviene del griego háptō («tocar», relativo al tacto)⁴. Algunos teóricos, como Herbert Read, han extendido el significado de la palabra háptica, refiriéndose por exclusión a todo el conjunto de sensaciones no visuales y no auditivas que experimenta un individuo.

A su vez la tecnología háptica, también es conocida como comunicación cinestésica, proporciona una forma de recrear el sentido del tacto mediante la aplicación de una serie de fuerzas, vibraciones y movimientos a la persona¹².

Por otro lado el háptica feedback, también conocido como retroalimentación táctil, es una tecnología que recrea el sentido del tacto al aplicar fuerzas, vibraciones o movimientos al usuario. Esta tecnología se utiliza en muchos dispositivos de uso diario, como los teléfonos móviles cuando vibran al recibir una notificación.

Por lo tanto la tecnología háptica se refiere al conjunto de interfaces tecnológicos que interaccionan con el ser humano mediante el sentido del tacto. Se trata de una tecnología emergente desde los 90 que está evolucionando prometedoramente en algunos campos de aplicación.

Antecedentes

Una de las formas más tempranas de uso de las tecnologías hápticas en dispositivos se introdujo en los sistemas servo de los grandes aviones con la intención de operarlos remotamente. Estos sistemas primitivos actuaban en una única vía de tal forma que la fuerza aplicada a un mando se multiplicaba hacia las superficies de control aerodinámico (aleros, etc.), pero el piloto no obtenía como respuesta un factor de la fuerza resistente. Es de pensar que en los inicios los pilotos de los pequeños aviones sin servo tenían todas las sensaciones acerca de la resistencia sobre las actuaciones de una palanca y esto suponía una seguridad en ciertas situaciones de peligro. Lo cierto es que con los servos iniciales el piloto no obtenía esta sensación. Para resolver este problema se instaló un sistema de control que proporcionaba una resistencia a la palanca del piloto proporcional al ángulo de ataque, este es un ejemplo de interfaz háptico⁵.

En septiembre de 1994, se desarrolló el chaleco Aura Interactor. El chaleco es un dispositivo portátil de retroalimentación de fuerza que monitorea una señal de audio y utiliza tecnología de actuador electromagnético para convertir las ondas de sonido de bajos en vibraciones que pueden representar acciones tales como un golpe o una patada. El chaleco se conecta a la salida de audio de un estéreo, TV o VCR y la señal de audio se reproduce a través de un altavoz incorporado en el chaleco⁶.



Figure 8 Exhibición de un chaleco con retroalimentación de fuerza Aura Interactor, 1994 (International Data Group, 2016).

En noviembre del mismo año, la película 4-D Honey, I Shrunk the Audience! simulaba que los ratones se soltaban y corrían por el auditorio. Para simular la sensación de una manada de ratones escapados, se bombea aire a través de un pequeño tubo de plástico que hace que se agite violentamente e imite la sensación de las colas de los ratones rozando las piernas de los miembros de la audiencia⁵.

Juegos

Algunos dispositivos desarrollados para juegos tienen estas características. Es de destacar sobre todo en algunos joysticks y controladores de juego proporcionan un retorno háptico como en el caso del Dualsense. La forma más sencilla en los juegos es lo que se denomina tabletas vibradoras (rumble packs) con los que el usuario siente como hay irregularidades tales como un terreno desigual cuando conduce un automóvil.



Figure 9 Haptic Gaming Vest, el chaleco que crea más de 30 sensaciones físicas al jugar un videojuego (Sin embargo, 2020).

Pantallas hápticas

Hoy en día están apareciendo nuevas tecnologías para aumentar la sensación que transmite una pantalla al tocarla. Las pantallas táctiles se han convertido en un dispositivo indispensable en los interiores de los automóviles actuales. La enorme cantidad de funciones, controles y modos de funcionamiento que se ponen disposición realza la necesidad de facilitar su accionamiento. Sin embargo, se presenta un problema derivado de la ausencia de botones físicos, y es que el conductor debe desplazar la mirada de la carretera para asegurarse de que ubica el dedo en el lugar correcto en caso de autopistas, la distancia puede ser de 40 metros, lo que conlleva un riesgo añadido al volante.

En los últimos tiempos han aparecido innovaciones aplicadas como las pantallas hápticas, las cuales permiten emular la presión de un botón sin dejar de ser táctiles, algo que permite acometer el problema anteriormente mencionado así como resultar más familiar al usuario más tradicional.

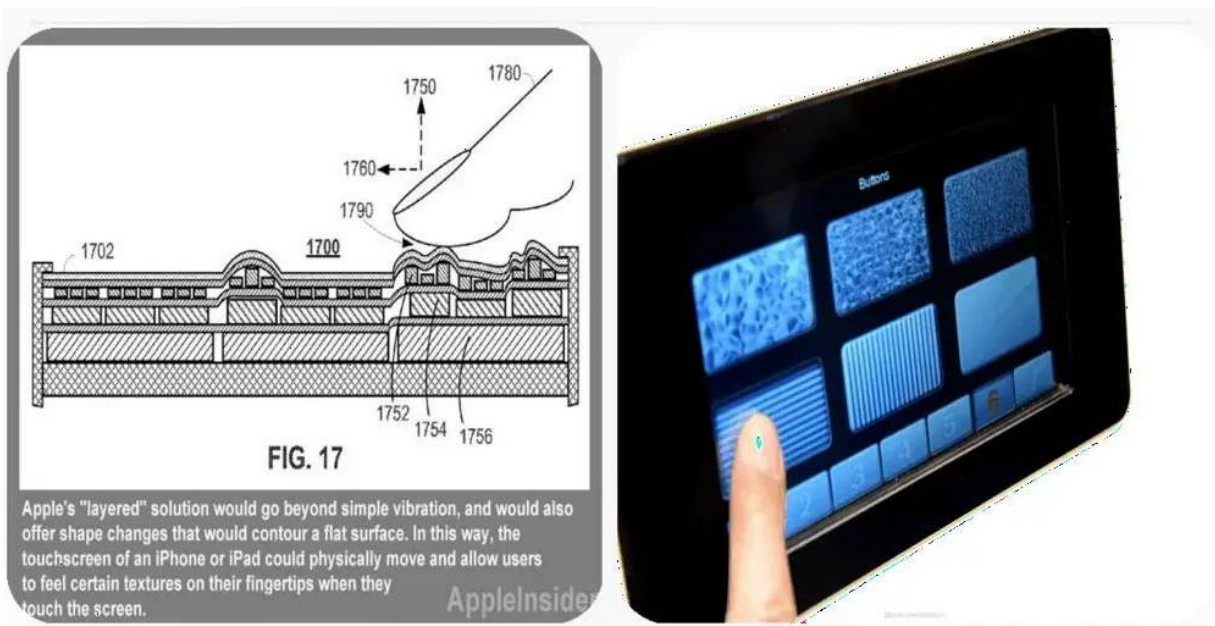


Figure 10 Actualidad Telefonía (D. Valero, 2014).

Las pantallas hápticas principalmente hacen uso de cristales piezoeléctricos incrustados tras la lámina principal. Este cristal se expande al hacerle circular una pequeña corriente eléctrica cuando se sobre él, lo que genera la impresión de estar haciendo un 'click'. Acompañado de ello suele ir un sonido para aumentar más la sensación.

La pantalla táctil háptica funciona mediante la generación de una fuerza que el usuario puede sentir cuando toca la pantalla. Esta fuerza se crea a través de una serie de vibraciones o movimientos, proporcionando una retroalimentación que replica la sensación de tocar un objeto físico. Esta interacción es posible gracias a actuadores electromagnéticos o piezoeléctricos colocados debajo de la pantalla que producen pequeñas vibraciones⁸.

- Actuadores electromagnéticos: Estos dispositivos convierten la energía eléctrica en movimiento mecánico a través del uso de campos magnéticos.
- Actuadores piezoeléctricos: Estos funcionan bajo el principio de la piezoelectricidad, que es la capacidad de algunos materiales para generar una carga eléctrica en respuesta a la aplicación de presión mecánica.

Entornos Hápticos en realidad virtual

Las interfaces hápticas van ganando cada vez más terreno en parte de la realidad virtual, añadiendo la sensación del tacto a los entornos visuales, tal y como es en el sistema denominado: 3D-Mobile Immersive Workstation¹⁰. Es decir, mediante el uso de estos artilugios, se puede percibir la presencia de objetos tridimensionales en un ambiente virtual.

La herramienta más clara y que se ajusta de mejor manera, son los guantes hápticos, variadas empresas han entrado de lleno en este campo concreto, la posibilidad de sentir un objeto no real al “agarrarlo” utilizando estos dispositivos, lo hace una experiencia totalmente atractiva. NeuroDigital Technologies, tiene en su haber los guantes Gloveone. Estos tienen 10 actuadores, localizados en las yemas de los dedos y la palma de la mano, y un plus añadido, funcionan con un gran número de gafas de VR.

Asimismo, crean de cierta manera una retroalimentación de fuerzas; se generan sensaciones en tus músculos y tendones. La persona que se encuentre utilizando estos dispositivos enviará información al sistema informático y recibirá una sensación en alguna parte de su cuerpo como feedback.



Figure 11 Guantes de VR: Gloveone (Kit Digital, 2018).

Medicina

Existen diversos interfaces hápticos en el caso de simulación médica y que pueden resultar útiles a la hora de intentar minimizar los daños ocasionados por el uso de procedimientos invasivos (laparoscopia/radiología de intervención) y permitir obtener de forma remota. En el futuro, es muy posible que los médicos expertos puedan hacer operaciones remotas. Además se han utilizado dispositivos hápticos en rehabilitación asistida por robots¹⁰.

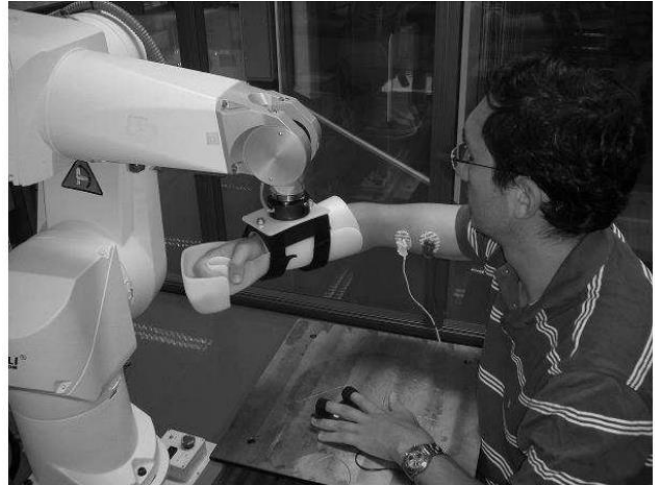


Figure 12 Robot de asistencia para tareas de rehabilitación de los miembros superiores, en el laboratorio del centro tecnológico Cartif (Rodríguez, 2011).

TEORÍA DE VIBRACIONES

Vibración

En su forma más sencilla, una vibración se puede considerar como la oscilación o el movimiento repetitivo de un objeto alrededor de una posición de equilibrio. La posición de equilibrio es a la que llegará cuando la fuerza que actúa sobre él sea cero. Este tipo de vibración se llama vibración de cuerpo entero, lo que quiere decir que todas las partes del cuerpo se mueven juntas en la misma dirección en cualquier momento¹⁷.

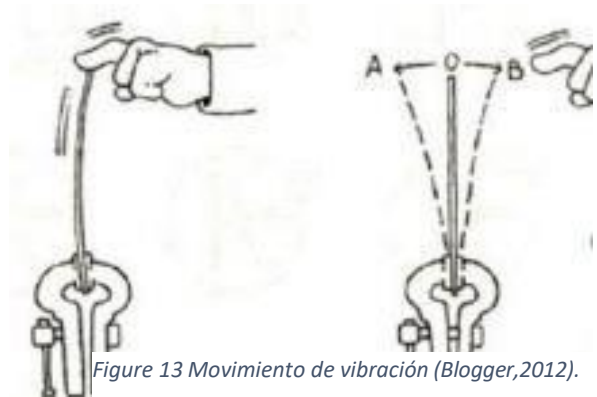


Figure 13 Movimiento de vibración (Blogger, 2012).

La palabra vibración se refiere a los movimientos oscilatorios (hacia adelante o hacia atrás) de las estructuras, de los sistemas mecánicos o de sus componentes. Por lo general, la vibración está caracterizada por el desplazamiento, la velocidad o la aceleración¹⁸.

Las vibraciones se caracterizan por su frecuencia y por su amplitud.

- **Frecuencia** es el número de veces por segundo que se realiza el ciclo completo de oscilación y se mide en Hercios (Hz) o ciclos por segundo.
- **Amplitud** se puede medir en: Aceleración m/s^2 , en velocidad m/s y en desplazamiento m , que indican la intensidad de la vibración.

La característica común de todos estos fenómenos es su periodicidad. Existe un esquema de movimiento o desplazamiento que se repite una y otra vez. Este esquema puede ser sencillo o complicado. La figura 14 muestra el ejemplo de dos casos, el ciclo mas bien complicado de variaciones de presión dentro del corazón de un gato y la curva sinusoidal pura de las vibraciones de un diapasón. En ambos casos el eje horizontal representa el avance continuo de tiempo y puede identificarse la duración del tiempo, el periodo T , dentro del cual se realiza u ciclo completo de vibración¹⁹.

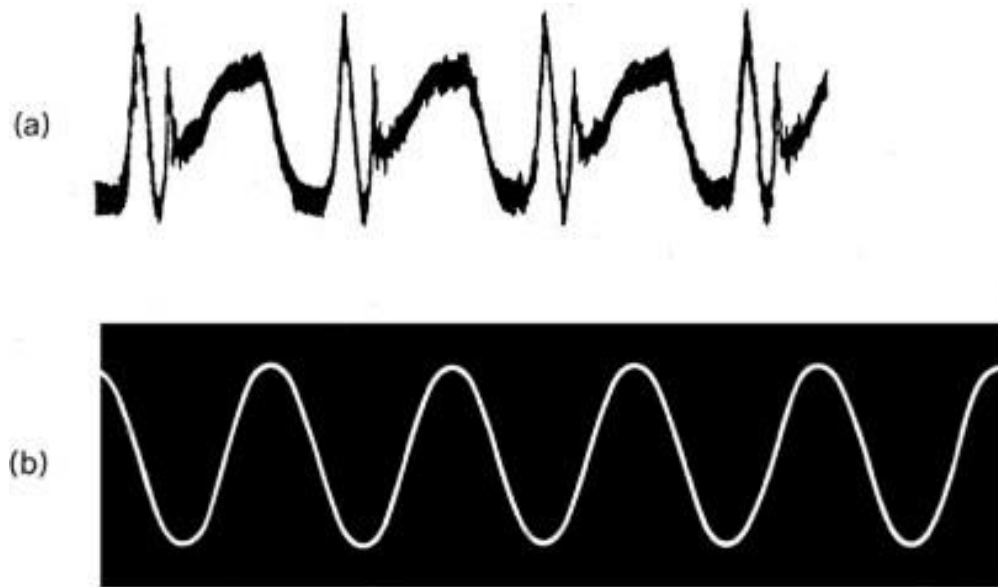


Figure 14 (a) Variaciones de presión dentro del corazón de un gato. (b) Vibraciones de un diapasón. (A.P. French, 2012).

La vibración de un objeto es causada por una fuerza de excitación. Esta fuerza se puede aplicar externamente al objeto o puede tener su origen a dentro del objeto.

Una estructura vibrante tiene cuatro propiedades básicas: masa, rigidez, amortiguamiento y desplazamiento. La naturaleza de la oscilación está determinada no sólo por la masa sino también por la rigidez y el amortiguamiento propio de la estructura.

Diapasón. El diapasón es una pieza metálica, inventada por el británico **John Shore** en el **siglo XVIII**, que tiene forma de **horquilla**. La vibración del diapasón genera un sonido que se toma como referencia a la hora de afinar un instrumento, tanto uno fabricado por el ser humano como la voz.

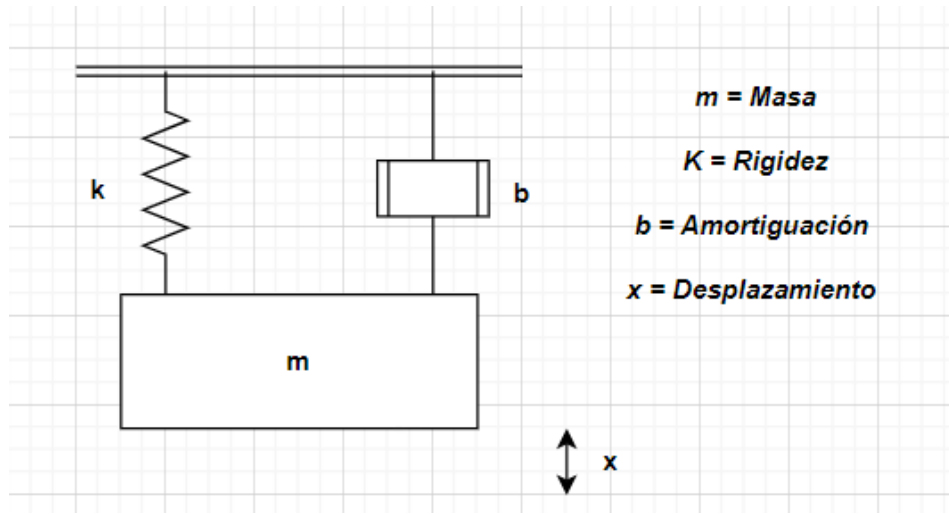


Figure 15 Sistema Masa-Resorte-Amortiguador (Tlacumulco A. , 2024).

Las vibraciones mecánicas aparecen cuando la estructura se perturba a partir de su posición de equilibrio aplicando bien un impulso o una excitación periódica.

- Una excitación del tipo impulso produce una vibración libre de la estructura, que vibra a una o más frecuencias naturales (o frecuencias de resonancia) de la estructura y genera una respuesta de cierta magnitud.
- Una excitación periódica produce una vibración forzada de la estructura, que vibra a la frecuencia de la excitación periódica.

Cuando el amortiguamiento está presente en cualquier vibración libre o forzada, el movimiento de la estructura eventualmente se reduce a cero debido a la disipación de energía.

Cada estructura tiene una o más frecuencias naturales de vibración. Las frecuencias naturales (también llamadas frecuencias de resonancia) es la frecuencia a la cual la rigidez y las fuerzas de inercia se anulan entre sí. En análisis modal, los picos de la función de respuesta en frecuencia (FRF) se usan para identificar las frecuencias naturales y modos de vibración de la estructura.

Clasificaciones de las vibraciones

Las vibraciones se pueden clasificar de varias maneras a continuación se muestra un diagrama general de cómo es que pueden agruparse.

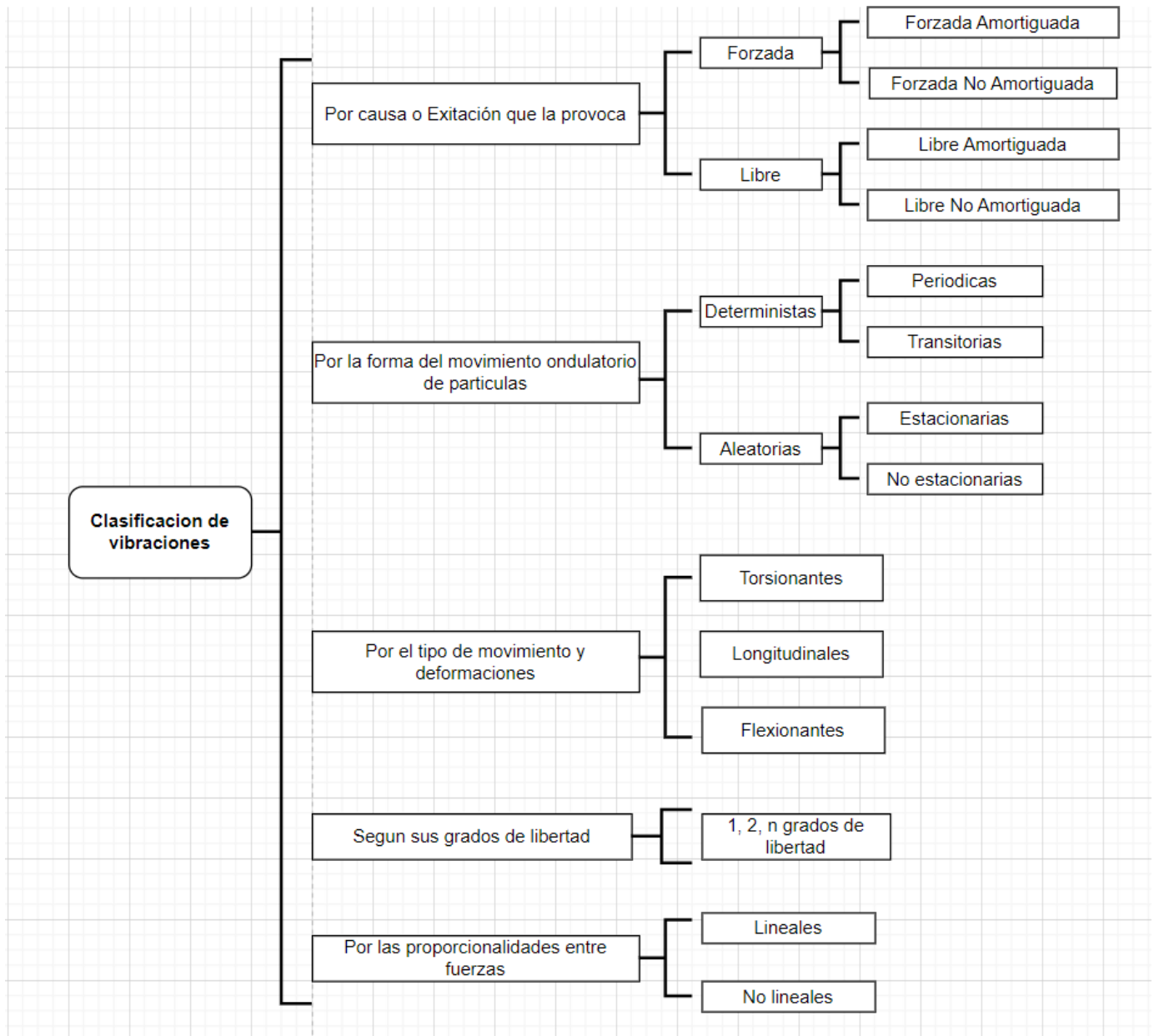


Figure 16 Clasificación de las vibraciones (Tlacomulco A. , 2024).

Vibración Libre Amortiguada

En el caso de una vibración libre amortiguada la masa oscila con su frecuencia natural alrededor del punto de equilibrio con una magnitud que tiende a cero debido a la disipación de la energía. Debido a que la fuerza de rozamiento es directamente proporcional a la velocidad de la masa, el término de amortiguamiento se obtiene multiplicando la constante de amortiguamiento c por la velocidad. El amortiguamiento se introduce como un valor negativo en la fuerza resultante. La expresión de la fuerza resultante es igual a la masa multiplicada por la aceleración:

$$mg - k(\Delta + x) - c\dot{z} = m\ddot{x}$$

Donde:

m = Masa

g = Gravedad

Δ = Elongación del muelle en reposo

x = Desplazamiento

c = Constante de amortiguamiento

k = Constante de rigidez del muelle

$\dot{z}$ = Velocidad

$\ddot{x}$ = Aceleración

Recordemos que en el punto de equilibrio las fuerzas de la estructura son $mg = k\Delta$, por tanto la ecuación de movimiento para un sistema de vibración libremente amortiguado puede expresarse como una ecuación diferencial de segundo orden:

$$m\ddot{x} + c\dot{x} + kx = 0$$

Nótese que todos los términos estructurales están presentes: masa, amortiguamiento, rigidez y desplazamiento. La siguiente imagen muestra la gráfica en función del tiempo de la vibración resultante amortiguada como una onda tipo seno. El valor pico de la Función de Respuesta en Frecuencia (FRF) es la frecuencia natural de la estructura.

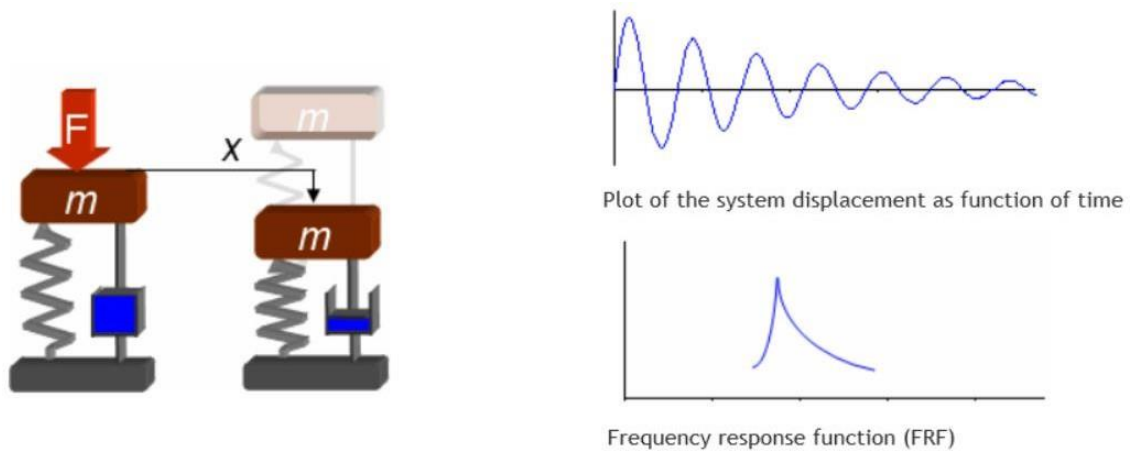


Figure 17 Vibración resultante amortiguada como una onda tipo seno (Molero B., 2016).

Unidades de medición de vibración

El desplazamiento es sencillamente la distancia desde una posición de referencia, o punto de equilibrio. Aparte de un desplazamiento variable, un objeto vibrando tendrá una velocidad variable y una aceleración variable. La velocidad se define como la proporción de cambio en el desplazamiento y en el sistema inglés, se mide por lo general en pulgadas por Segundo (PPS). Aceleración se define como la proporción de cambio en la velocidad y en el sistema inglés se mide en unidades G, o sea la aceleración promedio debida a la gravedad en la superficie de la tierra.

En el sistema inglés de medición, el desplazamiento se mide generalmente en mils (milésimos de pulgada), y el valor pico a pico se usa por convención. La velocidad generalmente se mide en pulgadas por segundo y la convención es de usar el valor pico o el valor RPC. La aceleración se mide generalmente en Gs. 1 G es la aceleración debida a la gravedad en la superficie de la tierra. El G en realidad no es una unidad de aceleración, es sencillamente una cantidad de aceleración a que estamos sometidos como habitantes de la tierra. A veces la aceleración se mide en pulgadas por segundo por segundo (pulgadas/seg²) o m/seg², que son unidades verdaderas. Un G es igual a 386 pulgadas / seg² o 9.81 m/seg². El procedimiento de convertir una señal de desplazamiento hacia velocidad o de velocidad hacia aceleración es equivalente a la operación matemática de diferenciación

Los parámetros de vibración se miden de forma casi universal empleando las unidades métricas ISO que se muestran en la tabla 1. Sin embargo, es muy habitual utilizar el valor de la gravedad "g" para expresar la aceleración, aunque no forma parte del sistema ISO de unidades coherentes.

Tabla 1 Unidades típicas de medición de vibración.

	Sistema Internacional	sistema imperial
desplazamiento	micrón –pico a pico (1 micra = 0,001 mm)	mils-pico a pico (1000 mils – 1 pulgada)
velocidad	mm / s pico o rms	pulg/s pico o rms IPS pico o rms
aceleración	sol, pico o rms ($9,81 \text{ EM}^2 = 1\text{g}$)	sol ($9,8 \text{ EM}^2$) , pico o rms 386,1 político / $\text{s}^2 = 1\text{g}$

ACTUADORES HÁPTICOS

Los actuadores hápticos son dispositivos que permiten a los usuarios experimentar sensaciones táctiles y de fuerza en la interacción con diferentes dispositivos y aplicaciones. Estos dispositivos utilizan tecnologías como la vibración y la presión para crear una retroalimentación táctil realista.

Existen diferentes tipos de actuadores hápticos, como la Masa Giratoria Excéntrica (ERM) y los Actuadores Lineales Resonantes (LRA), que se encuentran en ordenadores portátiles y equipos móviles. Estos actuadores generan vibraciones y movimientos que permiten a los usuarios sentir y tocar virtualmente.

Opciones de retroalimentación háptica

Los diseñadores que buscan añadir retroalimentación háptica a HMI tienen tres opciones para elegir: ERM, LRA y actuadores piezoeléctricos. Los tres funcionan aplicando el mismo principio básico, según el cual un microcontrolador supervisa el funcionamiento y un conductor controla un actuador para generar vibración (Figura 13). Las diferencias radican en la forma en que se crea esa vibración: el ERM utiliza un peso giratorio descentrado, el LRA utiliza un peso magnético suspendido por muelles helicoidales, y el dispositivo piezoeléctrico se basa en el efecto piezoeléctrico (inverso) que provoca un cambio en las dimensiones de un cristal u objeto de cerámica cuando se somete a un campo eléctrico¹¹.

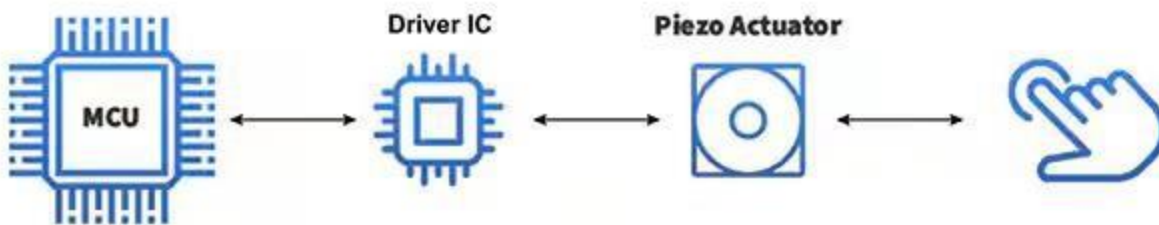


Figure 18 Los sistemas de pantalla táctil háptica comprenden un microcontrolador, un conductor y un actuador. En este ejemplo el actuador es un dispositivo piezoeléctrico, pero los ERM y los LRA son alternativas populares. (Boréas Technologies, 2020).

Los ERM y los LRA han demostrado ser populares para los diseños portátiles principalmente porque ofrecen una opción de bajo voltaje (aproximadamente 3 voltios) y requieren un diseño de conductor simple.

Por el contrario, los dispositivos piezoeléctricos convencionales requieren voltajes mucho más altos (hasta 200 voltios) para generar suficiente deformación mecánica para una buena experiencia de consumo. Estos altos voltajes son necesarios porque la tecnología convencional de conductores hápticos piezoeléctricos se basa generalmente en la tecnología de amplificadores de audio a escala, en lugar de haber sido diseñada desde el principio teniendo en cuenta la baja potencia. Un desafío adicional para el diseño ha sido la falta de conductores de actuadores hápticos de baja potencia dedicados, lo que ha obligado a los diseñadores a recurrir a soluciones menos eficientes¹¹.

Sin embargo, una ventaja clave de las soluciones piezoeléctricas es soportar hápticos de alta definición que mejoran la retroalimentación más allá de la simple vibración monótona. Por ejemplo, los actuadores piezoeléctricos pueden producir vibraciones que varían en frecuencia y amplitud y pueden utilizarse para significar diferentes resultados al pulsar un solo botón (figura 14)¹¹.

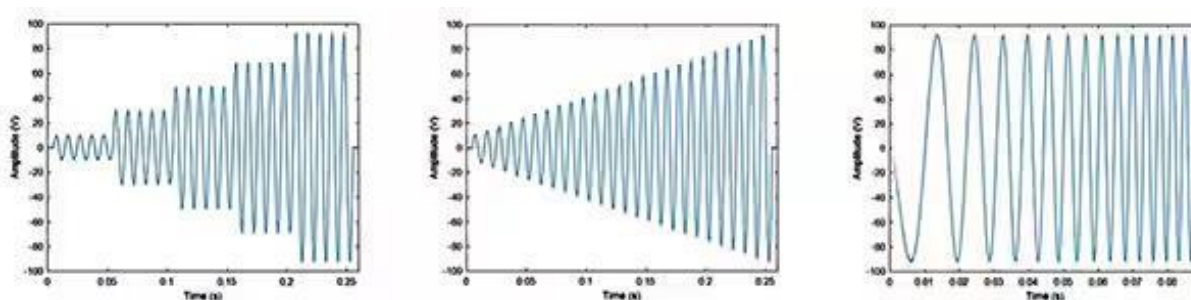


Figure 19 Los actuadores piezoeléctricos hápticos permiten un rango de vibraciones que pueden ser usadas para indicar diferentes resultados de la activación de un solo botón. (Bóreas Technologies, 2020)

En resumen, las soluciones de retroalimentación háptica de alta definición requieren:

- Amplio ancho de banda: para permitir un rango de frecuencias y modos de vibración
- Aceleración alta [g]: para una fuerza de retroalimentación más fuerte
- Gran desplazamiento: que aumenta la sensibilidad de retroalimentación háptica
- Baja latencia: el tiempo de respuesta rápida aumenta el rango de retroalimentación

El Actuador Resonante Lineal (LRA)

Los actuadores lineales de resonancia son dispositivos mecánicos que utilizan la resonancia para producir movimientos lineales de alta precisión y velocidad. Estos actuadores se han vuelto cada vez más populares en la industria debido a su capacidad para ofrecer una solución más eficiente y precisa en comparación con los sistemas de accionamiento convencionales¹⁴.



Figure 20 Actuador Resonante Lineal (indeed motors, 2020).

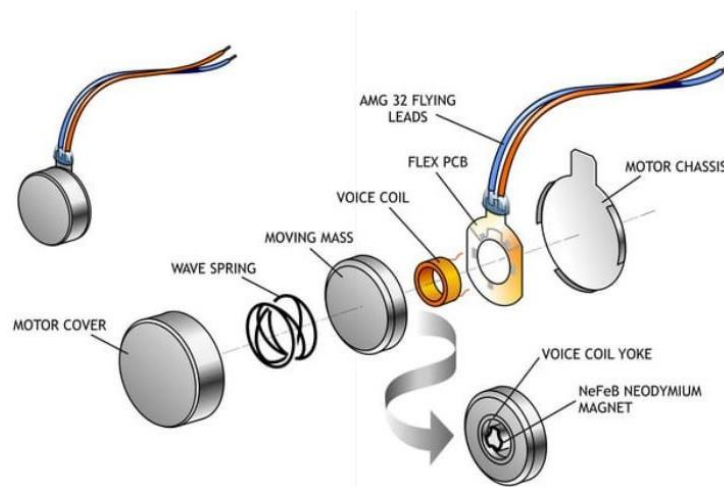


Figure 21 Estructura de un Actuador Resonante Lineal (Microdrives, 2022).

Resonancia y Frecuencia Natural

Todo cuerpo que tiene una masa y una elasticidad, es capaz de vibrar y lo hace a unas frecuencias (Hertz= ciclo por segundo = cps) que dependen de los grados de libertad que tenga la estructura y de la distribución de sus masas.

Todos los objetos tienen una frecuencia que los caracteriza y ésta se denomina frecuencia natural. Incluso pueden tener muchas frecuencias naturales en función de su geometría. Si se emite una vibración cerca del objeto, éste comienza a vibrar “sin que nadie lo toque” debido a su frecuencia natural. Es la frecuencia a la que un objeto seguirá vibrando después de golpearlo¹⁶.



Figure 22 Representación gráfica de una frecuencia (Gutierrez,2020)

Si a un objeto se le envía una vibración con la misma frecuencia que su frecuencia natural, la vibración del objeto comenzará a ser más pronunciada y por consiguiente, se acumulará cada vez más, así como también se acumulará cada vez más energía, a esto se le conoce como **resonancia**¹⁶.

En este sentido, la resonancia es un fenómeno que amplifica una vibración. Se produce cuando una vibración se transmite a otro objeto cuya frecuencia natural es igual o muy cercana a la de la fuente, en los sistemas eléctricos se utiliza la frecuencia de resonancia para describir cuando un circuito alcanza la respuesta máxima de oscilación.

Funcionamiento

Los actuadores lineales de resonancia utilizan la resonancia mecánica para producir movimientos lineales. En el caso de los actuadores lineales de resonancia, el objeto vibrante es la pieza móvil del actuador, que se mueve hacia adelante y hacia atrás en un movimiento lineal.



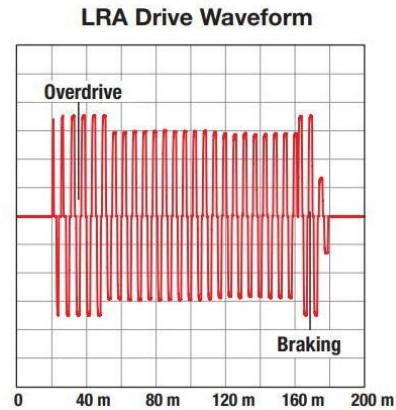
Figure 23 Movimiento lineal en el eje Z de un actuador lineal resonante (Vibration motors, 2020).

Un actuador lineal resonante es un motor vibratorio de precisión que produce una fuerza oscilante en un solo eje, este utiliza una tensión para accionar una bobina móvil presionada contra una masa móvil conectada a un muelle.

Aunque la frecuencia y la amplitud de un actuador lineal resonante pueden ajustarse cambiando la entrada de tensión, el actuador debe accionarse a su frecuencia de resonancia para generar una cantidad significativa de fuerza para una corriente grande. En las condiciones anteriores, la bobina electrificada genera un campo magnético que mueve el imán para entrar en contacto con el repelente, de modo que el muelle puede impulsar el bloque de masa hacia arriba y hacia abajo para generar fuerza.

El movimiento lineal se logra mediante la aplicación de una señal de frecuencia específica al motor de resonancia. Cuando la frecuencia de la señal coincide con la frecuencia natural del motor de resonancia, se produce una amplificación de la vibración en la pieza móvil del actuador, lo que resulta en un movimiento lineal suave y preciso¹³.

Los actuadores lineales de resonancia constan de tres componentes principales: un motor de resonancia, un amplificador de potencia y un controlador. El motor de resonancia está compuesto por una pila de laminaciones de hierro y bobinas de cobre. El amplificador de potencia proporciona la energía necesaria para impulsar el motor de resonancia, mientras que el controlador regula la frecuencia y la amplitud de la vibración del motor de resonancia.



Gráfica 3 Forma de onda típica en un LRA (Texas Instruments, 2013).

Componentes de los actuadores de resonancia lineal

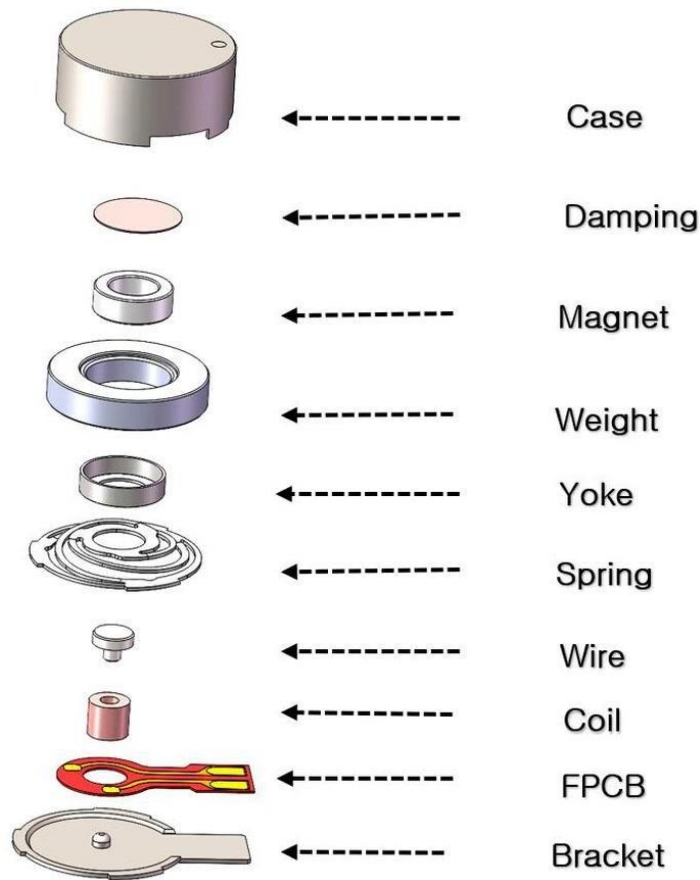


Figure 24 Componentes internos de un LRA (Ineed Motors, 2022).

Tabla 2 Descripción de los componentes de un LRA.

Part	Función
Case	Fija el muelle
Spring	Decide el peso y la frecuencia
Yoke	Trayectoria del campo magnético
Weight	Decide la fuerza de vibración del muelle y la frecuencia en propoción a la masa de peso
Magnet	Aplica fuerza electromagnética a la bobina
Coil	Aplica la fuerza electromagnética junto con el imán
Bracket	Fija la bobina y el FPC
Case	Fija el muelle

En lugar de transferir directamente la fuerza producida por la bobina móvil a la piel, el dispositivo optimiza el consumo de energía aprovechando la frecuencia de resonancia del muelle. Si la bobina de voz empuja la masa magnética contra el muelle a la frecuencia de resonancia de éste, el dispositivo puede producir una vibración de mayor amplitud de forma más eficiente. Dado que la bobina móvil es accionada por una corriente alterna, modelando la frecuencia y la amplitud de vibración deseadas, la frecuencia y la amplitud pueden modificarse independientemente

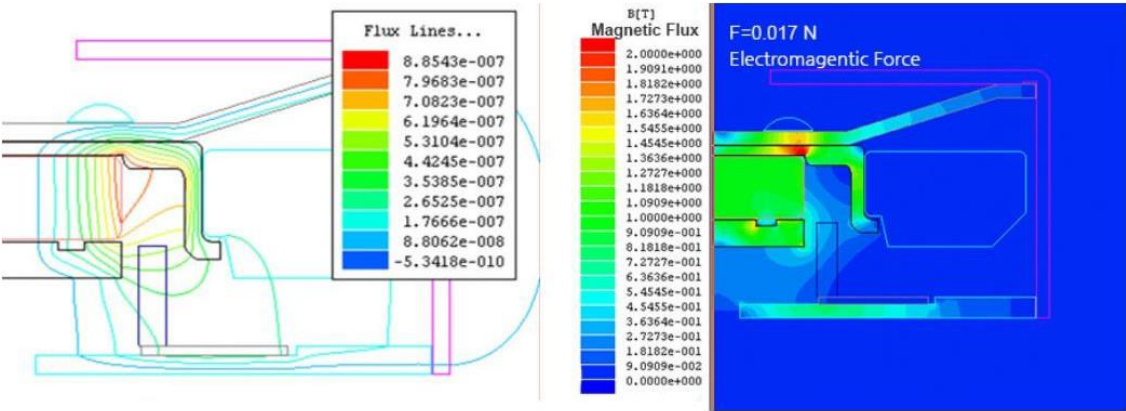


Figure 25 Dispersión de fuerza electromagnética sobre el háptico (NFP Motor, 2022).

Motor de masa giratoria excéntrica (ERM)

Un motor de masa giratoria excéntrica (ERM) es un motor eléctrico giratorio con una masa descentrada. Al girar, la masa se desplaza, creando la sensación de vibración.

Durante muchos años, los ERM han sido el tipo de motor háptico más popular por su bajo coste, sencillez y eficacia. Sin embargo, sus vibraciones carecen de precisión y pueden tardar en arrancar y detenerse, lo que limita la gama de sensaciones que pueden producir. Los dispositivos ERM pueden presentarse en envases de monedas o de barras.

Como muestra la figura 26, los ERM constan de una masa giratoria descentrada que, al girar, crea una vibración omnidireccional que se propaga por todo el dispositivo, como las alertas por vibración que recibes cuando tu móvil está en modo silencio o vibración.

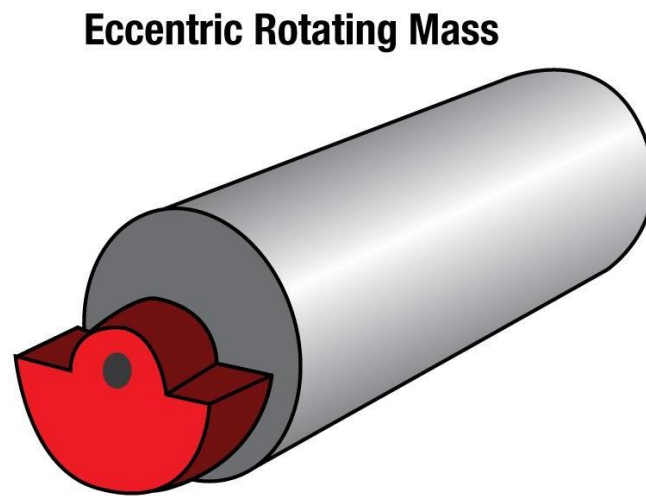


Figure 26 El Motor ERM (E. Siegle, 2011).

Por desgracia, debido a la construcción del ERM, la capacidad de crear perfiles de onda sofisticados es limitada. La frecuencia y la amplitud de cada onda están acopladas a la tensión de control de entrada, lo que te deja sólo una variable con la que jugar para crear diferentes efectos. En general, sólo puedes crear diferentes combinaciones de pulsaciones o velocidad.

El ERM se limita al funcionamiento básico de las vibraciones y no es adecuado para movimientos y efectos hápticos más complicados. Como motor, consume mucha energía y necesita más potencia para girar más rápido e inducir vibraciones mayores. La tensión de entrada controla la velocidad del motor, que a su vez afecta tanto a la amplitud como a la frecuencia de vibración.

Debido a su inercia inherente, consecuencia de la masa excéntrica, tarda un tiempo definido en acelerar hasta alcanzar la velocidad de giro deseada, y el tiempo de subida hasta el 90% del valor objetivo es del orden de 50 a 100 milisegundos, al igual que el tiempo de caída hasta detenerse. Esta latencia significa que el ECM no es adecuado para efectos hápticos de repetición rápida, como los de los juegos.

Los ERM se encuentran con mayor frecuencia en los mandos de juegos y recientemente se han encontrado en casos de uso en automoción, debido a su capacidad para producir vibraciones fuertes y activas.

Funcionamiento

El motor de vibración tiene un peso excéntrico y, a medida que gira, la fuerza centrípeta hace que el motor se mueva. La rotación se produce aplicando una corriente al devanado del inducido en el eje del motor. Dado que estos se encuentran dentro de un campo magnético, formado por imanes permanentes dentro del cuerpo del motor, se crea una fuerza por rotación del eje. Para asegurarse de que la rotación continúa en la misma dirección, se invierte la corriente en el devanado. Esto se logra mediante el uso de cepillos metálicos estáticos en las terminales del motor conectadas al conmutador que gira con el eje y los devanados. Diferentes partes del conmutador están conectadas a la escobilla durante la rotación, y la corriente se invierte, manteniendo la dirección de rotación.

Un motor de vibración ERM es un motor tradicional con un estator y un rotor y un peso excéntrico. El peso y la velocidad de la excéntrica determinan la amplitud y frecuencia de la vibración. La fuente de alimentación del motor del rotor suele ser de 1 V a 5 V, y la fuente de alimentación del motor cilíndrico de tamaño pequeño es generalmente de 3 V, y la resistencia terminal es de 35 ohmios. La estructura interna del motor ERM es la siguiente:

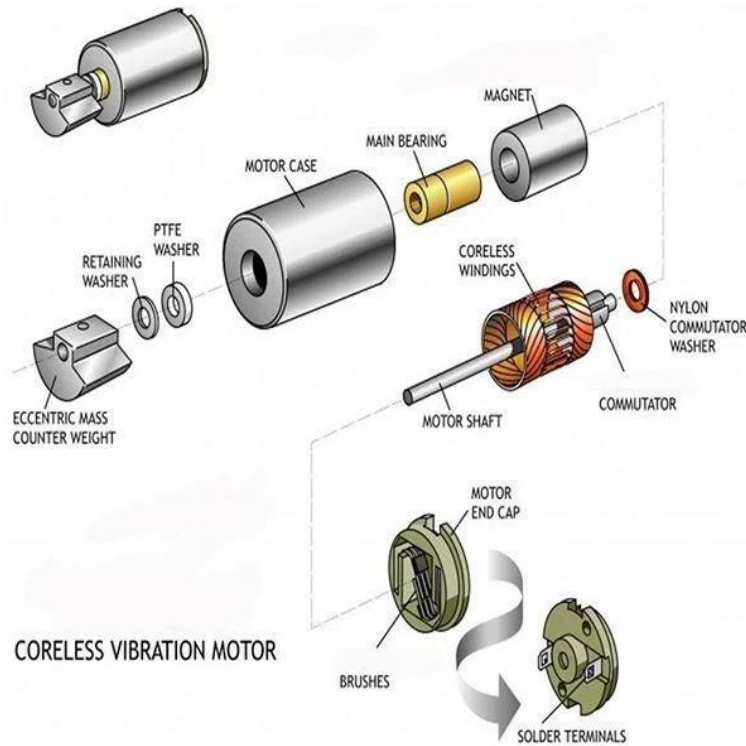
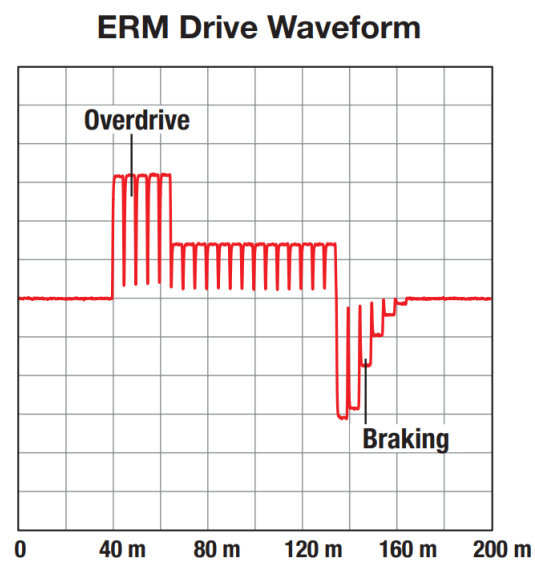


Figure 27 Diagrama explosivo del interior de un ERM (Ineed motors, 2022).

El motor ERM es accionado por corriente continua. Cuanto mayor sea el voltaje, más rápida será la frecuencia de vibración. Si desea detener la vibración, debe agregar un voltaje inverso. A continuación se muestra una forma de onda de voltaje de accionamiento de motor ERM típica.



Gráfica 4 Forma de onda para un ERM típica (Texas Instruments, 2013).

Diferencias entre LRA Y ERMS

En un motor ERM, la vibración sólo puede producirse después de que el motor alcance su velocidad de funcionamiento; incluso cuando se sobrecarga el motor para producir una aceleración más rápida, el motor puede necesitar entre 20 y 50 ms antes de alcanzar la intensidad de vibración deseada.

Por desgracia, el tiempo de parada de un LRA puede ser significativamente mayor que el de un motor ERM. Un LRA puede tardar hasta 300 ms en dejar de vibrar debido al continuo almacenamiento de energía cinética en el muelle interno durante el funcionamiento. Afortunadamente, también se puede utilizar un mecanismo de frenado activo para un LRA: al realizar un desplazamiento de fase de 180 grados de la señal de CA suministrada al actuador, la vibración se puede detener muy rápidamente (en aproximadamente 10 ms) produciendo una fuerza opuesta a la oscilación del muelle.

Los LRA mejoran la experiencia háptica global porque pueden personalizarse según las necesidades del usuario. Siguen evolucionando, reduciendo el consumo de energía y el espacio ocupado en la placa, al tiempo que producen efectos táctiles aún más nítidos y potentes.

Tabla 3 Tabla comparativa entre LRA Y ERM.

Construcción		Tamaño	Ventaja	Desventaja
Motor (tipo de rotación) -Cepillo y conmutador -Potencia: CC		$\varnothing 7$ - $\varnothing 12$ con espesor diferente	Precio bajo, fácil instalación, bajo nivel de ruido y tecnología madura	El tiempo de vida es difícil de aumentar a 1 millón de veces
Actuador lineal usando resonancia -Sin cepillo y conmutador -Potencia: onda sinusoidal de CA		$\varnothing 8 \times 2.5$ mm $\varnothing 8 \times 3.2$ mm $\varnothing 10 \times 3.0$ mm	Respuesta rápida, larga vida útil, fácil instalación, bajo nivel de ruido y frecuencia de vibración controlable	Proceso de fabricación complejo

Ventajas uso de LRA'S

- Los motores de vibración lineal (LRA) son capaces de vibrar a una frecuencia más estable con menos consumo de energía, ofreciendo una mejor calidad para la experiencia háptica. Compacto / ultra-delgado, de bajo ruido de alta eficiencia y fiabilidad, diseño mejorado caída de impacto.
- Puede adaptarse a una amplia variedad de usuarios diferentes tipos de aplicaciones.
- Rápida respuesta de vibración (tiempo de subida, tiempo de bajada) y minimizar el diseño de vibración residual
- Motor lineal vibra en dirección Z con una vida útil muy larga de 1 millón de ciclos en el modo de prueba 2s On 1s Off como un ciclo.
- Motor Lineal también tienen una muy fuerte aceleración / fuerza de vibración en su valor pico de aceleración alrededor de 1,7G.
- Para utilizar estos actuadores lineales resonantes, los usuarios necesitan añadir controladores IC en el PCB para activar estos motores.

TEXAS INSTRUMENTS

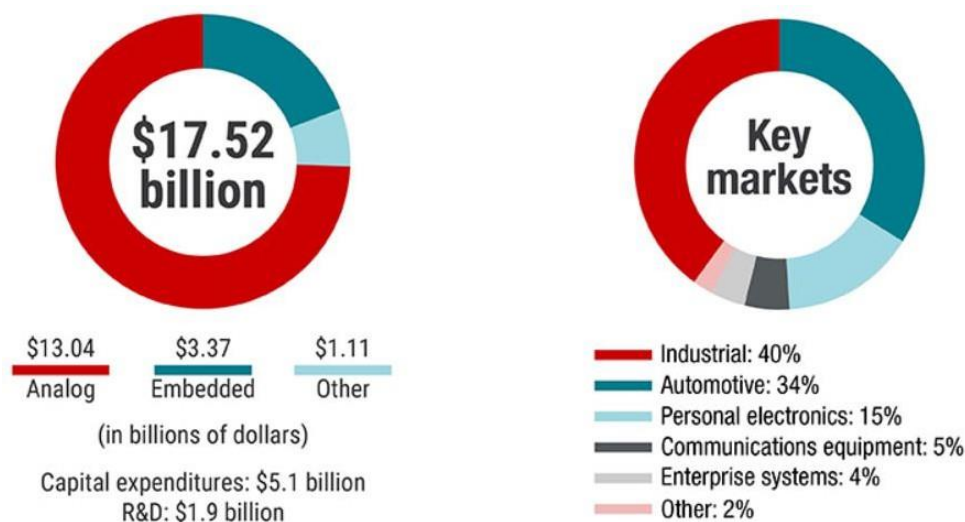
Es una empresa mundial de semiconductores que diseña, fabrica, prueba y vende chips de procesamiento analógico e integrado. Con unos 80.000 productos, ayuda a más de 100.000 clientes a gestionar de manera eficiente la energía, detectar y transmitir datos con precisión, y controlar o procesar aplicaciones básicas en fabricación industrial, automoción, electrónica de consumo, dispositivos de comunicaciones y sistemas empresariales²⁴.



Figure 28 Logotipo Texas Instruments (Texas Instruments, 2024).

Datos clave

- Fundada en 1930
- Sede central en Dallas, Texas
- Cotiza en bolsa (Nasdaq: TXN)
- Haviv Ilan es presidente y director general
- Se producen decenas de miles de millones de chips cada año



Gráfica 5 Ingresos 2023 Texas Instruments (Texas Instruments, 2024).

PROTOCOLOS DE COMUNICACIÓN

I2c Interface

El bus I2C emplea dos señales, SDA (datos) y SCL (reloj), para comunicarse entre los circuitos integrados de un sistema. El bus transfiere datos en serie, un bit cada vez. La dirección de 8 bits y los bytes de datos se transfieren con el bit más significativo (MSB) en primer lugar. Además, cada byte transferido en el bus es reconocido por el dispositivo receptor con un bit de reconocimiento.

Cada operación de transferencia comienza con el dispositivo maestro conduciendo una condición de inicio en el bus y termina con el dispositivo maestro conduciendo una condición de parada en el bus. El bus utiliza transiciones en el pin de datos (SDA) mientras el reloj está a nivel lógico alto para indicar las condiciones de inicio y parada. Una transición de alto a bajo en la señal SDA indica un inicio, y una transición de bajo a alto indica una parada. Las transiciones normales de bits de datos deben producirse dentro del tiempo bajo del periodo de reloj.

La figura 29 muestra una secuencia típica. El dispositivo maestro genera la dirección de esclavo de 7 bits y el bit de lectura-escritura (R/W) para iniciar la comunicación con un dispositivo esclavo. A continuación, el dispositivo maestro espera una condición de confirmación. El dispositivo esclavo mantiene baja la señal SDA durante el periodo de reloj de confirmación para indicar que ha recibido la señal. Cuando se produce este reconocimiento, el maestro transmite el siguiente byte de la secuencia. A cada dispositivo se le asigna una dirección esclava única de 7 bits más un bit R/W (1 byte). Todos los dispositivos compatibles comparten las mismas señales a través de un bus bidireccional que utiliza una conexión cableada-AND.

El número de bytes que pueden transmitirse entre las condiciones de arranque y parada no está limitado. Cuando se transfiere la última palabra, el maestro genera una condición de parada para liberar el bus.

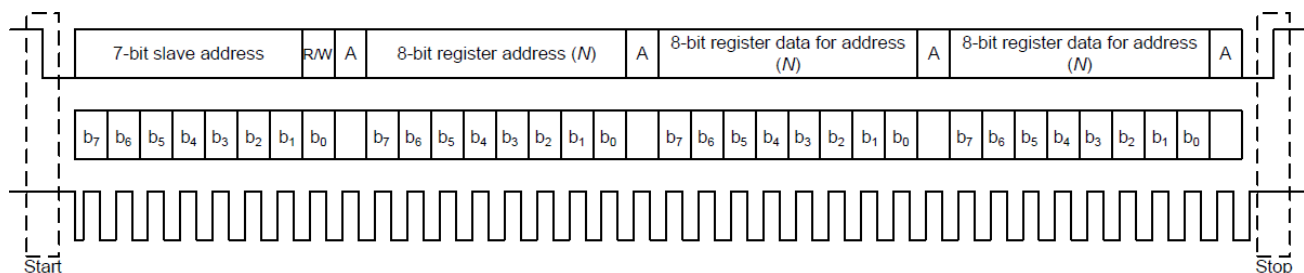


Figure 29 Secuencia típica de i2c (Texas Instruments, 2018).

Utilizar resistencias pull up externas para las señales SDA y SCL para establecer el nivel lógico-alto para el bus. Se recomiendan resistencias pull up con valores entre $660\ \Omega$ y $4,7\ k\Omega$

El dispositivo DRV26xx funciona como un I2C-esclavo de 1,8-V umbrales lógicos, pero puede funcionar hasta la tensión VDD. La dirección del dispositivo es 0x5A (7 bits), o 1011010 en binario, que equivale a 0xB4 (8 bits) para escritura y 0xB5 (8 bits) para lectura.

Transferencias de un byte y de múltiples bytes

La interfaz de control serie admite operaciones de lectura y escritura de un byte y de varios bytes para todos los registros. Durante las operaciones de lectura de bytes múltiples, el dispositivo DRV26xx responde con datos de un byte cada vez y comenzando en el registro con signo. El dispositivo responde mientras el dispositivo maestro siga respondiendo con acuses de recibo. El DRV26xx soporta direccionamiento I2C secuencial. Para las transacciones de escritura, se ha producido una transacción de escritura secuencial I2C si se emite un registro seguido de datos para ese registro así como para los registros restantes que le siguen. Para las transacciones de escritura secuencial I2C, el registro emitido sirve como punto de partida y la cantidad de datos transmitidos posteriormente antes de que se transmita una parada o un inicio determina cuántos registros se escriben.

UART

En esta comunicación, hay dos tipos de UART disponibles, es decir, UART de transmisión y UART de recepción, y la comunicación entre estos dos se puede realizar directamente entre sí. Para ello bastan dos cables para comunicar entre dos UARTs. El flujo de datos provendrá de los pines de transmisión (Tx) y recepción (Rx) de los UART. En UART, la transmisión de datos desde Tx UART a Rx UART se puede realizar de forma síncrona (no hay señal CLK para sincronizar los bits o/pa).

Los datos se pueden transmitir desde UART utilizando el bus de datos en forma paralela a otros dispositivos como microcontrolador, memoria, procesador, etc. Después de recibir los datos paralelos del bus, forma un paquete de datos agregando tres bits como inicio, parada y paridad. Lee el paquete de datos bit a bit y convierte los datos recibidos en forma paralela para descartar los tres bits del paquete de datos. Finalmente, el paquete de datos recibido por la UART se transfiere en paralelo al bus de datos recibido.

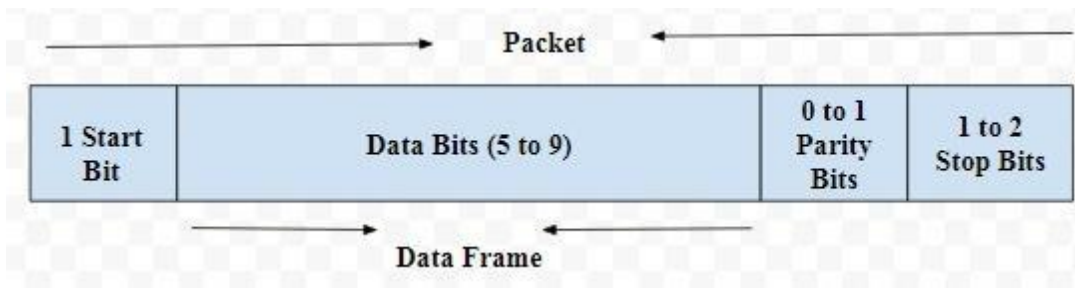


Figure 30 Transmisión de paquete de datos UART (El procus,2017).

PYTHON

Python es un lenguaje de programación informático que se utiliza a menudo para crear sitios web y software, automatizar tareas y realizar análisis de datos. Python es un lenguaje de propósito general, lo que significa que se puede utilizar para crear una variedad de programas diferentes y no está especializado en ningún problema específico.



Figure 31 Python (Medium, 2022).

QT DESIGNER

Qt Designer es la herramienta de Qt para diseñar y construir interfaces gráficas de usuario (GUIs) con Qt Widgets. Permite componer y personalizar ventanas o diálogos de una manera "lo que ves es lo que obtienes" (WYSIWYG), y probarlos usando diferentes estilos y resoluciones.

Los widgets y formularios creados con Qt Designer se integran a la perfección con el código programado, utilizando el mecanismo de señales y ranuras de Qt, para que se pueda asignar fácilmente comportamientos a los elementos gráficos. Todas las propiedades establecidas en Qt Designer pueden modificarse dinámicamente dentro del código. Además, funciones como la promoción de widgets y los plugins personalizados le permiten utilizar sus propios componentes con Qt Designer.



Figure 32 Logo Qt designer (Qt designer, 2018).

VISUAL STUDIO

Visual Studio es una herramienta de desarrollo eficaz que permite completar todo el ciclo de desarrollo en un solo lugar. Es un entorno de desarrollo integrado (IDE) completo que puede usar para escribir, editar, depurar y compilar el código. A continuación, implemente la aplicación. Visual Studio incluye compiladores, herramientas de finalización de código, control de código fuente, extensiones y muchas más características para mejorar cada fase del proceso de desarrollo de software.

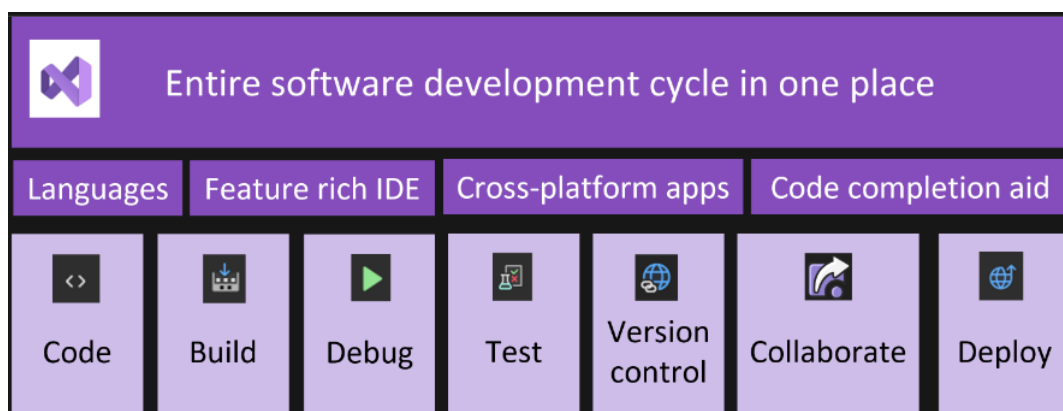


Figure 33 Capacidades de desarrollo (Microsoft, 2024).

CAPITULO III

DESARROLLO DEL PROYECTO

DESCRIPCIÓN GENERAL DEL PROYECTO

A continuación se presenta el desarrollo de una interfaz prototipo para un banco de pruebas que es capaz de programar diferentes efectos de vibración en diferentes actuadores.

Para integrar los hápticos en un prototipo, en este caso una interfaz de usuario se necesitan tres elementos: el actuador háptico, que traduce las señales eléctricas en movimiento físico; el controlador electrónico (driver), que suministra el voltaje y la corriente al actuador; y el MCU, normalmente integrado en un microcontrolador, que indica al controlador electrónico el patrón específico de voltaje/corriente que debe suministrar al actuador (Figura 28).

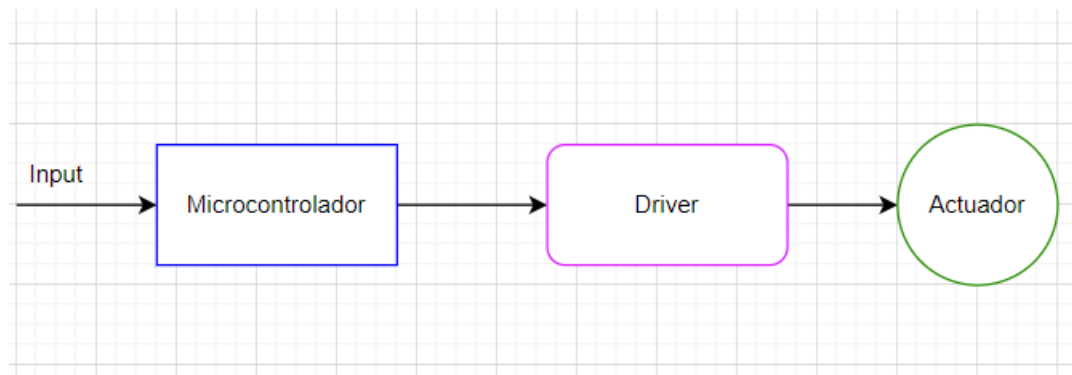


Figure 34 Esquema simplificado de integración de un háptico (Tlacomulco A. , 2024).

Propuesta de diagrama general del proyecto

Para comenzar se describe el diseño del diagrama a bloques del sistema general, así como los protocolos de comunicación que se utilizaran.

Dicho sistema consta de una HMI que puede ser ejecutada en cualquier equipo de cómputo que se comunica a través del protocolo de comunicación UART con el microcontrolador, a su vez el Microcontrolador (Maestro) se comunica con el driver (Esclavo) a través del protocolo de comunicación I2C que finalmente acciona cualquiera de los dos tipos de hápticos en función de los parámetros programados.

A continuación se muestra el diseño del sistema.

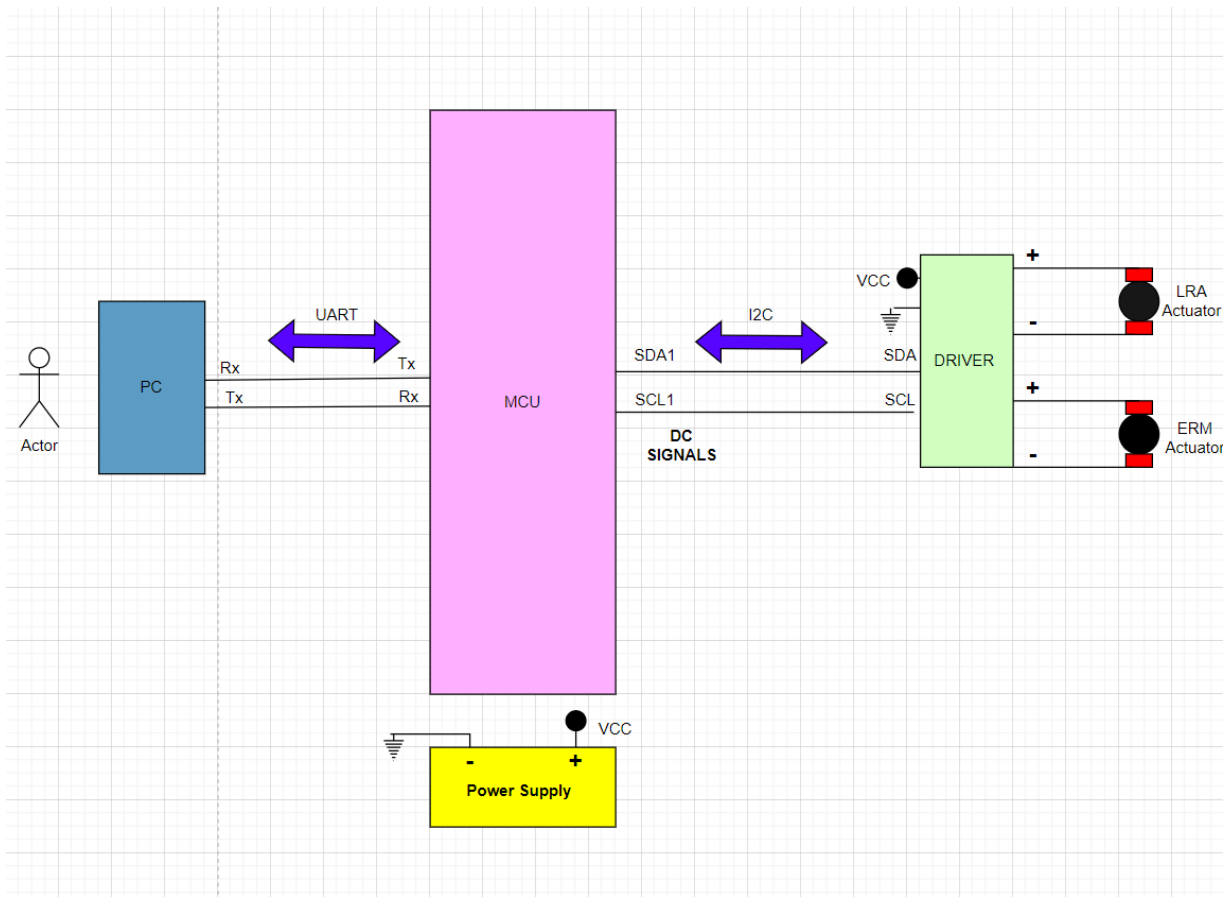


Figure 35 Diagrama a bloques del sistema general (Tlacomulco A. , 2024).

Definición y selección de materiales (Hardware)

Actuadores Hápticos

Dentro de la variedad de hápticos que ofrecen distintas empresas como ineed motors o Texas instruments se ha optado por elegir los que a continuación se describen.

Para LRA:

Los diseños que requieren tiempos de respuesta más rápidos, una mayor eficiencia energética y una vibración más fuerte pueden utilizar el dispositivo LRA HD-LA0803-LW10-R de 8 mm de diámetro por 3.2 mm de altura de PUI Audio. La resistencia del HD-LA0803-LW10-R está especificada en 25 Ω ($\pm 15\%$). El consumo de energía del HD-LA0803-LW10-R es de unos 180 milivatios (mW), (2VRMS x 90 mA), este dispositivo LRA tiene un rango de temperatura de funcionamiento de -20 a +70 °C.

Para ERM:

Los diseños que requieren tiempos de respuesta más rápidos, una mayor eficiencia energética y una vibración más fuerte pueden utilizar el dispositivo HD-EM0802-LW30-R de PUI Audio.

HAPTIC	MODELO	
LRA	HD-LA0803-LW10-R	
ERM	HD-EM0802-LW30-R	

Aunque las anteriores solo serán para probar los diferentes efectos, el sistema debe ser capaz de poder controlar cualquier actuador háptico y los efectos de vibración serán diferentes pues cada actuador tiene fuerzas de vibración diferentes.

Descripción del funcionamiento del driver

El dispositivo DRV26xx es un controlador háptico que se basa en la FEM de retorno producida por un actuador para proporcionar un sistema de bucle cerrado que ofrece un control extremadamente flexible de los actuadores LRA y ERM a través de un bus compartido compatible con I2C o de un controlador de bucle cerrado.

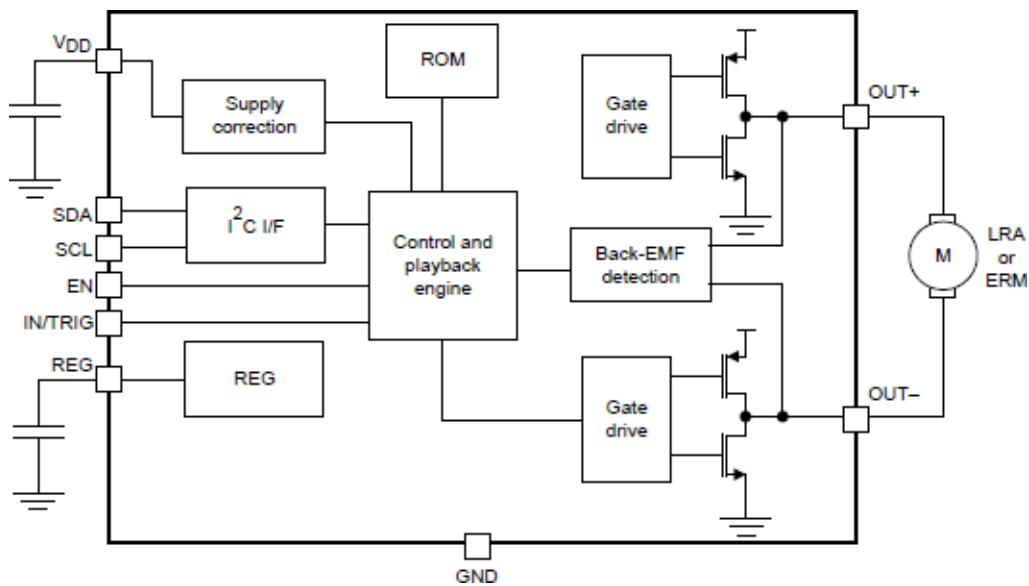


Figure 36 Diagrama a bloques simplificado del driver TI (Texas Instruments, 2018).

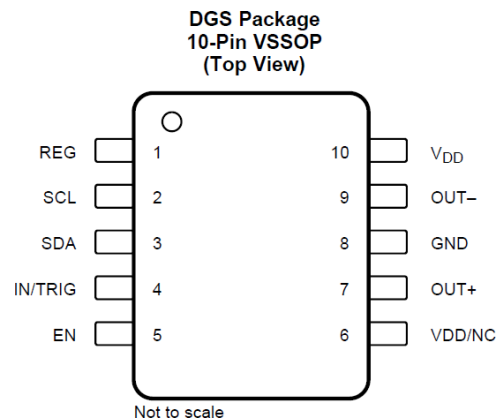


Figure 37 Pinout DR26xx (Texas Instruments, 2024).

Driver

Para la selección del driver que permite el control de los efectos hápticos se ha decidido utilizar un driver de la familia de Texas Instruments pues como previamente se ha mencionado en el marco teórico han sido pioneros en el desarrollo de esta nueva tecnología.

Texas Instruments es líder en tecnología de controladores para aplicaciones hápticas. Sus controladores aumentan el rendimiento y reducen el tiempo de integración de la tecnología háptica en su

sistema. Sólo TI ofrece una solución táctil total con una gama completa de controladores hápticos para cualquier aplicación. La tecnología de controladores de TI mejora la experiencia táctil al mejorar realmente el rendimiento de estos actuadores.

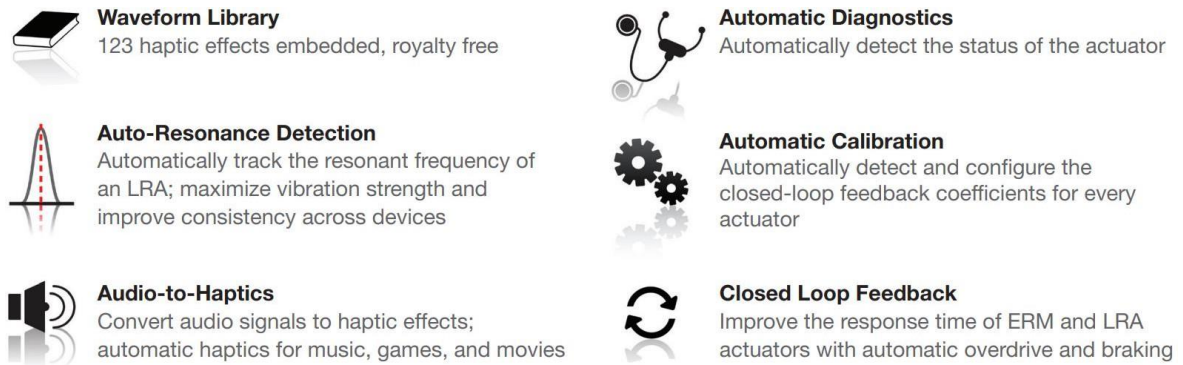


Figure 38 Características que ofrece Texas Instrument, reducen tiempos de integración (TI, 2013)

Tabla 4 Selección de Driver de la familia Texas Instruments

Features	DRV2603	DRV2604	DRV2605	DRV2605 (1.8 V)
ERM and LRA Drive Modes	✓	✓	✓	✓
Drive Strength Independent of Supply	✓	✓	✓	✓
Efficient H-Bridge Driver Implementation	✓	✓	✓	✓
Automatic Input Level Translation	✓	✓	✓	✓
LRA Auto Resonance Tracking and Smart Loop	✓	✓	✓	✓
I ² C Control for Device Parameters		✓	✓	✓
ERM Mode With Smart Loop		✓	✓	✓
Automatic Actuator Calibration and Diagnostic		✓	✓	✓
External Trigger Mode and Multi-Device Trigger		✓	✓	✓
Waveform RAM (2 kB)		✓		
Real-Time I ² C Playback Mode		✓	✓	✓
Integrated Immersion Library			✓	✓
Audio-to-Haptics			✓	✓
1.8 V Support				✓
Package Type	QFN	WCSP	WCSP	QFN

Microcontrolador

La selección de un microcontrolador adecuado para controlar el driver de Texas Instruments es una decisión crucial en el diseño de sistema propuesto. A continuación, se detallan las consideraciones que se tuvieron en cuenta para esta selección:

- *Compatibilidad de Señales:* El driver 26XX está diseñado para trabajar con señales de control específicas. SDA Y SCL, se debe elegir un microcontrolador que pueda generar estas señales de forma adecuada para garantizar el funcionamiento correcto del sistema.
- *Procesamiento y Velocidad:* El microcontrolador debe ser capaz de realizar cálculos y decisiones en tiempo real. La velocidad de procesamiento es un factor clave, especialmente en aplicaciones que requieren respuestas rápidas.
- *Consumo de Energía:* En dispositivos portátiles como un banco de pruebas, el consumo energético del microcontrolador debe ser mínimo. Es importante seleccionar un microcontrolador que ofrezca modos de bajo consumo y que sea eficiente en su operación.
- *Memoria y Almacenamiento:* La cantidad de memoria RAM y ROM disponible en el microcontrolador determinará la complejidad del software que se puede implementar.
- *Voltaje de Operación:* El microcontrolador debe ser compatible con el rango de voltaje de operación del driver 26XX (2.2 V a 5.2 V). Esto es vital para evitar daños y asegurar un funcionamiento eficiente.
- *Salida de Señal:* El microcontrolador debe poder proporcionar las señales de control necesarias (PWM, lógica alta/baja) que el driver 26xx requiere para su operación.
- *Facilidad de Programación:* La disponibilidad de herramientas de desarrollo, bibliotecas y ejemplos puede facilitar la implementación y reducir el tiempo de desarrollo.
- *Costo y Disponibilidad:* Se debe elegir un microcontrolador que sea rentable, barato y fácil de adquirir para el éxito del proyecto. Pues al ser un prototipo la implementación de un microcontrolador costoso puede no ser factible para el desarrollo de un producto mínimo viable.

Teniendo todas las consideraciones anteriores en cuenta se ha optado por utilizar el microcontrolador Arduino Nano, ya que basados en que es el desarrollo de un primer prototipo los costos de implementación no deben ser altos, además de que dicho microcontrolador cumple con todos los criterios mencionados anteriormente.

El microcontrolador Arduino NANO utiliza el microprocesador ATmega328. Este procesador funciona a una frecuencia de reloj de 16 MHz y dispone de 2 kB de RAM, 1 kB de EEPROM y 32 kB de memoria Flash (para programación y almacenamiento de datos).

Alimentación

El microcontrolador Arduino NANO funciona en un rango de tensión de 7-12 V gracias a su regulador de tensión integrado. El microprocesador, por su parte, funciona a una tensión de 5V. En funcionamiento normal, el microcontrolador consume hasta 19mA (si no recibe alimentación) y puede aceptar una corriente máxima de 40mA en cada uno de sus pines IO.

Pinout

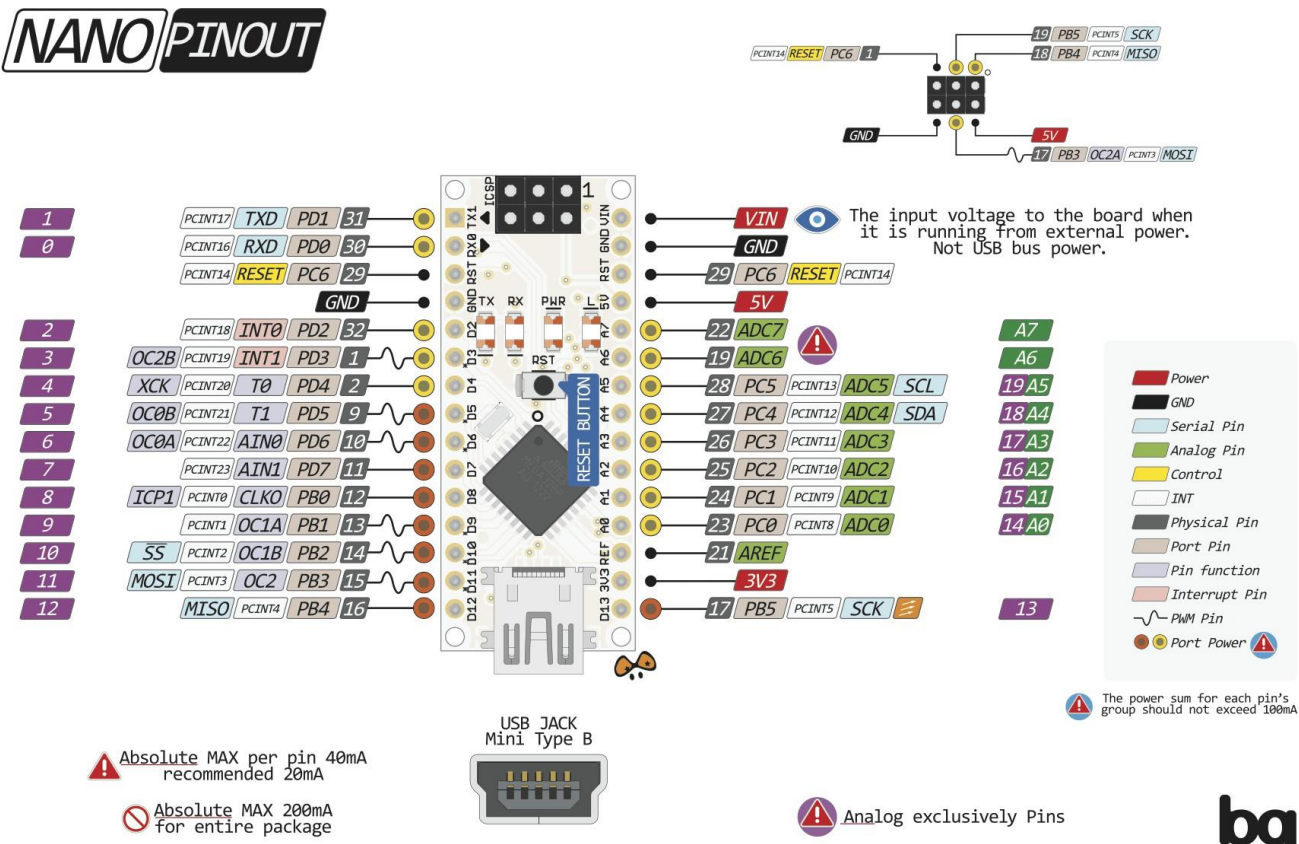


Figure 39 PINOUT Arduino Nano (Rochester Electronic, 2024).

Definición y selección de herramientas (Software)

En cuanto a la parte de la implementación del software se deben tener en consideración las herramientas que se tendrán en cuenta para el desarrollo del sistema dividiéndolas en 3 partes importantes: el desarrollo del diseño de la interfaz, el desarrollo del código de control del driver y el desarrollo del código de la unión de las funcionalidades del microcontrolador con la parte desarrollada del HMI.

Tabla 5 Selección de las herramientas para desarrollo de software

Función	Software	Lenguaje de desarrollo
Diseño de la interfaz	QT designer	Python
Código del microcontrolador	Arduino IDE	Arduino
Union de Interfaz + Cod. Micro	Visual Studio	Python

DESARROLLO DE CASOS DE PRUEBA PRELIMINARES

Validación del funcionamiento del LRA

Es importante realizar una primera validación del funcionamiento de uno de los hápticos , para ello se procede a la implementación de un código de prueba para su verificación, en este caso para un háptico de tipo LRA.

A continuación se describe el procedimiento que se llevo a cabo.

1. Descargar el entorno de programación IDE Arduino.
2. Descargar las librerías necesarias para el funcionamiento del DRV26xx, las cuales son; Adafruit 26xx Library, que permite el acceso a los registros de memoria del driver y a los efectos de vibracion predefinidos y respectivamente Adafruit BusIO la cual es una librería de ayuda a abstraer transacciones y registros I2C y SPI.
3. Realizar las conexiones de pines necesarias entre el microcontrolador y el driver que se muestran a continuación.

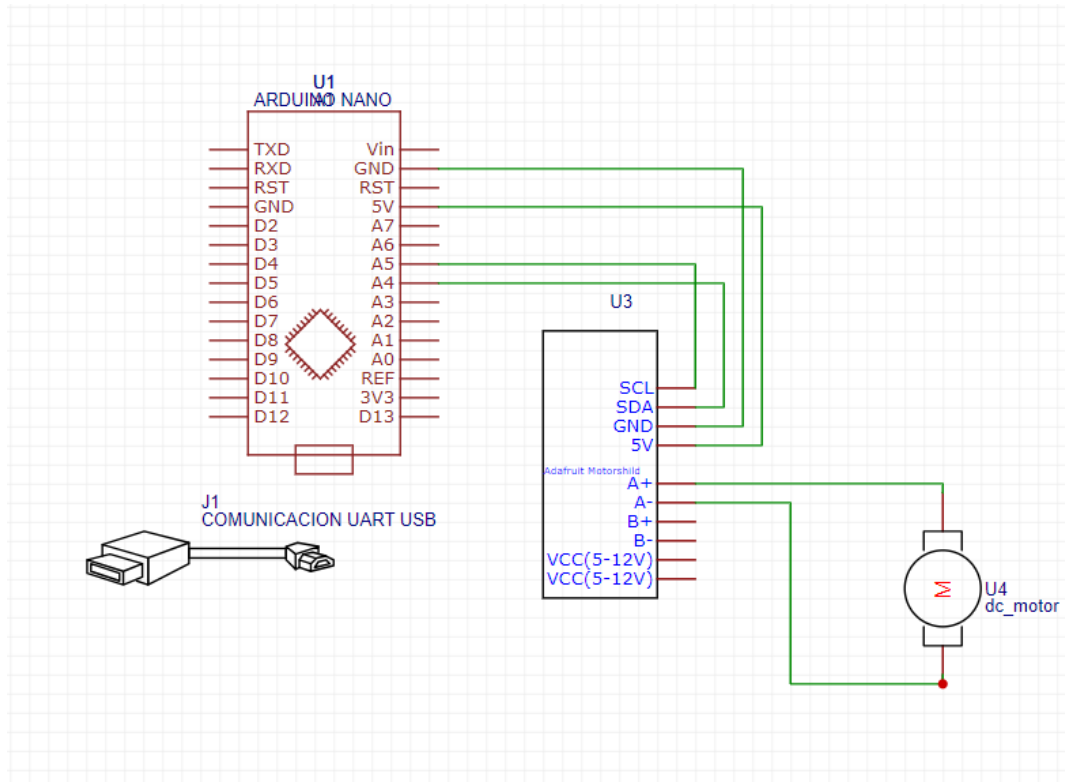


Figure 40 Diagrama de conexiones para código de prueba de LRA (Tlacomulco A. ,2024).

4. Setear el Puerto, la placa de desarrollo y el baud rate en el IDE de Arduino.

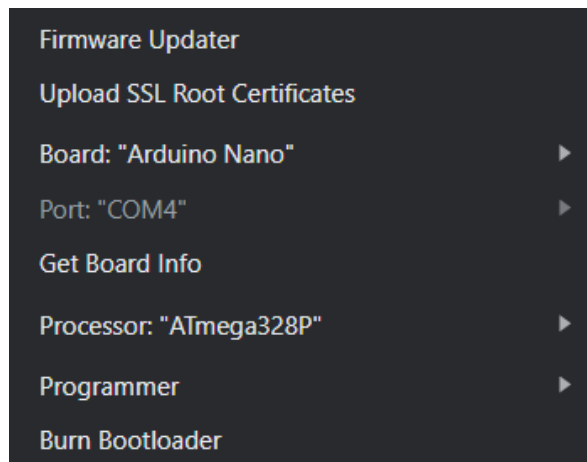


Figure 41 Seteo del microcontrolador en arduino, (Tlacomulco A. , 2024).

5. Cargar el Código de prueba que a continuación se presenta el cual es un ejemplo de prueba de la misma libreria de Adafruit (Ver Código complete en el anexo A).

```

basic.ino
1 #include <Wire.h>
2 #include "Adafruit_DRV2605.h"
3
4 Adafruit_DRV2605 drv;
5
6 void setup() {
7   Serial.begin(9600);
8   Serial.println("Adafruit DRV2605 Basic test");
9   if (!drv.begin()) {
10    Serial.println("Could not find DRV2605");
11    while (1) delay(10);
12  }
13
14  drv.selectLibrary(1);
15
16  // I2C trigger by sending 'go' command
17  // default, internal trigger when sending GO command
18  drv.setMode(DRV2605_MODE_INTTRIG);
19 }
20
21 uint8_t effect = 1;
22
23 void loop() {
24   // Serial.print("Effect #"); Serial.println(effect);
25
26   if (effect == 1) {
27     Serial.println("11.2 Waveform Library Effects List");
28   }
29
30   if (effect == 1) {
31     Serial.println(F("1 Strong Click - 100%"));
32   }
33   if (effect == 2) {
34     Serial.println(F("2 Strong Click - 60%"));
35   }
36   if (effect == 3) {
37     Serial.println(F("3 Strong Click - 30%"));
38   }
39   if (effect == 4) {
40     Serial.println(F("4 Sharp Click - 100%"));
41   }

```

Figure 42 Código de ejemplo para prueba de LRA's (Tlacomulco A. , 2024).

```

Adafruit DRV2605 Basic test
11.2 Waveform Library Effects List
1 - Strong Click - 100%
2 - Strong Click - 60%
3 - Strong Click - 30%
4 - Sharp Click - 100%
5 - Sharp Click - 60%
6 - Sharp Click - 30%
7 - Soft Bump - 100%
8 - Soft Bump - 60%
9 - Soft Bump - 30%
10 - Double Click - 100%
11 - Double Click - 60%
12 - Triple Click - 100%
13 - Soft Fuzz - 60%
14 - Strong Buzz - 100%
15 - 750 ms Alert 100%
16 - 1000 ms Alert 100%
17 - Strong Click 1 - 100%
18 - Strong Click 2 - 80%
19 - Strong Click 3 - 60%
20 - Strong Click 4 - 30%
21 - Medium Click 1 - 100%
22 - Medium Click 2 - 80%
23 - Medium Click 3 - 60%
24 - Sharp Tick 1 - 100%
25 - Sharp Tick 2 - 80%
26 - Sharp Tick 3 - 60%
27 - Short Double Click Strong 1 - 100%
28 - Short Double Click Strong 2 - 80%
29 - Short Double Click Strong 3 - 60%
30 - Short Double Click Strong 4 - 30%
31 - Short Double Click Medium 1 - 100%
32 - Short Double Click Medium 2 - 80%
33 - Short Double Click Medium 3 - 60%
34 - Short Double Sharp Tick 1 - 100%
35 - Short Double Sharp Tick 2 - 80%
36 - Short Double Sharp Tick 3 - 60%
37 - Long Double Sharp Click Strong 1 - 100%
38 - Long Double Sharp Click Strong 2 - 80%
39 - Long Double Sharp Click Strong 3 - 60%
40 - Long Double Sharp Click Strong 4 - 30%
41 - Long Double Sharp Click Medium 1 - 100%
42 - Long Double Sharp Click Medium 2 - 80%
43 - Long Double Sharp Click Medium 3 - 60%
44 - Long Double Sharp Tick 1 - 100%
45 - Long Double Sharp Tick 2 - 80%
46 - Long Double Sharp Tick 3 - 60%
47 - Buzz 1 - 100%
48 - Buzz 2 - 80%
49 - Buzz 3 - 60%
50 - Buzz 4 - 40%
51 - Buzz 5 - 20%
52 - Pulsing Strong 1 - 100%
53 - Pulsing Strong 2 - 60%
54 - Pulsing Medium 1 - 100%
55 - Pulsing Medium 2 - 60%
56 - Pulsing Sharp 1 - 100%
57 - Pulsing Sharp 2 - 60%
58 - Transition Click 1 - 100%
59 - Transition Click 2 - 80%
60 - Transition Click 3 - 60%
61 - Transition Click 4 - 40%
62 - Transition Click 5 - 20%
63 - Transition Click 6 - 10%
64 - Transition Click 7 - 10%

```

Figure 43 Salida en el monitor serial de los efectos reproducidos.

Después de haber realizado las conexiones necesarias y cargar el código al microcontrolador se pudo verificar el funcionamiento no solo del actuador háptico, si no también del del microcontrolador y de la comunicación en general de estos dos componentes.

Desarrollo de código para casos de prueba de los efectos hápticos

Modo Waveform Sequencer

El secuenciador de formas de onda pone en cola identificadores de formas de onda para su reproducción. Un identificador de forma de onda es un valor entero que hace referencia a la posición del índice de una forma de onda en la biblioteca ROM.

La reproducción comienza en la dirección de registro 0x04 cuando el usuario activa el bit GO (registro 0x0C). Cuando finaliza la reproducción de esa forma de onda, el secuenciador de formas de onda reproduce el identificador de forma de onda guardado en el registro 0x05 si la siguiente forma de onda es distinta de cero. El secuenciador de formas de onda continúa de esta manera hasta que alcanza un valor de identificador de cero o hasta que se reproducen los ocho identificadores (direcciones de registro 0x04 a 0x0B), cualquiera que sea el escenario que se alcance primero.

El rango del identificador de forma de onda es de 1 a 123. El MSB de cada registro de secuencia puede utilizarse para implementar un retardo entre formas de onda de secuencia. Cuando el MSB está alto, los bits 6-0 indican la duración del tiempo de espera. El tiempo de espera para ese paso se convierte entonces en $\text{WAV_FRM_SEQ}[6:0] \times 10 \text{ ms}$.

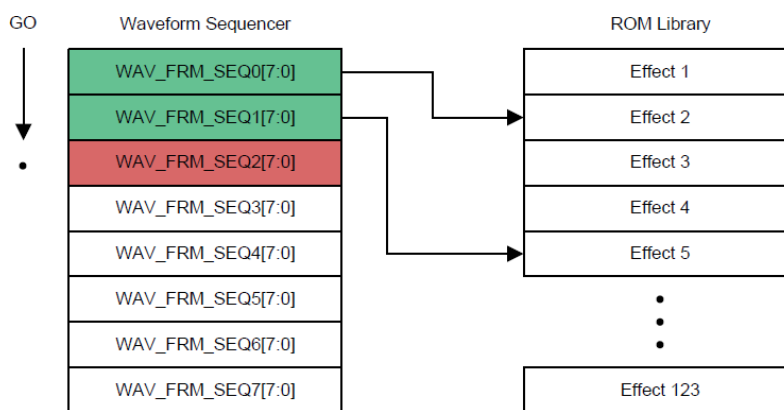


Figure 44 Programación del secuenciador de forma de onda (Texas Instruments Incorporated, 2018).

El código presentado anteriormente reproduce los 123 efectos de onda que contienen las librerías de driver de manera consecutiva permitiendo mostrar la gama de efectos de vibración que vienen pre definidos, sin embargo este código los reproduce todos de manera consecutiva y en tiempos muy cortos, por lo tanto para poder desarrollar un demostrador que permita experimentar estos efectos en una variedad de combinaciones es necesario hacer casos de prueba con diferentes iteraciones en tiempos y combinaciones de los mismos para familiarizarnos con las sensaciones que producen.

A continuación se presenta el desarrollo de un código que permite ingresar el número de efecto que se desea sentir entre 1 y 123, seguidamente este se reproducirá indefinidamente hasta seleccionar uno nuevo o en su defecto, tiene una función que permite detenerlo ingresando la letra “S” .

```

efectosrepetitivosconstop.ino
1/
18 void loop() {
19   static int effect = 0; // Variable estática para mantener el efecto actual
20   static unsigned long lastTime = 0; // Variable para el temporizador
21
22   // Espera hasta que se reciba un número a través del Serial
23   if (Serial.available() > 0) {
24     String input = Serial.readStringUntil('\n'); // Lee la entrada hasta el salto de línea
25
26     if (input.equals("s")) { // Verifica si la entrada es 's'
27       effect = 0; // Detiene el efecto
28       Serial.println("Effect stopped.");
29     } else {
30       int newEffect = input.toInt(); // Convierte la entrada a un número
31
32       if (newEffect >= 1 && newEffect <= 123) { // Verifica si el ID del efecto está en el rango válido
33         effect = newEffect; // Actualiza el efecto si es válido
34         Serial.print("Playing Effect #");
35         Serial.println(effect);
36       } else {
37         Serial.println("Invalid effect ID. Please enter a number between 1 and 123.");
38         effect = 0;
39       }
40     }
41
42     // Limpia el buffer de entrada
43     while (Serial.available() > 0) {
44       Serial.read();
45     }
46   }
47 }

```

Output Serial Monitor X

Message (Enter to send message to 'Arduino Nano' on 'COM4')

```

Playing Effect #119
Playing Effect #67
Invalid effect ID. Please enter a number between 1 and 123.
Effect stopped.
Basic test for each effect: Please enter a number between 1 and 123 or 's' to stop.
Playing Effect #120
Playing Effect #67
Invalid effect ID. Please enter a number between 1 and 123.
Playing Effect #56
Effect stopped.

```

Figure 45 Código para casos de prueba d eefectos (Tlacomulco A., 2024).

El código completo puede verse en el anexo B.

Después de una serie de pruebas entre los efectos que se pueden experimentar, se llegaron a los resultados presentados en la primera sección del capítulo IV resultados, en cuanto a la sensación experimentada con cada uno de ellos, pudiendo agruparlos rescatando los efectos que producían “mayor diferencia entre sensaciones”.

DESARROLLO DE SOFTWARE

Desarrollo del código para el microcontrolador

Para poder programar un código de control para el DRV26xx es necesario comprender como es que funciona el mismo, que modos de reproducción de onda existe, en que modos de operación trabaja, cuales y que funciones tienen sus registros de memoria, que valores deben modificarse o calcularse para obtener diferentes intensidades de vibración etc.

La reproducción de un efecto háptico puede producirse de múltiples maneras. El modo PWM, el modo RTP, el modo audio a vibración y el modo de entrada analógica pueden proporcionar la forma de onda en tiempo real. Las formas de onda también se pueden reproducir desde la ROM, en cuyo caso se utiliza el modo de reproducción de formas de onda y la forma de onda se reproduce mediante un bit GO interno (registro 0x0C), o mediante un disparador externo.

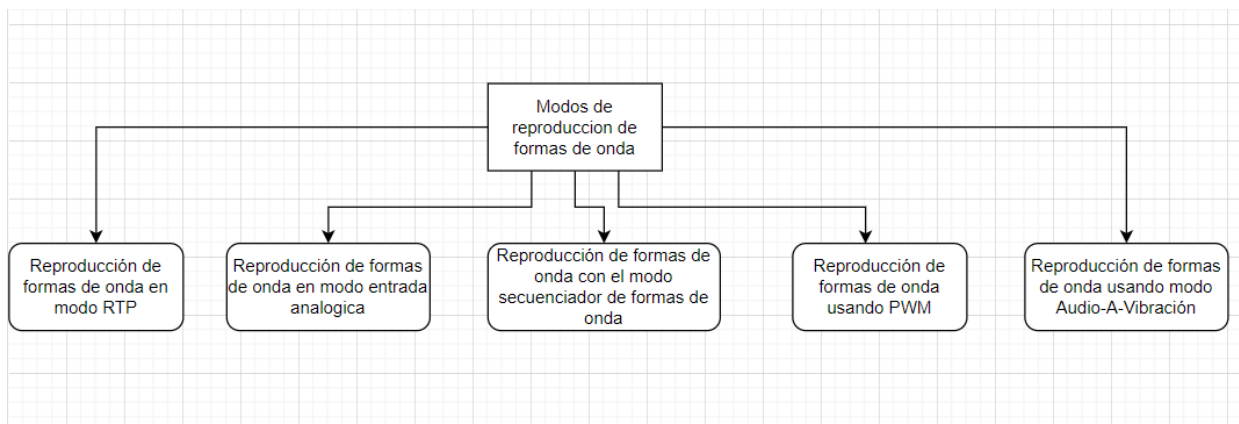


Figure 46 Modos de operación del driver (Tlacomulco A., 2024).

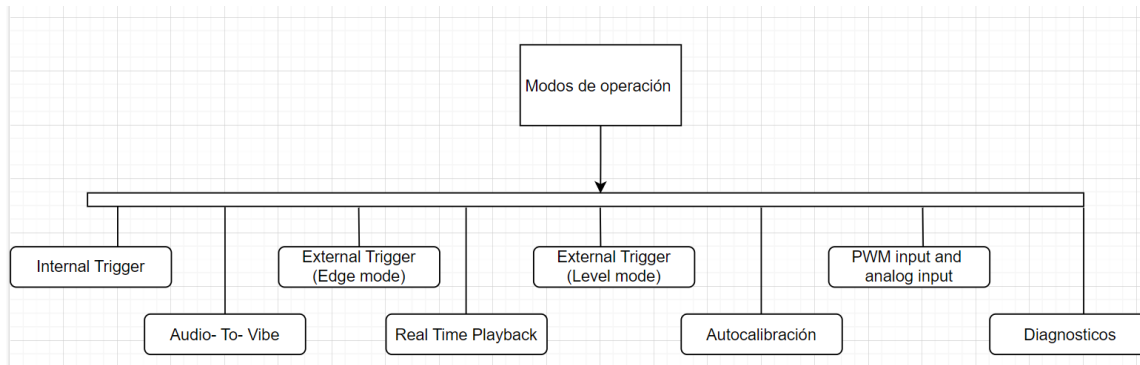


Figure 47 Modos de operación del driver (Tlacomulco A., 2024).

El desarrollo de este prototipo se centrará solo en la reproducción de formas de onda a través del secuenciador de formas de onda para LRA y ERMS. En el modo de operación se estará programando el trigger interno, y de la misma manera también se utilizara el modo de auto calibración.

Una vez entendido lo anterior se procede a explicar el proceso de programación que debe seguir para el desarrollo del código de control del driver.

Inicialización del dispositivo

Para iniciar el dispositivo y comenzar una transacción I2C:

1. 1. Aplicar alimentación al pin VDD.
2. 2. Poner el pin ENABLE a nivel alto(High) o conecte el pin ENABLE a VDD.
3. El dispositivo realizará una secuencia de arranque rápida (250 μ s) y pasará al modo STANDBY.
4. Salir del modo STANDBY poniendo a cero el bit STANDBY en el registro 0x01. Una sola escritura en el registro 0x01 puede desactivar STANDBY y activar el MODE del dispositivo.

Internamente al dispositivo, se producirá una secuencia de arranque cada vez que se aplique alimentación. Este ciclo de encendido tarda menos de 250 μ s, una vez que la alimentación es estable. Después de los 250 μ s iniciales, el dispositivo está listo para funcionar y pasará por defecto al estado de STANDBY (STANDBY = 1).

Si se produce un intento de enviar una transacción I2C al dispositivo antes de que el dispositivo complete la secuencia de inicio interna, el dispositivo devolverá un nACK. Si se produce un nACK, reintente la transacción hasta que se produzca un ACK exitoso.

Modo de espera **STAND BY**

El dispositivo dispone de un modo de bajo consumo que puede activarse o desactivarse por hardware o software. Para controlar el actuador, el pin ENABLE debe estar alto y el bit STANDBY (registro 0x01, bit 6) debe estar bajo.

Tabla 6 Ajustes de control para modo STANDBY

EN PIN	STANDBY BIT	ESTADO DEL DISPOSITIVO
High	0	Enabled
Low	X	Low-Power Mode
X	1	Low-Power Mode

Si el pin ENABLE está bajo o el bit STANDBY es "1", el dispositivo entrará en modo de bajo consumo. Cuando está en modo de bajo consumo, la circuitería interna se desactiva y algunos registros son inaccesibles; sin embargo, los datos de los registros del dispositivo y de la ROM permanecen. Para salir del modo STANDBY, el pin ENABLE debe estar a nivel alto y el bit STANDBY a nivel bajo.

Acceso al mapa de registros I2C

El DRV26xx está controlado por una serie de registros I2C. Para acceder a estos registros, primero debe poner el pin EN alto y utilizar la dirección I2C de 7 bits 0x5A. La Tabla 7 muestra la dirección de 7 bits, la dirección de lectura I2C y la dirección de escritura I2C.

Tabla 7 Ajustes de registros para I2C.

	HEX	BINARY
Dirección 7-bit I2C	0x5A	101 1010
Dirección 7-bit I2C + Bit escritura	0xB4	1011 0100
Dirección 7-bit I2C + Bit lectura	0xB5	1011 0101

Tensión nominal y de sobremarcha

Los registros de tensión nominal y de sobremarcha establecen las tensiones de escala completa y de sobremarcha utilizadas en los datos de la forma de onda. Por ejemplo, si la tensión nominal se fija en 3,3 V y una forma de onda requiere una salida del 100%, la tensión de salida será de 3,3 V. Para la tensión de sobremarcha, si la tensión de sobremarcha se fija en 4 V y el acelerómetro parte de una aceleración cero o se encuentra en una transición de baja aceleración a alta aceleración, el controlador indicará sobremarcha y utilizará 4 V.

ERM – Tensión nominal y de sobremarcha

- 1.- Si se quiere utilizar las bibliotecas de formas de onda integradas en el DRV2605, utilizar el bucle abierto, ya que las formas de onda se sintonizan utilizando el modo de bucle abierto.
2. Lazo cerrado continúe en el paso 3. Lazo abierto vaya al paso 5.
3. Tensión nominal en lazo cerrado: Ajuste el registro RatedVoltage (0x16) a la tensión nominal especificada en la hoja de datos del actuador. Utilice la siguiente ecuación para convertir la tensión al valor binario apropiado.

$$RatedVoltage (0x16) = \frac{V_{rated\ voltage} \times 255}{5.36V}$$

4. Voltaje de Sobremarcha de Lazo Cerrado: Ajuste la tensión de sobremarcha (ODClamp) (0x17) a la tensión de sobremarcha del actuador especificada en la hoja de datos del actuador. Utilice la siguiente ecuación para convertir la tensión al valor binario apropiado.

$$V_{peak} = V_{overdrive} \times \frac{DriveTime + IDissTime + BlankingTime}{Drivetime - 300uS}$$

5. Tensión nominal en bucle abierto: En el modo de bucle abierto, la tensión nominal no está referenciada por el motor de control. A diferencia del lazo cerrado, donde el 100% de la salida es igual a la tensión nominal, en el modo de lazo abierto, el 100% de la salida es igual a la tensión de sobremarcha.

6. Voltaje de sobremarcha en lazo abierto: Ajuste la Tensión de Sobremarcha (ODClamp) (0x17) a la tensión de sobremarcha del actuador especificada en la hoja de datos del actuador. Utilice la siguiente ecuación para convertir la tensión al valor binario apropiado.

$$ODClamp (0x17) = \frac{V_{peak} \times 255}{5.44 V}$$

LRA - Tensiones nominales y de sobremarcha

1. Convierta la tensión nominal de la hoja de datos del LRA en una tensión "media de los absolutos" utilizando la siguiente ecuación. Si no se indica la tensión de sobremarcha se debe utilizar la tensión nominal.

$$V_{avg_{abs}} = V_{rms} \times \sqrt{1 - (4 \times Sample Time + 300uS)f_{LRA}}$$

2. Utilizando la tensión "media de los absolutos" de la ecuación anterior, convertir al valor binario apropiado mediante la siguiente ecuación.

$$RatedVoltage (0x16) = \frac{V_{rated voltage} \times 255}{5.3V}$$

Insertar el valor binario en el RATED VOLTAGE registro (0x16)

3. La tensión de sobremarcha de un LRA se especifica como tensión de pico. Utilizar la siguiente ecuación para convertir la tensión de sobremarcha al valor binario apropiado.

$$ODClamp (0x17) = \frac{V_{peak} \times 255}{5.6 V}$$

Introducir el valor binario en el registro OVERDRIVE CLAMP VOLATGE registro (0x17).

Ajuste de los registros de control

Seleccionar los valores adecuados para los registros Realimentación, Control 1, Control 2 y Control 3 en función del actuador. Estos registros se utilizan para ajustar el rendimiento del actuador. Consulte el anexo 4 para obtener una descripción de cada registro. Las secciones siguientes describen los valores propuestos para utilizar las bibliotecas ERM y LRA.

Registros de control de ERM

En la Tabla 8 se muestran los registros que son necesarios configurar cuando se utilizan las bibliotecas ERM. Todos los demás registros pueden utilizar normalmente la configuración predeterminada.

Tabla 8 Registros de configuración para ERM.

REGISTRO		Selección de parámetros		
Nombre	Dirección	Nombre	Bits	Ajuste
Feedback Control	0x1A	nERM_LRA	[7]	0- ERM (Default)
Control 3	0x1D	ERM_OpenLoop	[5]	1- Open Loop
Selección de Librería	0x03	LibrarySel	[2:0]	1- Library A

Registros de Control LRA

Para los actuadores LRA, los registros de la Tabla 9 deben estar configurados. Todos los demás registros pueden utilizar normalmente la configuración predeterminada.

Tabla 9 Registros de configuración para LRA.

REGISTRO		Selección de parámetros		
Nombre	Dirección	Nombre	Bits	Ajuste
Feedback Control	0x1A	nERM_LRA	[7]	1- LRA
Control 3	0x1D	LRA DirveMode	[2]	0- Una por ciclo
		LRA_OpenLoop	[0]	0- Auto resonancia encendida

Selección de Librería	0x03	LibrarySel	[2:0]	6-LRA Library
-----------------------	------	------------	-------	---------------

Selección de la biblioteca de formas de onda

Hay seis bibliotecas ROM en el DRV26xx y cada una contiene 123 efectos. Las librerías 1-5 son para motores ERM y fueron diseñadas para soportar varios tipos y características de motores ERM. La biblioteca 6 es la biblioteca LRA y utiliza realimentación de bucle cerrado para auto ajustarse a cualquier actuador LRA. Los efectos en cada biblioteca se crearon para lograr la misma sensación, pero la salida aparecerá ligeramente diferente para tener en cuenta las diferencias en las características del motor como el tiempo de arranque, aceleración y tiempo de frenado.

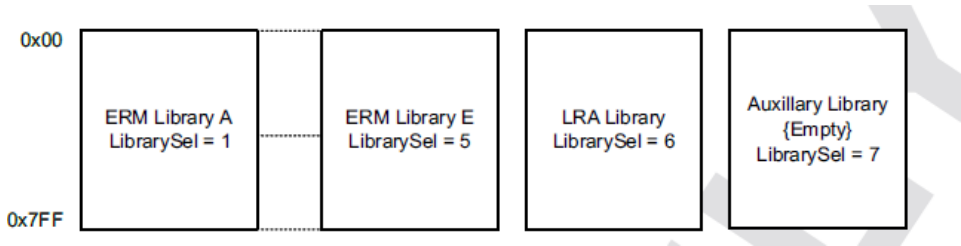


Figure 48 Librerías ROM DRV26XX (Texas Instruments, 2024).

Las bibliotecas ERM pueden elegirse en función de los tiempos de arranque y frenado del actuador. La biblioteca A es para motores que tienen tiempos de subida y parada más rápidos y la biblioteca E es para ERM que tienen tiempos de subida y parada más lentos.

Tabla 10 Librerías ROM par DRV26xx y propiedades del actuador.

Número	Librería	Propiedades del actuador	
		Rise time (ms)	Stop Time(ms)
1	Librería A	40-60	20-40
2	Librería B	40-60	5-15
3	Librería C	60-80	10-20
4	Librería D	100-140	15-25
5	Librería E	>140	>30

Propuesta de parámetros para la programación de la inicialización de los actuadores

A continuación se muestran los parámetros de inicialización propuestos para el actuador de tipo LRA.

Tabla 11 Parámetros de Inicialización para el actuador LRA

Registro			Selección de parámetros		
Nombre	Dirección	Valor (HEX)	Nombre	Bits	Ajuste
Rated Voltage	0x16	53	Ratedvoltage	[7:0]	2 Vrms
Overdrive Clamp Voltage	0x17	89	ODClamp	[7:0]	3 Vpeak
Feedback control	0x1A	B6	nERM_LRA	[7]	1 - LRA
			FBBrakeFactor	[6:4]	3 - 4x (default)
			Loop Response	[3:2]	2 - Fast (default)
			BEMFGain	[1:0]	2 – 1.8x/20x (default)
Auto-calibration Compensation Results	0x18	-	ACalComp	[7:0]	Escribir valor obtenido de la Auto Calibración
Auto-calibration Back-EMF Result	0x19	-	ACalBEMF	[7:0]	Escribir valor obtenido de la Auto Calibración
Control 1	0x1B	13	StartupBoost	[7]	0 - OFF
			AC_Couple	[5]	0 - DC Coupling / Modo Entrada digital
			DriveTime	[4:0]	19
Control 2	0x1C	AA	BiDir_Input	[7]	1 - Bi-direccional (default)
			Brake_Stabilizer	[6]	0 - OFF (default)
			AutoResGain	[5:4]	2 – Medio (default)
			Blanking Time	[3:2]	2 - Medio (default)

			IDissTime	[1:0]	2 - Medio(default)
Control 3	0x1D	80	NG_Thresh	[7:6]	2 - 4% (default)
			ERM_OpenLoop	[5]	0 – Lazo cerrado(default)
			SupplyCompDis	[4]	0 - ON (default)
			DataFormat_RTP	[3]	0 - Signado (default)
			LRA DriveMode	[2]	0 – Uno por ciclo
			nPWM_Analog	[1]	0 - PWM Input (default)
			LRA_OpenLoop	[0]	0 - Auto Resonancia On (default)
Library Selection	0x03	6	HiZ	[4]	0 - OFF (default)
			LibrarySel	[2:0]	6 – Librería LRA
Mode	0x01	0	Dev_Reset	[7]	0 - OFF (default)
			STANDBY	[6]	0 – Dispositivo Listo
			Mode	[2:0]	0 - Internal Trigger (default)

A continuación se muestran los parámetros de inicialización propuestos para el actuador de tipo ERM.

Tabla 12 Parámetros de Inicialización para el actuador ERM

Registro			Selección de parámetros		
Nombre	Dirección	Valor (HEX)	Nombre	Bits	Ajuste
Rated Voltage	0x16	90	RatedVoltage	[7:0]	3
Overdrive Clamp Voltage	0x17	A4	ODClamp	[7:0]	3.6
Feedback control	0x1A		nERM_LRA	[7]	0 - ERM (default)

		36	FBBrakeFactor	[6:4]	3 -4x (default)
			Loop Response	[3:2]	2 - Fast (default)
			BEMFGain	[1:0]	2 - 1.8x / 20x (default)
Auto-calibration Compensation Results	0x18	-	ACalComp	[7:0]	Escribir valor obtenido de la Auto Calibración
Auto-calibration Back-EMF Result	0x19	-	ACalBEMF	[7:0]	Escribir valor obtenido de la Auto Calibración
Control 1	0x1B	93	StartupBoost	[7]	1 - ON (default)
			AC_Couple	[5]	0 - DC Coupling /Modo Entrada Digital
			DriveTime	[4:0]	19
Control 2	0x1C	AA	BiDir_Input	[7]	1 - Bi-direccional (default)
			Brake_Stabilizer	[6]	0 - OFF (default)
			AutoResGain	[5:4]	2 - Medio (default)
			Blanking Time	[3:2]	2 – Medio (default)
			IDissTime	[1:0]	2 - Medio (default)
Control 3	0x1D	80	NG_Thresh	[7:6]	2- 4% (default)
			ERM_OpenLoop	[5]	0 – Lazo Cerrado (default)
			SupplyCompDis	[4]	0- ON (default)
			DataFormat_RTP	[3]	0 - Signado (default)
			LRA DriveMode	[2]	0 – Uno por ciclo
			nPWM_Analog	[1]	0 - PWM Input (default)

			LRA_OpenLoop	[0]	0 – Auto Resonancia On (default)
Library Selection	0x03	1	HiZ	[4]	0 - OFF (default)
			LibrarySel	[2:0]	1 - TS2200C Librería A con Overdrive
Mode	0x01	0	Dev_Reset	[7]	0 - OFF (default)
			STANDBY	[6]	0 – Dispositivo listo
			Mode	[2:0]	0 - Internal Trigger (default)

Autocalibración

La autocalibración es una función exclusiva que permite al driver ajustar la realimentación y el algoritmo de accionamiento a cualquier ERM o LRA concreto. Esto permite un mejor control, tiempo de respuesta y aceleración.

La auto-calibración no es lo mismo que la función de auto-resonancia LRA. En su lugar, la auto-calibración es un algoritmo que identifica y luego sintoniza el conductor basado en la "amplitud" de la back-EMF. Todos los ERM y LRA varían ligeramente en su construcción y, por lo tanto, tendrán un back-EMF único. Una vez realizada la autocalibración, el back-EMF de cada ERM o LRA se normaliza para que el motor de accionamiento lo vea igual.

Para realizar la autocalibración es necesario:

1. Conectar el actuador a los pines de salida y alimentar el dispositivo.
2. Salir del modo de STANDBY.

3. 3. Ajustar los siguientes registros a los valores apropiados:

- Rated Voltage (0x16)

- Overdrive Voltage(0x17)

- Feedback Control (0x1A) - Los bits [1:0] se dejarán en blanco y serán rellenos por el motor de autocalibración.

- Control 1 (0x1B), Control 2 (0x1C), y Control 3 (0x1D)

- Mode (0x01) - Configura el modo a Auto-Calibración

- Auto-calibration Memory Interface (0x1E) - el tiempo de auto-calibración puede incrementarse para mejorar la calibración, pero también puede dejarse por defecto para la calibración inicial

4. Activar el bit GO en el registro 0x0C para iniciar la Autocalibración.

5. Monitorear el bit GO hasta que cambie a cero, indicando que se ha completado la Auto-calibración o esperar hasta que el actuador deje de vibrar. No debería tardar más de 2 s.

6. Leer el bit Diag_Results en el Status Register (0x00). Diag_Resultado deberá ser "0" si la Autocalibración se ha realizado correctamente. Si es "1", asegurar que el actuador está conectado correctamente e intentarlo de nuevo.

7. Leer y guardar los valores de registro de ACalComp[7:0] (0x18), ACalBEMF[7:0] (0x19) y BEMFGain[1:0] del Registro FeedBack Control (0x1A). Estos son los valores devueltos por el motor de Autocalibración.

8. La autocalibración habrá finalizado.

A continuación se muestran los valores propuestos para un actuador LRA Y en seguida para un ERM.

Tabla 13 Parámetros de Auto calibración para el actuador LRA

Registro			Selección de parámetros		
Nombre	Dirección	Valor (HEX)	Nombre	Bits	Ajuste
Rated Voltage	0x16	53	RatedVoltage	[7:0]	2 Vrms
Overdrive Clamp Voltage	0x17	A0	ODClamp	[7:0]	2.82 Vpeak
Feedback control	0x1A	B6	nERM_LRA	[7]	1-LRA
			FBBrakeFactor	[6:4]	3 - 4x (default)
			Loop Response	[3:2]	1 -Medium
			BEMFGain	[1:0]	2 - 1.8x / 20x (default)
Control 1	0x1B	93	StartupBoost	[7]	1 - ON (default)
			AC_Couple	[5]	0 - DC Coupling / Modo entrada digital
			Drive Time	[4:0]	19
Control 2	0x1C	AA	BiDir_Input	[7]	1 - Bi-direccional (default)
			Brake_Stabilizer	[6]	0 - OFF (default)
			AutoResGain	[5:4]	2 - Medio (default)
			Blanking Time	[3:2]	2 - Medio (default)
			Diss Time	[1:0]	2 - Medio (default)
Control 3	0x1D	80	NG_Thresh	[7:6]	2-4% (default)
			ERM_OpenLoop	[5]	0 – Lazo cerrado (default)
			SupplyCompDis	[4]	0 - ON (default)
			DataFormat_RTP	[3]	0 – Signado (default)

			LRA DriveMode	[2]	0 – Uno por ciclo
			nPWM_Analog	[1]	0- PWM Input (default)
			LRA_OpenLoop	[0]	0 - Auto Resonancia On (default)
Mode	0x01	07	Dev_Reset	[7]	0 - OFF (default)
			STANDBY	[6]	0 – Dispositivo Listo
			Mode	[2:0]	7 - Auto Calibración
Auto-Calibration Memory Interface	0x1E	20	AutoCalTime	[5:4]	2 - 500 ms (default)
			OTP_Status	[2]	Solo Lectura
			OTP_Program	[0]	0 - OFF (default)
Go	0x0C	01	GO	[0]	1 – ON
Poll Go bit for “0”					
Status	0x00	Read (0xA8)	DeviceID	[7:5]	Read - 101
			Diag_Result	[3]	Read – 1
			Watchdog_status	[2]	Read – 0 / 1
			OverTemp	[1]	Read - 0
			OverCurrent	[0]	Read - 0
Auto-calibration Compensation Results	0x18	Read	ACalComp	[7:0]	Read value and store
Auto-calibration Back-EMF Result	0x19	Read	ACalBEMF	[7:0]	Read value and store
ControlFeedback	0x1A	Read	BEMFGain	[1:0]	Read bits [1:0] and store

Tabla 14 Parámetros de Auto calibración para el actuador ERM

Registro			Selección de parámetros		
<i>Nombre</i>	<i>Dirección</i>	<i>Valor (HEX)</i>	<i>Nombre</i>	<i>Bits</i>	<i>Ajuste</i>
Rated Voltage	0x16	90	RatedVoltage	[7:0]	3
Overdrive Clamp Voltage	0x17	A4	ODClamp	[7:0]	3.6
Feedback control	0x1A	36	nERM_LRA	[7]	0 - ERM (default)
			FBBrakeFactor	[6:4]	3 - 4x (default)
			Loop Response	[3:2]	1 -Medio
			BEMFGain	[1:0]	2 - 1.8x / 20x (default)
Control 1	0x1B	13	StartupBoost	[7]	0 - OFF
			AC_Couple	[5]	0 - DC Coupling / Modo Entrada Digital
			Drive Time	[4:0]	19
Control 2	0x1C	AA	BiDir_Input	[7]	1 - Bi-direccional (default)
			Brake_Stabilizer	[6]	0 - OFF (default)
			AutoResGain	[5:4]	2 - Medio (default)
			Blanking Time	[3:2]	2 - Medio (default)
			IDiss Time	[1:0]	2 - Medio (default)
Control 3	0x1D	0A	NG_Thresh	[7:6]	2 - 4% (default)
			ERM_OpenLoop	[5]	1- Lazo abierto
			SupplyCompDis	[4]	0 - ON (default)
			DataFormat_RTP	[3]	0 - Signado(default)
			LRA DriveMode	[2]	0 – Uno por ciclo
			nPWM_Analog	[1]	0- PWM Input (default)

			LRA_OpenLoop	[0]	0 - Auto Resonancia On (default)
Mode	0x01	07	Dev_Reset	[7]	0 - OFF (default)
			STANDBY	[6]	0 – Dispositivo Listo
			Mode	[2:0]	7 - Auto Calibración
Auto-Calibration Memory Interface	0x1E	20	AutoCalTime	[5:4]	2 - 500 ms (default)
			OTP_Status	[2]	Solo Lectura
			OTP_Program	[0]	0 - OFF (default)
Go	0x0C	1	GO	[0]	1 – ON
Poll Go bit for “0”					
Status	0x00	Read (0xA8)	DeviceID	[7:5]	Read - 101
			Diag_Result	[3]	Read – 1
			Watchdog_status	[2]	Read – 0 / 1
			OverTemp	[1]	Read - 0
			OverCurrent	[0]	Read - 0
Auto-calibration Compensation Results	0x18	Read	ACalComp	[7:0]	Read value and store
Auto-calibration Back-EMF Result	0x19	Read	ACalBEMF	[7:0]	Read value and store
ControlFeedback	0x1A	Read	BEMFGain	[1:0]	Read bits [1:0] and store

Programación de secuenciador de formas de onda

Para reproducir una forma de onda o una secuencia de formas de onda:

1. Salir del modo STANDBY poniendo el bit STANDBY del registro 0x01 a "0".
2. Inicializar el dispositivo con los ajustes adecuados (ERM/LRA, Lazo abierto / Lazo cerrado, etc.). Este procedimiento ya se ha explicado anteriormente en la sección de inicialización del dispositivo. Seguir los pasos de la sección Inicialización.
3. Seleccionar el modo de disparo (0 = Disparo interno, 1 = disparo por flanco externo, 2 = disparo por nivel externo) en el registro 0x01.
4. Escribir el primer número de índice de forma de onda en el primer registro de secuencia de forma de onda (0x04).
5. Escribir formas de onda adicionales en los registros de secuencia de forma de onda subsiguientes (0x05 -0x0B).
6. Si la secuencia contiene menos de ocho formas de onda, escribir el valor de terminación 0x00 en el registro de secuencia de forma de onda siguiente a la última forma de onda.
7. Para reproducir la forma de onda, ajuste el trigger alto según el modo de trigger seleccionado en el paso 3.
8. Una vez finalizada la secuencia, poner el aparato en STANDBY.

A continuación se muestra un ejemplo de los parámetros que deben ser programados, sin embargo cabe mencionar que los valores de los ID del secuenciador de forma de onda serán programados desde la interfaz.

Tabla 15 Secuencia de registros con el efecto 4, 7 y 5

Registro			Selección de parámetros		
Nombre	Dirección	Valor (HEX)	Nombre	Bits	Ajuste
Mode	0x01	0x00	Dev_Reset	[7]	0 - OFF (default)
			STANDBY	[6]	0 - Device Ready
			Mode	[2:0]	0 - Internal Trigger (default)
Waveform Sequencer	0x04	0x04	Wait + WavfrmSeq1	[7:0]	Escribir el Id de la forma de onda o en su defecto el tiempo de espera
	0x05	0x07	Wait + WavfrmSeq2	[7:0]	
	0x06	0x05	Wait + WavfrmSeq3	[7:0]	
	0x07	0x00	Wait + WavfrmSeq4	[7:0]	
	0x08	-	Wait + WavfrmSeq5	[7:0]	
	0x09	-	Wait + WavfrmSeq6	[7:0]	
	0x0A	-	Wait + WavfrmSeq7	[7:0]	
	0x0B	-	Wait + WavfrmSeq8	[7:0]	
GO	0x0C	0x01	GO	[0]	1 - ON

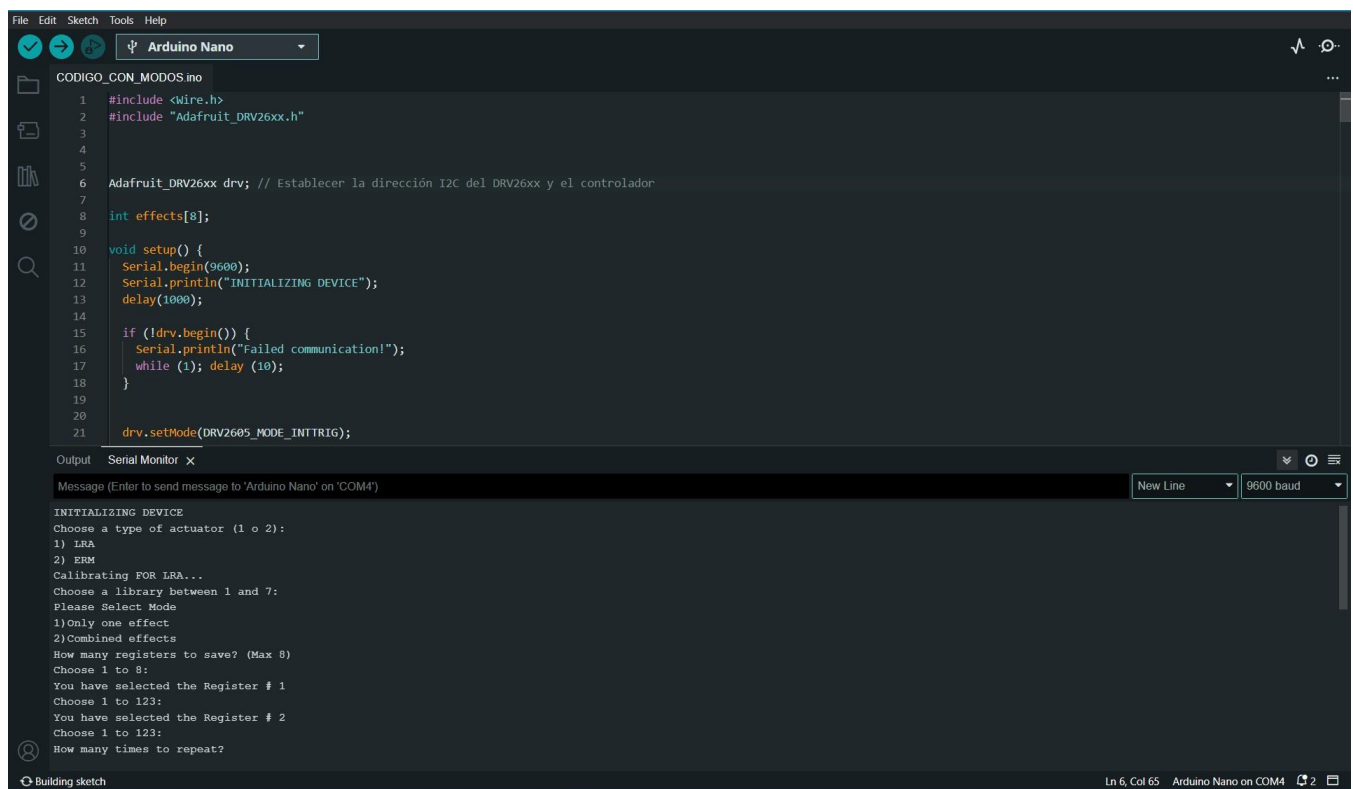
Resumen de la programación de driver

Para programar el driver son necesarias tres partes : Inicialización, Autocalibración y el Modo de reproducción de onda (para este caso el secuenciador de forma de onda), y con ellas el set up de los valores propuestos en los registros de memoria correspondientes.

Es por ello que para el entendimiento del proceso de cada uno de ellos, se ha desarrollado de forma resumida un diagrama de flujo para cada proceso de programación, que pueden ser consultados en la segunda sección del capítulo IV RESULTADOS.

Código de programación para el driver

A continuación se presenta el desarrollo del código para el control del microcontrolador: donde se puede observar la salida en el monitor serial de como se fueron ejecutando los comandos. El código completo se puede consultar en el anexo C.



The screenshot displays the Arduino IDE interface. The top menu bar includes File, Edit, Sketch, Tools, and Help. The toolbar shows icons for opening files, saving, and running the sketch. The file explorer on the left shows a folder named 'CODIGO_CON_MODOS.ino'. The main editor window displays the following code:

```
1 #include <Wire.h>
2 #include "Adafruit_DRV26xx.h"
3
4
5
6 Adafruit_DRV26xx drv; // Establecer la dirección I2C del DRV26xx y el controlador
7
8 int effects[8];
9
10 void setup() {
11   Serial.begin(9600);
12   Serial.println("INITIALIZING DEVICE");
13   delay(1000);
14
15   if (!drv.begin()) {
16     Serial.println("Failed communication!");
17     while (1); delay (10);
18   }
19
20   drv.setMode(DRV2605_MODE_INTTRIG);
21 }
```

The Serial Monitor window at the bottom shows the output of the program. It starts with "INITIALIZING DEVICE" and then prompts the user to choose a type of actuator (1 or 2). The user has selected 1 (LRA). The monitor then prompts the user to choose a library between 1 and 7. The user has selected 1 (only one effect). The monitor then prompts the user to choose how many registers to save (Max 8). The user has selected 1. The monitor then prompts the user to choose how many times to repeat. The user has selected 1. The monitor then displays the following text:

```
INITIALIZING DEVICE
Choose a type of actuator (1 o 2):
1) LRA
2) ERM
Calibrating FOR LRA...
Choose a library between 1 and 7:
Please Select Mode
1) only one effect
2) Combined effects
How many registers to save? (Max 8)
Choose 1 to 8:
You have selected the Register # 1
Choose 1 to 123:
You have selected the Register # 2
Choose 1 to 123:
How many times to repeat?
```

Figure 49 Salida del monitor serial. Ejecución del código para el microcontrolador. (Tlacumulco A., 2024).

Desarrollo de código para la HMI e integración de funcionalidades en Visual Studio

Una vez entendiendo las funciones con las que cuenta el código del microcontrolador se procede a hacer la construcción de la interfaz prototipo, para ello se comenzó primeramente con el diseño en QT designer, como se muestra a continuación.

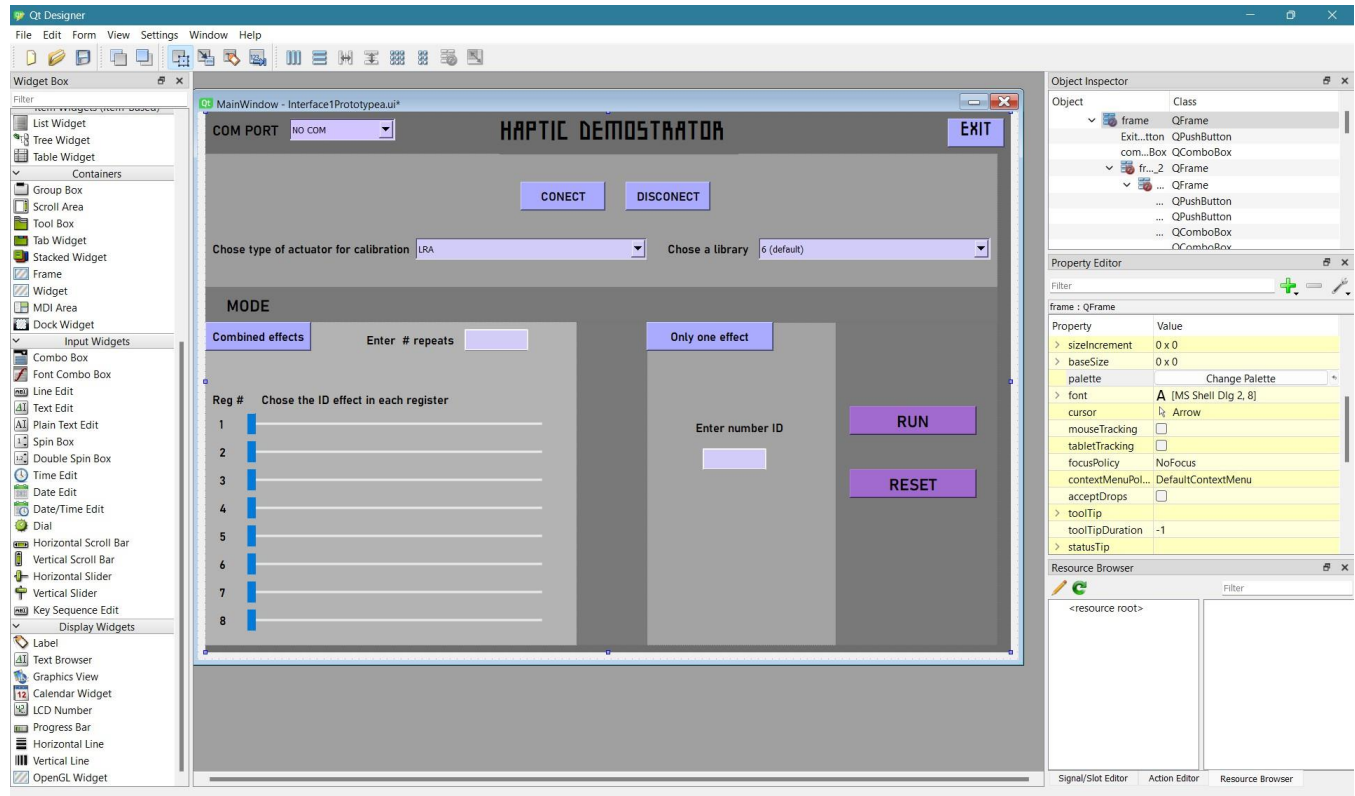


Figure 50 d=Diseño de Interfaz en qT designer (Tlacomulco A., 2024).

Se procede a guardar el diseño, en un archivo con terminación .UI , para este caso Interface1Prototype.UI, a continuación direccionarse a la carpeta donde se quiere guardar el código de Python, abrir la consola cmd desde esa dirección.

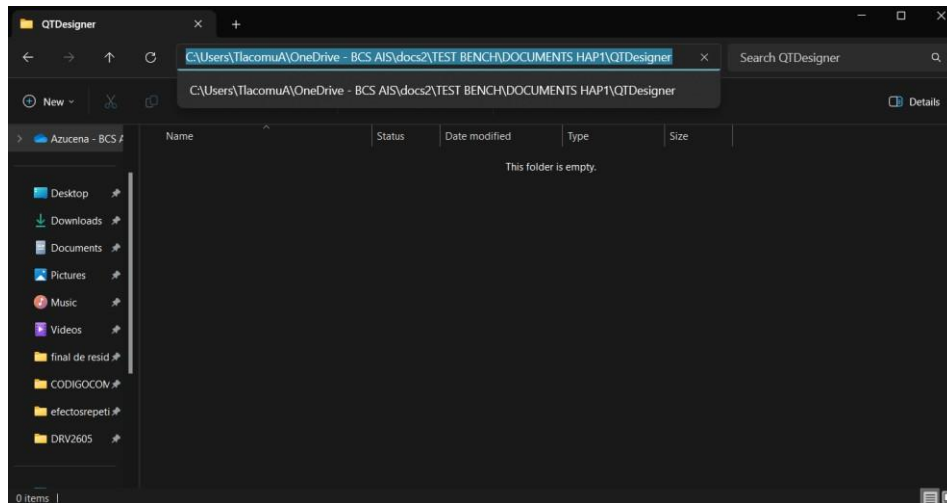


Figure 51 Posicionamiento en la ruta de la carpeta del proyecto (Tlacomulco A., 2024).

Dentro del cmd se ingresará el comando `pyuic5 -x` seguido del nombre como se nombrará al nuevo archivo de Python.

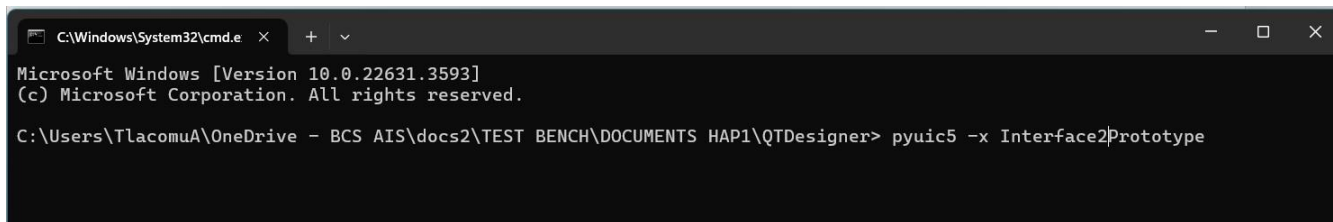


Figure 52 Conversión de archivo .UI a .PY (Tlacomulco A., 2024).

Entonces se obtiene un archivo de Python con el código del diseño de la interfaz.

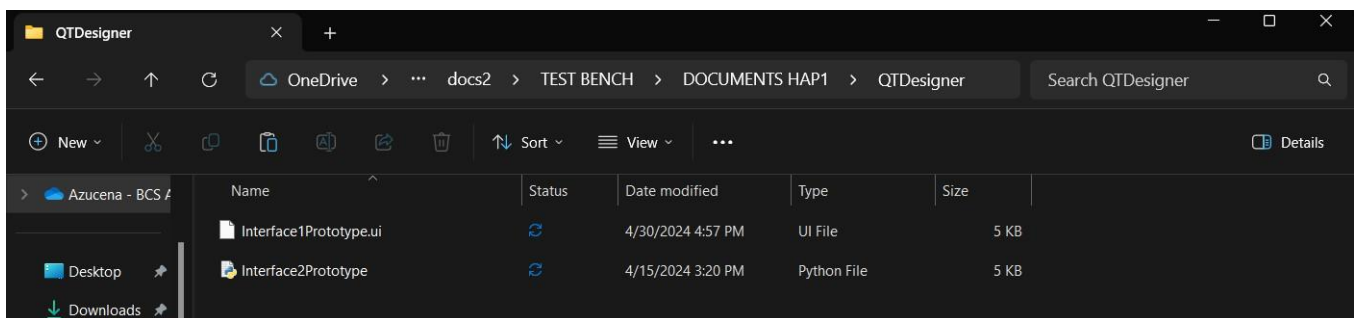


Figure 53 Obtención del código de Python.

Una vez teniendo esta base de código, se procede a realizar la unión de las funcionalidades desarrolladas en el código del microcontrolador junto a las desarrolladas en la interfaz en visual studio.

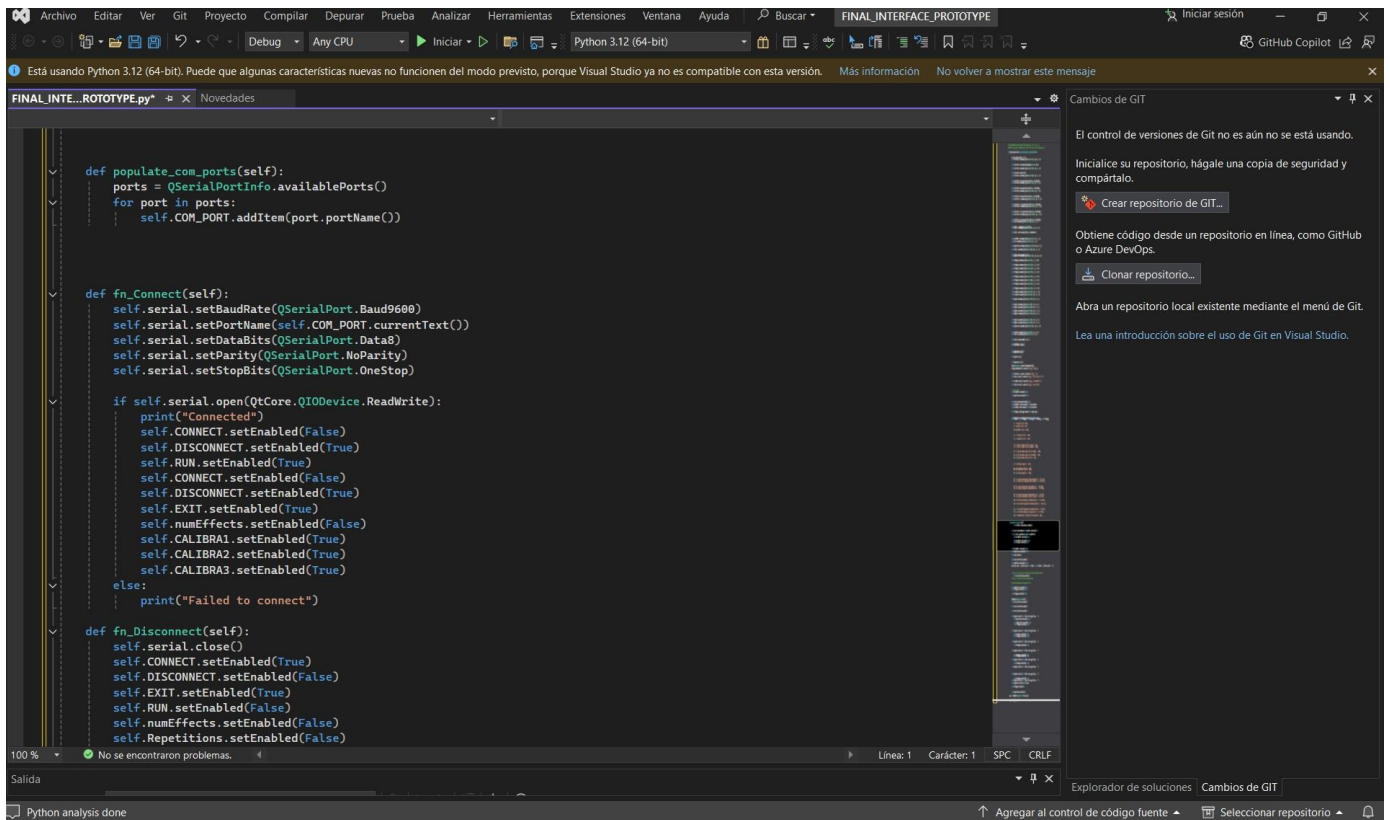


Figure 54 Integración de las funcionalidades en visual studio.

El código completo se puede consultar en el anexo D, y la visualización de la interfaz final se encontrará en el capítulo IV RESULTADOS Y CONCLUSIONES.

CAPITULO IV RESULTADOS Y CONCLUSIONES

RESULTADOS

Sección I Agrupación De Formas De Onda

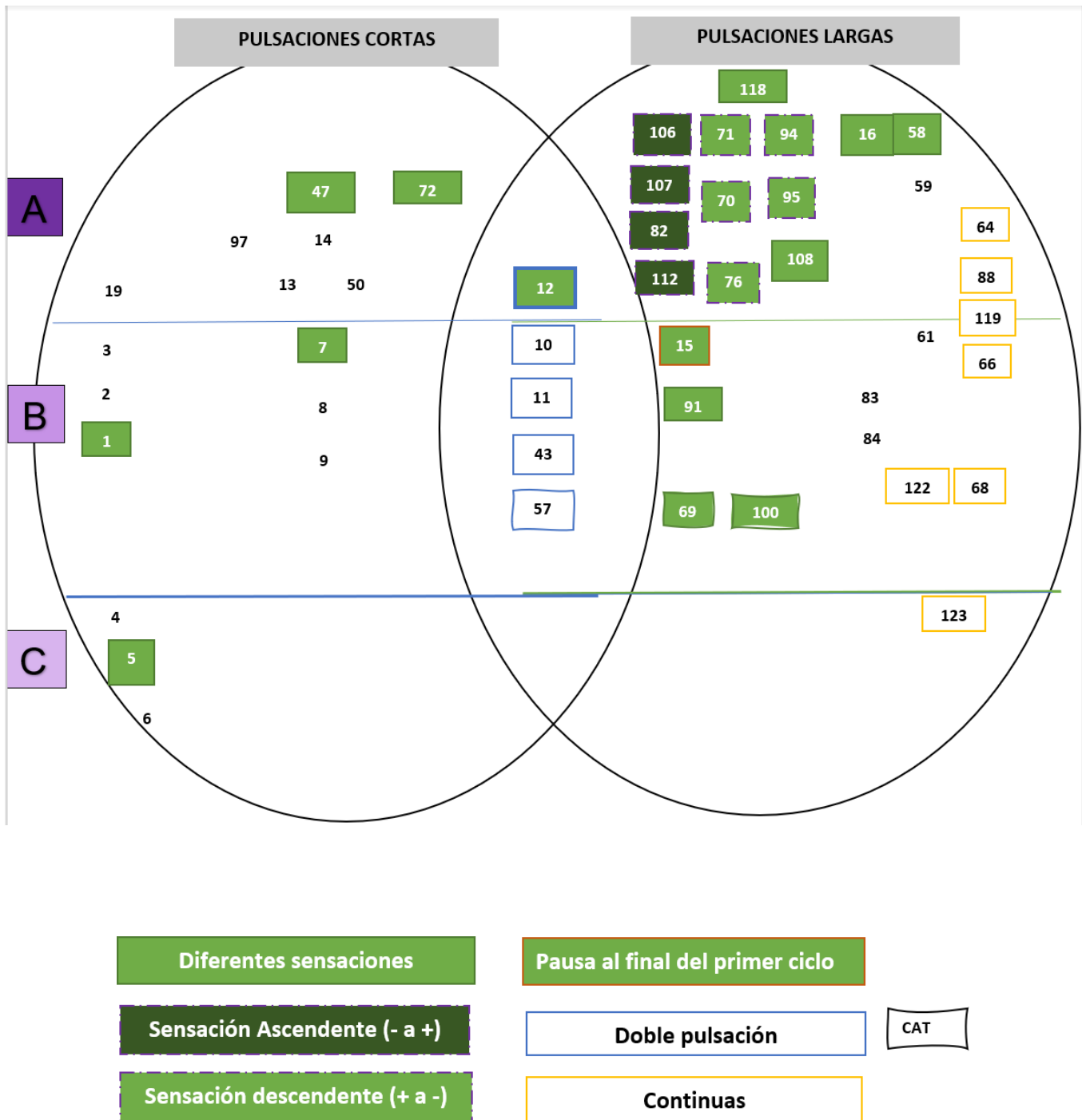


Figure 55 Diagrama de resultados de la agrupación de sensaciones.

El diagrama anterior muestra la agrupación obtenida de las diferentes sensaciones que se pueden sentir, diviendolos en 3 niveles de intensidad de fuerza de vibración, donde A representa el nivel más alto, B un nivel medio y C el nivel más bajo, a su vez se dividen en pulsaciones largas y cortas, donde dentro de ellas pueden existir continuas o con pausas en diferentes tiempos. La simbología que permitirá customizar cada efecto que se desee sentir, está incluida en dicho diagrama.

La funcionalidad del mismo, es la de familiarizar al usuario para la programación de las diferentes sensaciones que se deseen experimentar.

Sección II Diagramas De Flujo Para La Programación

Inicialización

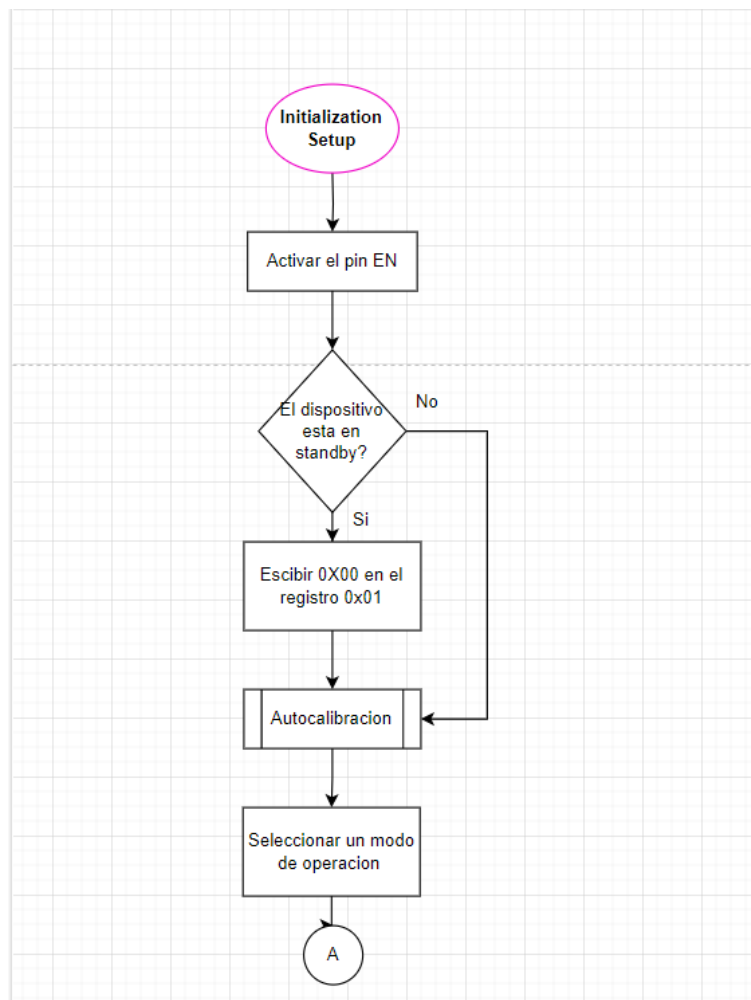


Figure 56 Diagrama de flujo para la programación de la inicialización partes A (Tlacomulco A.,2024)

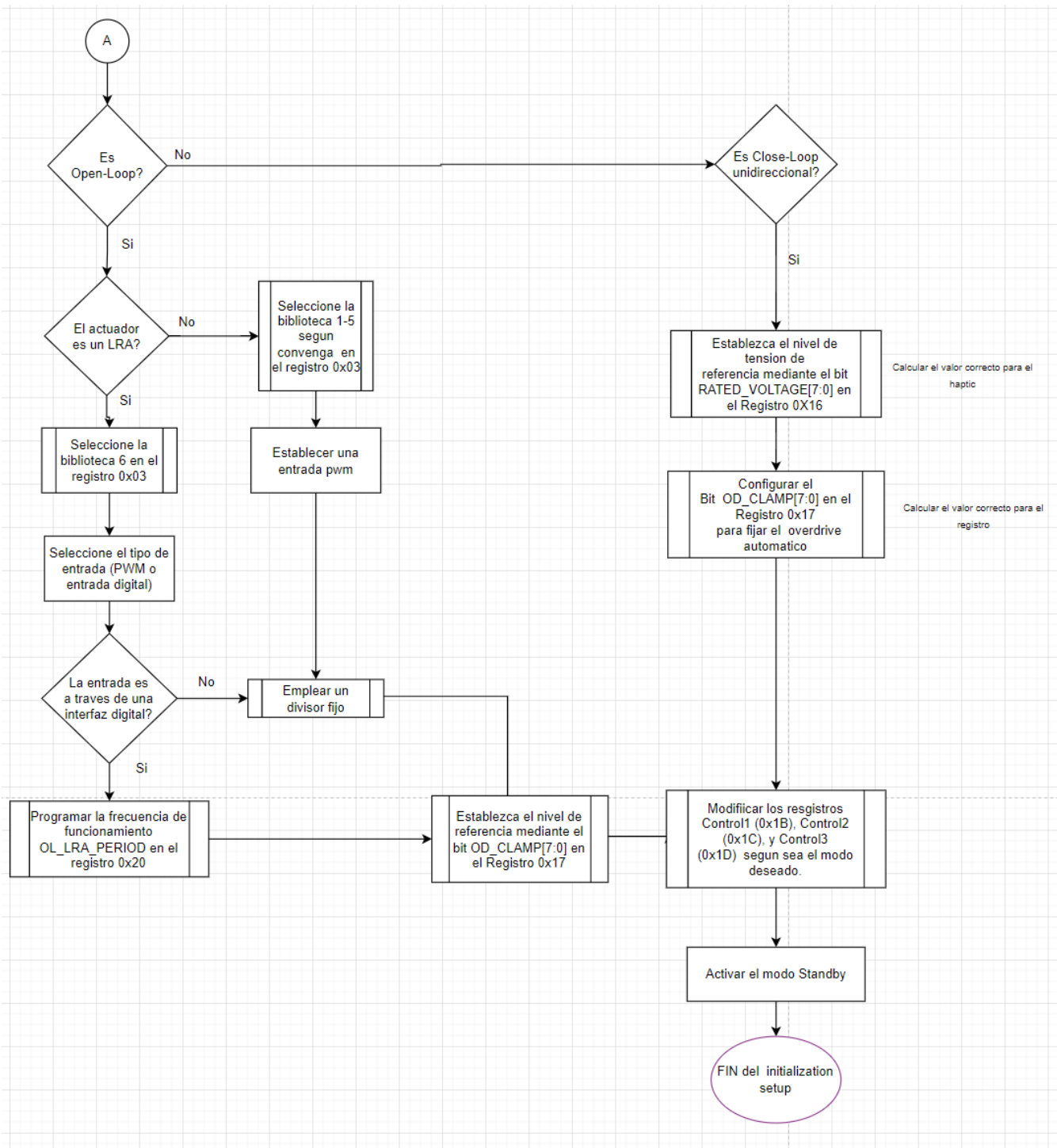


Figure 57 Diagrama de flujo para la programación de la inicialización parte B (Tlacomulco A.,2024).

Autocalibración

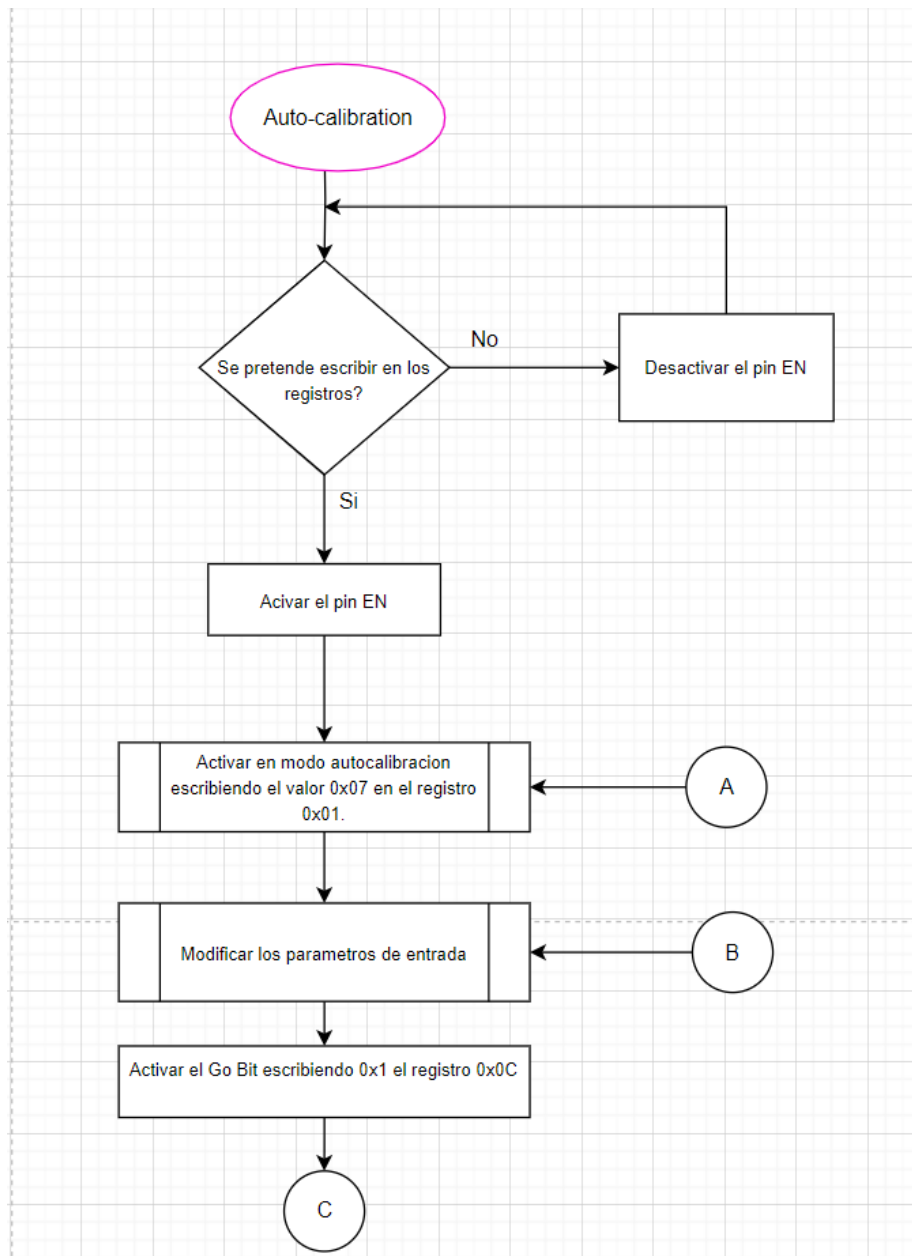


Figure 58 Diagrama de flujo para la programación de la Auto Calibración parte a (Tlacomulco A.,2024).

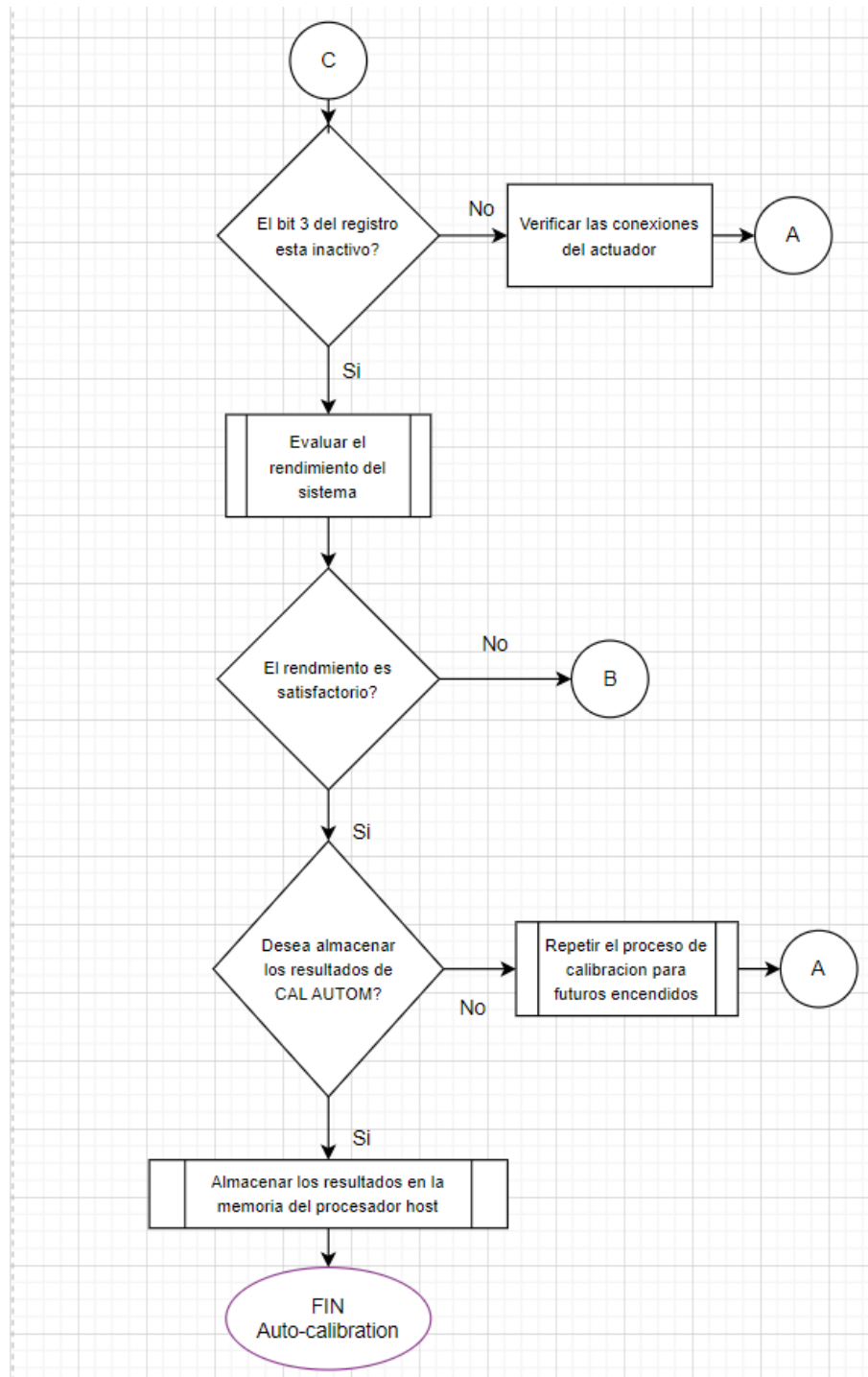


Figure 59 Diagrama de flujo para la programación de la Auto Calibración parte b (Tlacomulco A.,2024).

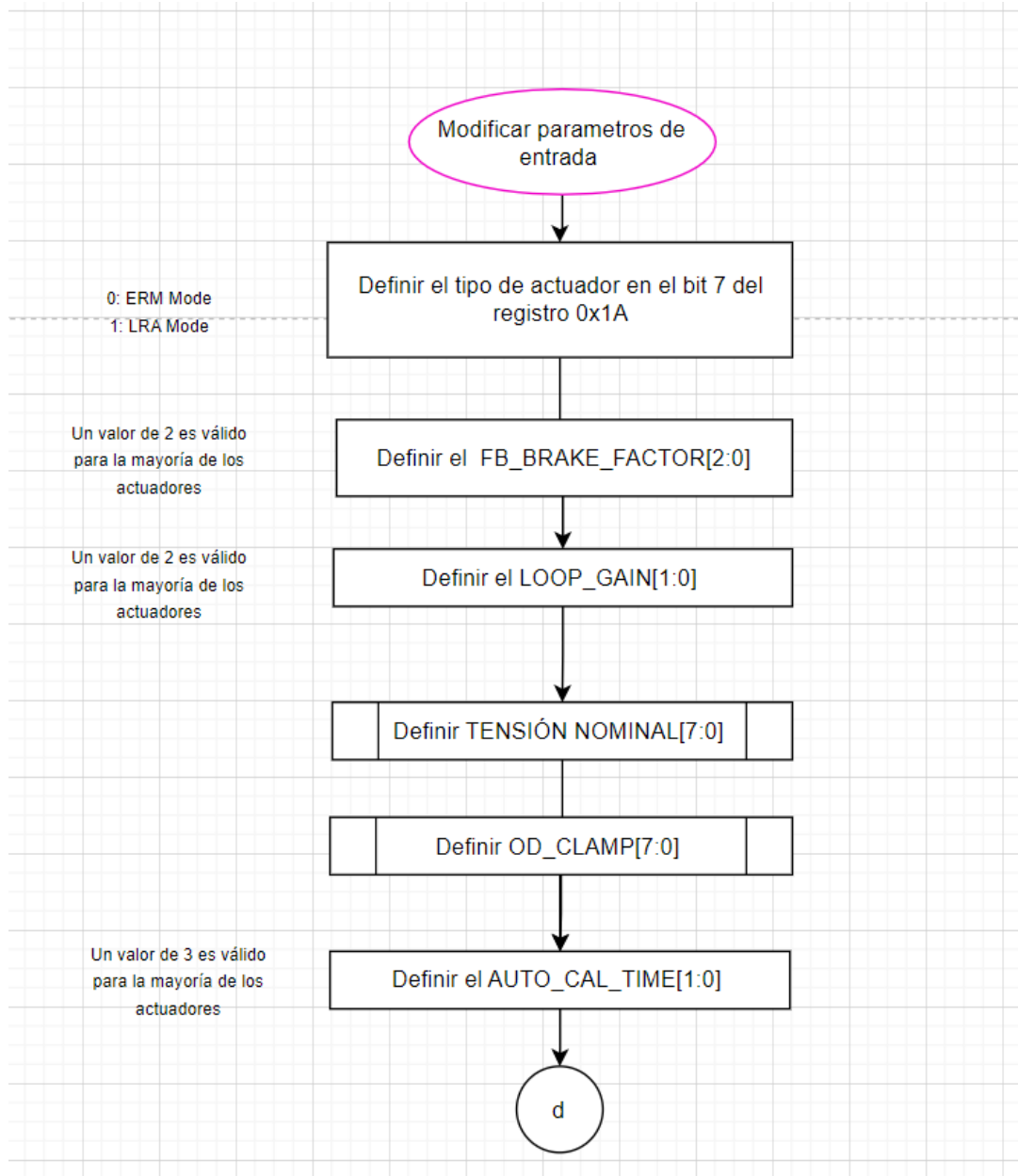


Figure 60 Diagrama de flujo para la programación del subproceso de la Auto Calibración parte a (Tlacomulco A.,2024).

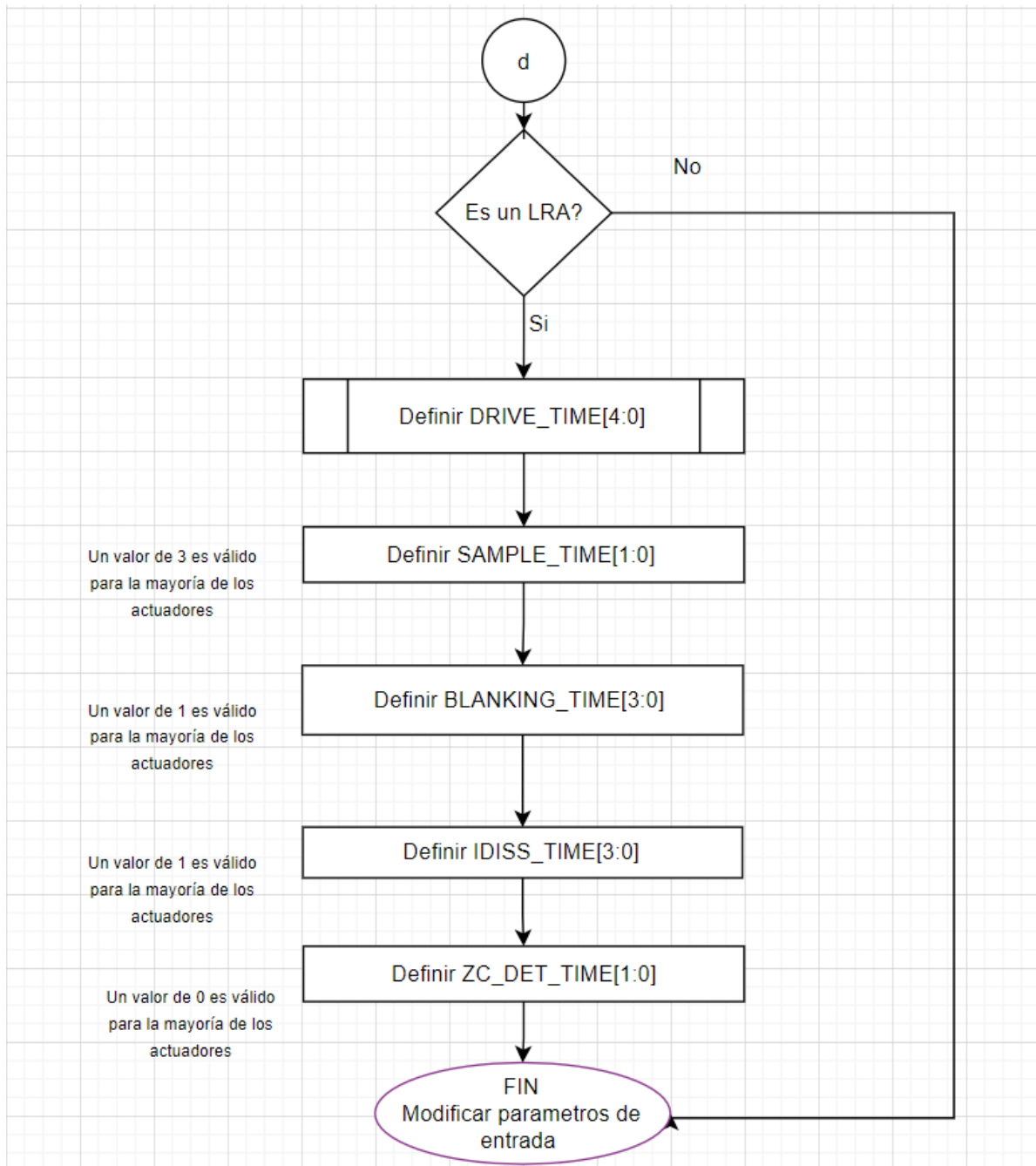


Figure 61 Diagrama de flujo para la programación del subproceso de la Auto Calibración parte b (Tlacomulco A.,2024).

Secuenciador de Formas de Onda

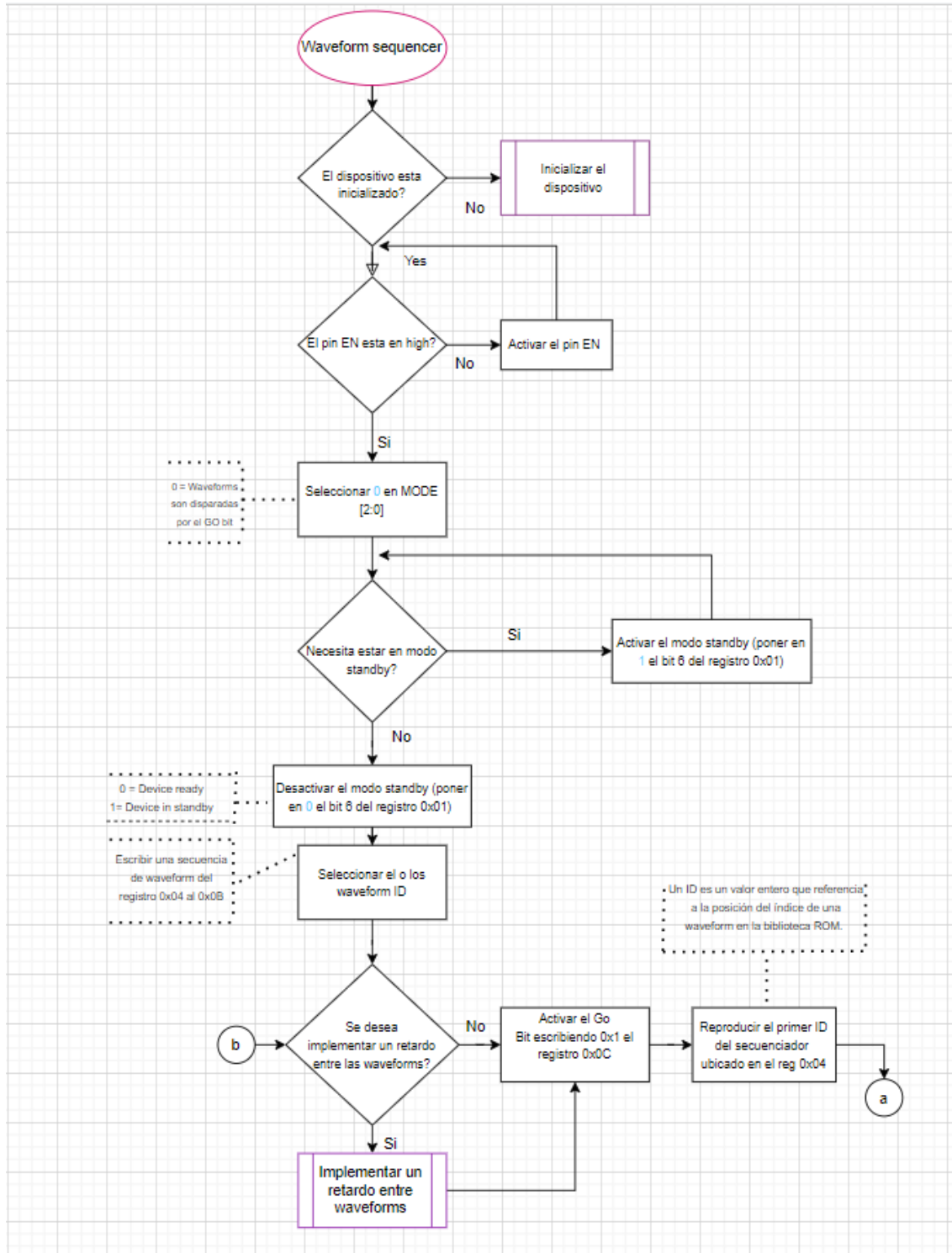


Figure 62 Diagrama de flujo para la programación de la reproducción de forma de onda parte a (Tlacomulco A.,2024).

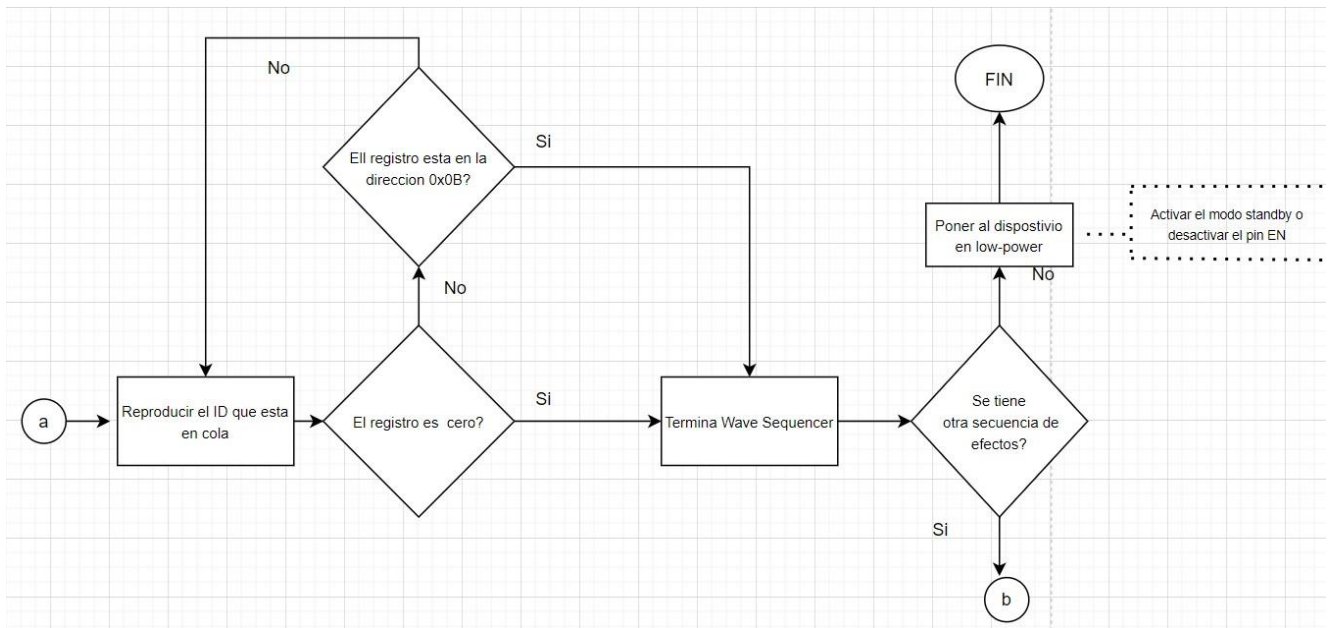


Figure 63 Diagrama de flujo para la programación de la reproducción de forma de onda parte b (Tlacomulco A.,2024).

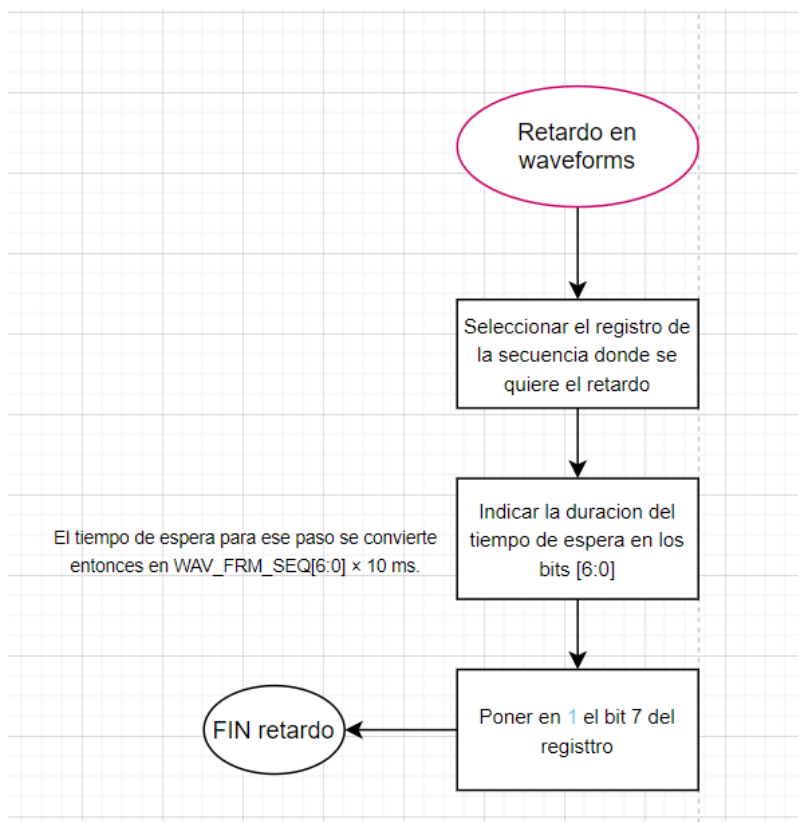


Figure 64 48 Diagrama de flujo para la programación del subproceso del reproductor de formas de onda (Tlacomulco A.,2024).

Sección III Resultado de la Interfaz prototipo

Al correr el programa en visual studio, es posible interactuar con la interfaz diseñada

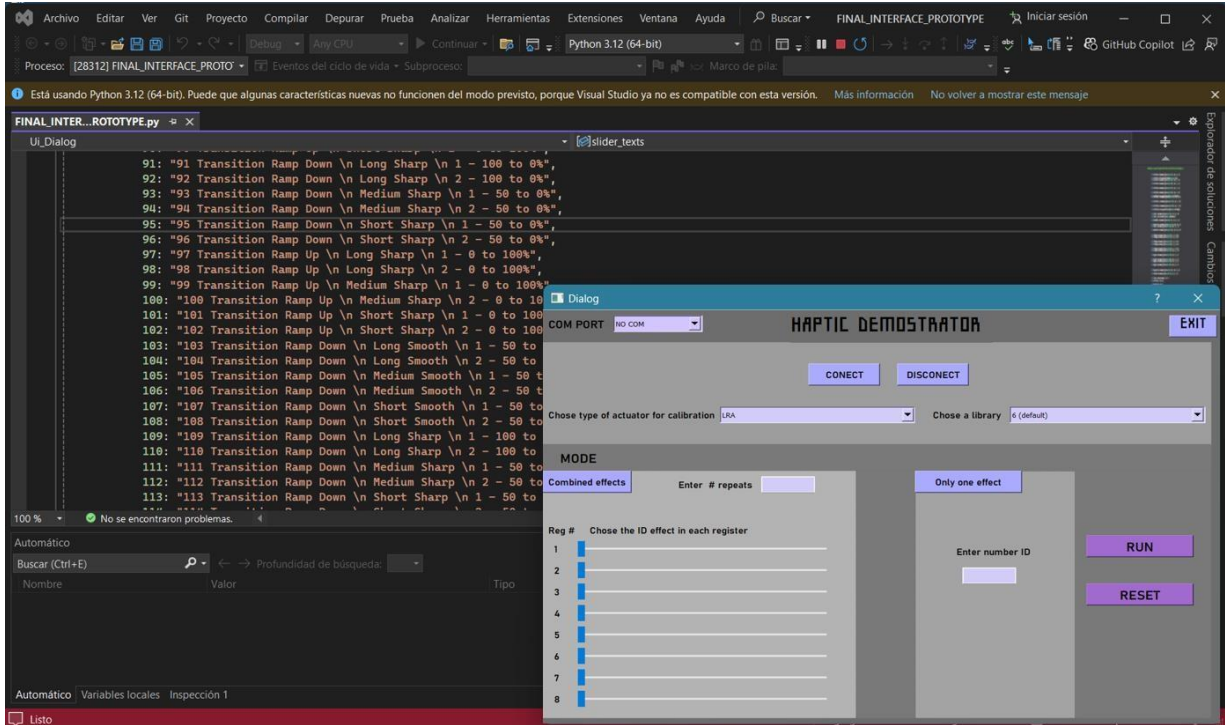


Figure 65 Interfaz prototipo (Tlacomulco A., 2024).

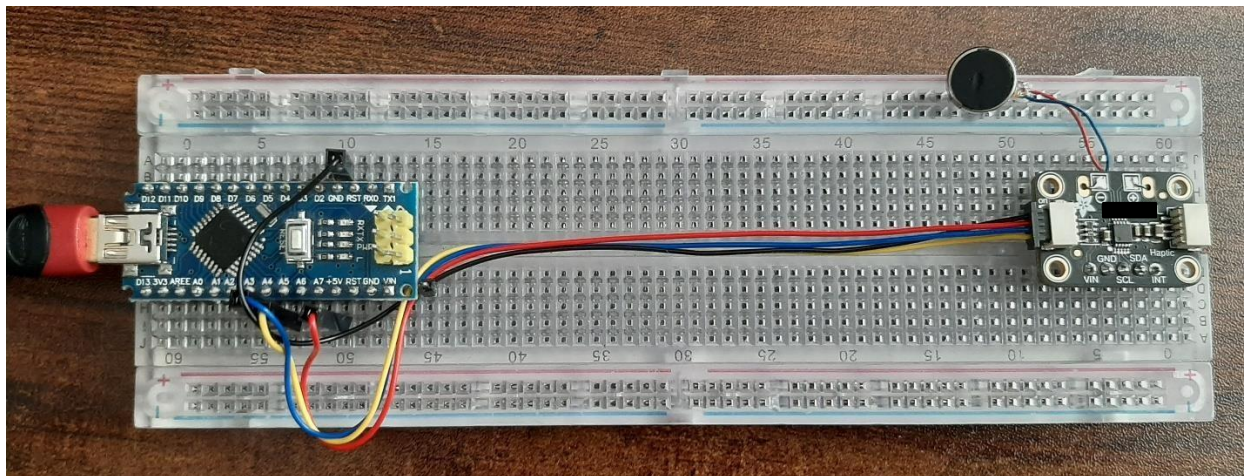


Figure 66 Montaje del sistema en prototipo (Tlacomulco A., 2024).

CONCLUSIONES

En síntesis, refiriéndose al proyecto se logró obtener la interfaz prototipo como un proyecto mínimo viable que permite reprogramar distintos efectos de vibración para diferentes actuadores, solucionando así el problema planteado al inicio de este documento.

A lo largo del desarrollo de este proyecto se presentaron distintas dificultades, una de las más importante fue lograr entender el funcionamiento del driver dentro de sus registros de memoria de una manera que permitiera customizarlo para usar el nivel de resonancia al actuador LRA al máximo, dentro de este contexto es importante mencionar el reto enfrentando de investigación sobre a integración y control de esta tecnología ya que no existe mucha información al respecto, además de la búsqueda y validación de las herramientas necesarias para la integración de todo el prototipo, ya que si bien no se menciona se intentó el mismo desarrollo con diferentes microcontroladores, softwares y lenguajes de programación.

Por otro lado con respecto al uso de la tecnología háptica , a medida que dependamos más de las interfaces electrónicas de usuario, necesitaremos información táctil que nos ayude a entender lo que hacen los sistemas. La tecnología háptica no sólo hace que las experiencias del usuario sean más satisfactorias, sino que también ayuda a mejorar el control, amplía la funcionalidad y aumenta la seguridad de los usuarios de los sistemas, al tiempo que ayuda a los fabricantes a diferenciar sus productos.

Finalmente hablando sobre el papel desenvuelto en este periodo de prácticas se logró alcanzar una formación superior a la tenida inicialmente, donde la constante exposición a la práctica empírica en la solución de problemas de ingeniería permitió aplicar los conocimientos con los que se contaba y a su vez solidificarlos. Además se estuvo expuesto a diferentes experiencias como puede ser el desarrollo de un producto desde las especificaciones de sistemas y software de las funcionalidades del mismo, el diseño electrónico de la PCBA con la integración de todos los submódulos, la manufactura en ensambles finales hasta la validación a través de pruebas como durabilidad, termal shock, etc.

CAPITULO V

COMPETENCIAS DESARROLLADAS

COMPETENCIAS OBTENIDAS

Programación y Desarrollo de Software: Manejo de lenguajes de programación como C#, C, C++ o Python.

Electrónica: Capacidad para desarrollar circuitos electrónicos que permitan integrar código de manera embebida, así como el conocimiento para la comprensión de la interacción entre componentes electrónicos que afectan directamente el buen funcionamiento del sistema, de la misma forma capacidad para la interpretación de diagramas electrónicos necesarios para la integración de las diferentes funcionalidades.

Conocimiento en protocolos de comunicación: Manejo de los diferentes protocolos de comunicación como CAN, LIN, I2C, UART, SPI, ETHERNET 10 BASE-T etc, para la integración de los diferentes microcontroladores, drivers, transceivers etc.

Conocimiento en el sector Automotriz: Capacidad para comprender los sistemas electrónicos y mecánicos de un automóvil, principales protocolos de comunicación, competencia y tendencia en el mercado automotriz así como principales áreas de oportunidad para la implementación de nueva tecnología en el diseño de HMI's.

Lean Six Sigma: Certificación en Six Sigma Green Belt, capacidad en la Gestión de Proyectos a través de la metodología Lean Six Sigma, con el propósito de optimizar procesos: Solucionando problemas en las líneas de producción a través del análisis estadístico. A demás de entrenamiento del uso de herramientas como minitab.

Certificación ISO 17025: Capacidad de aplicación de la Norma ISO 17025 “Requisitos generales para la competencia de los laboratorios de ensayo y calibración”; norma de calidad para laboratorios de ensayo y calibración. Se trata de un conjunto de requisitos que los laboratorios utilizan para demostrar que aplican un sistema de gestión de la calidad.

Diplomado en diseño y pruebas para la manufactura enfocado a electrónica: Competente para diseñar y evaluar la fabricación de PCB'S y procesos de manufactura o testeo en base a las normas IPC-610, IPC-600, IPC-221, IPC-2222, IPC-7351, IPC-2615, IPC-6012.

Manejo de diferentes softwares: Manejo de softwares como Mentor graphics, EasyEda, Cube IDE, Cube Mx, Hterm, Visual Studio, Qt designer, Note pad++, Eclipse, Canoe Data Analyser, TCANLIN Pro, KingsVist, Preform, Python, etc.

Trabajo en Equipo y Comunicación: Capacidad de trabajar en equipo y comunicarse efectivamente con otras áreas para el desarrollo de un producto, crucial para el éxito de los proyectos.

Adaptabilidad: Capacidad para adaptar rápidamente a los cambios, tiempos y prioridades., mostrando flexibilidad y resiliencia.

Planificación: Habilidad para establecer objetivos, desarrollar planes detallados y organizar recursos para alcanzar metas, además de realizar matrices de asignaciones y tiempos de entrega.

Referencias

1. Oropeza. “BCS Automotive Interface Solutions abre planta en Querétaro”. MexicoIndustry. Accedido el 26 de abril de 2024. [En línea]. Disponible: <https://mexicoindustry.com/noticia/bcs-automotive-interface-solutions-abre-planta-en-queretaro>
2. D. Knoedler. “Resultados de investigacion- Industria automotriz 2030 "La carrera hacia un futuro digital"”. IBM - United States. Accedido el 29 de junio de 2024. [En línea]. Disponible: <https://www.ibm.com/downloads/cas/RJPV6PGA>
3. P. Castellanos. “Conozca los fabricantes de autopartes en México | Metalmecánica”. Metalmecánica. Accedido el 5 de julio de 2024. [En línea]. Disponible: <https://www.metalmecanica.com/es/noticias/estas-son-las-autopartes-que-se-maquinan-en-mexico>
4. “háptico, háptica | Diccionario de la lengua española”. «Diccionario de la lengua española» - Edición del Tricentenario. Accedido el 5 de julio de 2024. [En línea]. Disponible: <https://dle.rae.es/háptico>
5. Colaboradores de los proyectos Wikimedia. “Háptica - Wikipedia, la enciclopedia libre”. Wikipedia, la enciclopedia libre. Accedido el 6 de julio de 2024. [En línea]. Disponible: <https://es.wikipedia.org/wiki/Háptica>
6. Contributors to Wikimedia projects. “Haptic suit - Wikipedia”. Wikipedia, the free encyclopedia. Accedido el 6 de julio de 2024. [En línea]. Disponible: https://en.wikipedia.org/wiki/Haptic_suit
7. D. Valero. “Pantalla háptica, la gran baza del iPhone 6 para sorprender”. ADSLZone. Accedido el 6 de julio de 2024. [En línea]. Disponible: <https://www.adslzone.net/2014/07/10/pantalla-haptica-la-gran-baza-del-iphone-6-para-sorprender/>
8. Electrostatics. “Pantalla táctil háptica | How it works, Application & Advantages”. Electricity - Magnetism. Accedido el 6 de julio de 2024. [En línea]. Disponible: <https://www.electricity-magnetism.org/es/pantalla-tactil-haptica/>
9. W., Krueger, Myron (1991). [Artificial reality II](#). Addison-Wesley. ISBN 0201522608. OCLC 21408597.
10. Rodríguez Molina. “Sistema de rehabilitación 3D usando dispositivo h'áptico”. Centro de Investigación y De Estudios Avanzados del Instituto Politécnico Nacional. Accedido el 5 de julio de 2024. [En línea]. Disponible: <https://www.ctrl.cinvestav.mx/~yuw/pdf/MaTesAR.pdf>

11. S. Keeping. “Los nuevos actuadores piezoeléctricos incluyen una realimentación háptica mejorada a los diseños de pantallas táctiles portátiles”,. Digi Key. Accedido el 6 de julio de 2024. [En línea]. Disponible: <https://www.digikey.com.mx/es/articles/new-piezoelectric-actuators-bring-enhanced-haptic-feedback>
12. P. Bree. “¿Qué es la tecnología háptica?” El Español. Accedido el 6 de julio de 2024. [En línea]. Disponible: https://www.lespanol.com/invertia/disruptores-innovadores/innovadores/tecnologicas/20191025/tecnologia-haptica/439207718_0.html
A. Ramirez. “Actuador Lineal De Resonancia - Transductor.net”. Transductor.net. Accedido el 6 de julio de 2024. [En línea]. Disponible: <https://transductor.net/actuador-lineal-de-resonancia/>
13. indeed motors. “¿Qué son los motores vibradores generales para teléfonos celulares en el mercado? - Noticias - Ineed Electronics (Hong Kong) Limited”. Proveedores y fabricantes de micromotores, motores de engranajes, motores de vibración, motores sin escobillas y fábrica - Ineed Electronics (Hong Kong) Limited. Accedido el 6 de julio de 2024. [En línea]. Disponible: <https://www.ineed-motor.com/news/what-are-general-cell-phone-vibrator-motor-60406945.html#:~:text=Un%20motor%20lineal%20es%20un,se%20genera%20un%20campo%20magnético.>
14. indeed motors. “China Personalizado 8mm Pequeños actuadores resonantes lineales Proveedores y fabricantes y fábrica - Hecho en China - Ineed Electronics”. Proveedores y fabricantes de micromotores, motores de engranajes, motores de vibración, motores sin escobillas y fábrica - Ineed Electronics (Hong Kong) Limited. Accedido el 6 de julio de 2024. [En línea]. Disponible: <https://www.ineed-motor.com/vibration-motor/coin-vibration-motor/linear-motor/8mm-small-linear-resonant-actuators.html>
15. Rodríguez, I. H. G. (2020, 14 de abril). ¿EN QUE CONSISTE LA FRECUENCIA NATURAL Y CUÁL ES SU IMPORTANCIA EN LAS ESTRUCTURAS? Ing. Héctor Gutiérrez Rodríguez. <https://ingett4x.blogspot.com/2020/04/en-que-consiste-la-frecuencia-natural-y.html>
16. G. White, Introduccion al analisis de las vibraciones. Woburn: Azima DLI, 2010. Accedido el 22 de julio de 2024. [En línea]. Disponible: <https://termogram.com/images/pdf/analisis-vibraciones/introduccion-al-analisis-de-vibraciones-azima-dli.pdf>
17. Blogger. “VIBRACIONES”. VIBRACIONES. Accedido el 23 de julio de 2024. [En línea]. Disponible: <https://riesgofisico-vibraciones.blogspot.com/2012/04/vibraciones.html>

18. P. French, Vibraciones y Ondas, 2a ed. Barcelona: Reverte, 2012. Accedido el 22 de julio de 2024. [En línea]. Disponible: <https://books.google.com.mx/books?hl=es&lr=&id=t9zeDwAAQBAJ&oi=fnd&pg=PR5&dq=teoria+de+vibraciones&ots=HbQczzZLHV&sig=u302zrMqZm1H71BZGRtJxYyXSTs#v=onepage&q&f=false>
19. Tech blog. “Linear Vibration Motor Introduction - BaoLoong M&E”. BMini Vibration Motors. Accedido el 31 de julio de 2024. [En línea]. Disponible: <http://www.vibrationmotors.com/techblogview.php?m=5&artid=3>
20. ineed motors. (2022, 27 de marzo). ERM y LRA--se aplican a los teléfonos móviles - Noticias - Ineed Electronics (Hong Kong) Limited. Proveedores y fabricantes de micromotores, motores de engranajes, motores de vibración, motores sin escobillas y fábrica - Ineed Electronics (Hong Kong) Limited. <https://www.ineed-motor.com/news/erm-and-lra-they-are-applied-to-cell-phones-60399758.html>
21. Siegel, E. (2011, 24 de julio). Haptics technology: picking up good vibrations - EE Times. EE Times. <https://www.eetimes.com/haptics-technology-picking-up-good-vibrations/>
22. Xukyo. “Visión general del microcontrolador Arduino NANO • AranaCorp”. Rochester Electronics. Accedido el 9 de agosto de 2024. [En línea]. Disponible: <https://www.aranacorp.com/es/vision-general-del-microcontrolador-arduino-nano/>
23. T. Instruments. “Información general sobre TI | TI.com”. Analog | Embedded processing | Semiconductor company | TI.com. Accedido el 9 de agosto de 2024. [En línea]. Disponible: <https://www.ti.com/es-mx/about-ti/company/ti-at-a-glance.html>
24. Electrosito.com. “Explicación De Los Conceptos Básicos De UART - Protocolo De Comunicación Y Sus Aplicaciones - Electrositio”. Electrositio. Accedido el 9 de agosto de 2024. [En línea]. Disponible: <https://electrositio.com/explicacion-de-los-conceptos-basicos-de-uart-protocolo-de-comunicacion-y-sus-aplicaciones/>

ANEXOS

Anexo A

Código de prueba para LRA

```
#include <Wire.h>
#include "Adafruit_DRV26XX.h"

Adafruit_DRV26XX drv;

void setup() {
  Serial.begin(9600);
  Serial.println("Adafruit DRV26XX Basic test");
  if (! drv.begin()) {
    Serial.println("Could not find DRV2605");
    while (1) delay(10);
  }

  drv.selectLibrary(1);

  // I2C trigger by sending 'go' command
  // default, internal trigger when sending GO command
  drv.setMode(DRV26XX_MODE_INTTRIG);
}

uint8_t effect = 1;

void loop() {
  // Serial.print("Effect #"); Serial.println(effect);

  if (effect == 1) {
    Serial.println("11.2 Waveform Library Effects List");
  }

  if (effect == 1) {
    Serial.println(F("1 - Strong Click - 100%"));
  }
  if (effect == 2) {
    Serial.println(F("2 - Strong Click - 60%"));
  }
  if (effect == 3) {
    Serial.println(F("3 - Strong Click - 30%"));
  }
  if (effect == 4) {
    Serial.println(F("4 - Sharp Click - 100%"));
  }
}
```

```

}
if (effect == 5) {
    Serial.println(F("5 - Sharp Click - 60%"));
}
if (effect == 6) {
    Serial.println(F("6 - Sharp Click - 30%"));
}
if (effect == 7) {
    Serial.println(F("7 - Soft Bump - 100%"));
}
if (effect == 8) {
    Serial.println(F("8 - Soft Bump - 60%"));
}
if (effect == 9) {
    Serial.println(F("9 - Soft Bump - 30%"));
}
if (effect == 10) {
    Serial.println(F("10 - Double Click - 100%"));
}
if (effect == 11) {
    Serial.println(F("11 - Double Click - 60%"));
}
if (effect == 12) {
    Serial.println(F("12 - Triple Click - 100%"));
}
if (effect == 13) {
    Serial.println(F("13 - Soft Fuzz - 60%"));
}
if (effect == 14) {
    Serial.println(F("14 - Strong Buzz - 100%"));
}
if (effect == 15) {
    Serial.println(F("15 - 750 ms Alert 100%"));
}
if (effect == 16) {
    Serial.println(F("16 - 1000 ms Alert 100%"));
}
if (effect == 17) {
    Serial.println(F("17 - Strong Click 1 - 100%"));
}
if (effect == 18) {
    Serial.println(F("18 - Strong Click 2 - 80%"));
}
if (effect == 19) {
    Serial.println(F("19 - Strong Click 3 - 60%"));
}

```

```

}
if (effect == 20) {
    Serial.println(F("20 - Strong Click 4 - 30%"));
}
if (effect == 21) {
    Serial.println(F("21 - Medium Click 1 - 100%"));
}
if (effect == 22) {
    Serial.println(F("22 - Medium Click 2 - 80%"));
}
if (effect == 23) {
    Serial.println(F("23 - Medium Click 3 - 60%"));
}
if (effect == 24) {
    Serial.println(F("24 - Sharp Tick 1 - 100%"));
}
if (effect == 25) {
    Serial.println(F("25 - Sharp Tick 2 - 80%"));
}
if (effect == 26) {
    Serial.println(F("26 - Sharp Tick 3 - 60%"));
}
if (effect == 27) {
    Serial.println(F("27 - Short Double Click Strong 1 - 100%"));
}
if (effect == 28) {
    Serial.println(F("28 - Short Double Click Strong 2 - 80%"));
}
if (effect == 29) {
    Serial.println(F("29 - Short Double Click Strong 3 - 60%"));
}
if (effect == 30) {
    Serial.println(F("30 - Short Double Click Strong 4 - 30%"));
}
if (effect == 31) {
    Serial.println(F("31 - Short Double Click Medium 1 - 100%"));
}
if (effect == 32) {
    Serial.println(F("32 - Short Double Click Medium 2 - 80%"));
}
if (effect == 33) {
    Serial.println(F("33 - Short Double Click Medium 3 - 60%"));
}
if (effect == 34) {
    Serial.println(F("34 - Short Double Sharp Tick 1 - 100%"));
}

```

```

}
if (effect == 35) {
    Serial.println(F("35 - Short Double Sharp Tick 2 - 80%"));
}
if (effect == 36) {
    Serial.println(F("36 - Short Double Sharp Tick 3 - 60%"));
}
if (effect == 37) {
    Serial.println(F("37 - Long Double Sharp Click Strong 1 - 100%"));
}
if (effect == 38) {
    Serial.println(F("38 - Long Double Sharp Click Strong 2 - 80%"));
}
if (effect == 39) {
    Serial.println(F("39 - Long Double Sharp Click Strong 3 - 60%"));
}
if (effect == 40) {
    Serial.println(F("40 - Long Double Sharp Click Strong 4 - 30%"));
}
if (effect == 41) {
    Serial.println(F("41 - Long Double Sharp Click Medium 1 - 100%"));
}
if (effect == 42) {
    Serial.println(F("42 - Long Double Sharp Click Medium 2 - 80%"));
}
if (effect == 43) {
    Serial.println(F("43 - Long Double Sharp Click Medium 3 - 60%"));
}
if (effect == 44) {
    Serial.println(F("44 - Long Double Sharp Tick 1 - 100%"));
}
if (effect == 45) {
    Serial.println(F("45 - Long Double Sharp Tick 2 - 80%"));
}
if (effect == 46) {
    Serial.println(F("46 - Long Double Sharp Tick 3 - 60%"));
}
if (effect == 47) {
    Serial.println(F("47 - Buzz 1 - 100%"));
}
if (effect == 48) {
    Serial.println(F("48 - Buzz 2 - 80%"));
}
if (effect == 49) {
    Serial.println(F("49 - Buzz 3 - 60%"));
}

```

```

}
if (effect == 50) {
    Serial.println(F("50 - Buzz 4 - 40%"));
}
if (effect == 51) {
    Serial.println(F("51 - Buzz 5 - 20%"));
}
if (effect == 52) {
    Serial.println(F("52 - Pulsing Strong 1 - 100%"));
}
if (effect == 53) {
    Serial.println(F("53 - Pulsing Strong 2 - 60%"));
}
if (effect == 54) {
    Serial.println(F("54 - Pulsing Medium 1 - 100%"));
}
if (effect == 55) {
    Serial.println(F("55 - Pulsing Medium 2 - 60%"));
}
if (effect == 56) {
    Serial.println(F("56 - Pulsing Sharp 1 - 100%"));
}
if (effect == 57) {
    Serial.println(F("57 - Pulsing Sharp 2 - 60%"));
}
if (effect == 58) {
    Serial.println(F("58 - Transition Click 1 - 100%"));
}
if (effect == 59) {
    Serial.println(F("59 - Transition Click 2 - 80%"));
}
if (effect == 60) {
    Serial.println(F("60 - Transition Click 3 - 60%"));
}
if (effect == 61) {
    Serial.println(F("61 - Transition Click 4 - 40%"));
}
if (effect == 62) {
    Serial.println(F("62 - Transition Click 5 - 20%"));
}
if (effect == 63) {
    Serial.println(F("63 - Transition Click 6 - 10%"));
}
if (effect == 64) {
    Serial.println(F("64 - Transition Hum 1 - 100%"));
}

```

```

}
if (effect == 65) {
    Serial.println(F("65 - Transition Hum 2 - 80%"));
}
if (effect == 66) {
    Serial.println(F("66 - Transition Hum 3 - 60%"));
}
if (effect == 67) {
    Serial.println(F("67 - Transition Hum 4 - 40%"));
}
if (effect == 68) {
    Serial.println(F("68 - Transition Hum 5 - 20%"));
}
if (effect == 69) {
    Serial.println(F("69 - Transition Hum 6 - 10%"));
}
if (effect == 70) {
    Serial.println(F("70 - Transition Ramp Down Long Smooth 1 - 100 to 0%"));
}
if (effect == 71) {
    Serial.println(F("71 - Transition Ramp Down Long Smooth 2 - 100 to 0%"));
}
if (effect == 72) {
    Serial.println(F("72 - Transition Ramp Down Medium Smooth 1 - 100 to 0%"));
}
if (effect == 73) {
    Serial.println(F("73 - Transition Ramp Down Medium Smooth 2 - 100 to 0%"));
}
if (effect == 74) {
    Serial.println(F("74 - Transition Ramp Down Short Smooth 1 - 100 to 0%"));
}
if (effect == 75) {
    Serial.println(F("75 - Transition Ramp Down Short Smooth 2 - 100 to 0%"));
}
if (effect == 76) {
    Serial.println(F("76 - Transition Ramp Down Long Sharp 1 - 100 to 0%"));
}
if (effect == 77) {
    Serial.println(F("77 - Transition Ramp Down Long Sharp 2 - 100 to 0%"));
}
if (effect == 78) {
    Serial.println(F("78 - Transition Ramp Down Medium Sharp 1 - 100 to 0%"));
}
if (effect == 79) {
    Serial.println(F("79 - Transition Ramp Down Medium Sharp 2 - 100 to 0%"));
}

```

```

}
if (effect == 80) {
    Serial.println(F("80 - Transition Ramp Down Short Sharp 1 - 100 to 0%"));
}
if (effect == 81) {
    Serial.println(F("81 - Transition Ramp Down Short Sharp 2 - 100 to 0%"));
}
if (effect == 82) {
    Serial.println(F("82 - Transition Ramp Up Long Smooth 1 - 0 to 100%"));
}
if (effect == 83) {
    Serial.println(F("83 - Transition Ramp Up Long Smooth 2 - 0 to 100%"));
}
if (effect == 84) {
    Serial.println(F("84 - Transition Ramp Up Medium Smooth 1 - 0 to 100%"));
}
if (effect == 85) {
    Serial.println(F("85 - Transition Ramp Up Medium Smooth 2 - 0 to 100%"));
}
if (effect == 86) {
    Serial.println(F("86 - Transition Ramp Up Short Smooth 1 - 0 to 100%"));
}
if (effect == 87) {
    Serial.println(F("87 - Transition Ramp Up Short Smooth 2 - 0 to 100%"));
}
if (effect == 88) {
    Serial.println(F("88 - Transition Ramp Up Long Sharp 1 - 0 to 100%"));
}
if (effect == 89) {
    Serial.println(F("89 - Transition Ramp Up Long Sharp 2 - 0 to 100%"));
}
if (effect == 90) {
    Serial.println(F("90 - Transition Ramp Up Medium Sharp 1 - 0 to 100%"));
}
if (effect == 91) {
    Serial.println(F("91 - Transition Ramp Up Medium Sharp 2 - 0 to 100%"));
}
if (effect == 92) {
    Serial.println(F("92 - Transition Ramp Up Short Sharp 1 - 0 to 100%"));
}
if (effect == 93) {
    Serial.println(F("93 - Transition Ramp Up Short Sharp 2 - 0 to 100%"));
}
if (effect == 94) {
    Serial.println(F("94 - Transition Ramp Down Long Smooth 1 - 50 to 0%"));
}

```

```

}
if (effect == 95) {
    Serial.println(F("95 - Transition Ramp Down Long Smooth 2 - 50 to 0%"));
}
if (effect == 96) {
    Serial.println(F("96 - Transition Ramp Down Medium Smooth 1 - 50 to 0%"));
}
if (effect == 97) {
    Serial.println(F("97 - Transition Ramp Down Medium Smooth 2 - 50 to 0%"));
}
if (effect == 98) {
    Serial.println(F("98 - Transition Ramp Down Short Smooth 1 - 50 to 0%"));
}
if (effect == 99) {
    Serial.println(F("99 - Transition Ramp Down Short Smooth 2 - 50 to 0%"));
}
if (effect == 100) {
    Serial.println(F("100 - Transition Ramp Down Long Sharp 1 - 50 to 0%"));
}
if (effect == 101) {
    Serial.println(F("101 - Transition Ramp Down Long Sharp 2 - 50 to 0%"));
}
if (effect == 102) {
    Serial.println(F("102 - Transition Ramp Down Medium Sharp 1 - 50 to 0%"));
}
if (effect == 103) {
    Serial.println(F("103 - Transition Ramp Down Medium Sharp 2 - 50 to 0%"));
}
if (effect == 104) {
    Serial.println(F("104 - Transition Ramp Down Short Sharp 1 - 50 to 0%"));
}
if (effect == 105) {
    Serial.println(F("105 - Transition Ramp Down Short Sharp 2 - 50 to 0%"));
}
if (effect == 106) {
    Serial.println(F("106 - Transition Ramp Up Long Smooth 1 - 0 to 50%"));
}
if (effect == 107) {
    Serial.println(F("107 - Transition Ramp Up Long Smooth 2 - 0 to 50%"));
}
if (effect == 108) {
    Serial.println(F("108 - Transition Ramp Up Medium Smooth 1 - 0 to 50%"));
}
if (effect == 109) {
    Serial.println(F("109 - Transition Ramp Up Medium Smooth 2 - 0 to 50%"));
}

```

```

}
if (effect == 110) {
    Serial.println(F("110 - Transition Ramp Up Short Smooth 1 - 0 to 50%"));
}
if (effect == 111) {
    Serial.println(F("111 - Transition Ramp Up Short Smooth 2 - 0 to 50%"));
}
if (effect == 112) {
    Serial.println(F("112 - Transition Ramp Up Long Sharp 1 - 0 to 50%"));
}
if (effect == 113) {
    Serial.println(F("113 - Transition Ramp Up Long Sharp 2 - 0 to 50%"));
}
if (effect == 114) {
    Serial.println(F("114 - Transition Ramp Up Medium Sharp 1 - 0 to 50%"));
}
if (effect == 115) {
    Serial.println(F("115 - Transition Ramp Up Medium Sharp 2 - 0 to 50%"));
}
if (effect == 116) {
    Serial.println(F("116 - Transition Ramp Up Short Sharp 1 - 0 to 50%"));
}
if (effect == 117) {
    Serial.println(F("117 - Transition Ramp Up Short Sharp 2 - 0 to 50%"));
}
if (effect == 118) {
    Serial.println(F("118 - Long buzz for programmatic stopping - 100%"));
}
if (effect == 119) {
    Serial.println(F("119 - Smooth Hum 1 (No kick or brake pulse) - 50%"));
}
if (effect == 120) {
    Serial.println(F("120 - Smooth Hum 2 (No kick or brake pulse) - 40%"));
}
if (effect == 121) {
    Serial.println(F("121 - Smooth Hum 3 (No kick or brake pulse) - 30%"));
}
if (effect == 122) {
    Serial.println(F("122 - Smooth Hum 4 (No kick or brake pulse) - 20%"));
}
if (effect == 123) {
    Serial.println(F("123 - Smooth Hum 5 (No kick or brake pulse) - 10%"));
}

// set the effect to play

```

```

drv.setWaveform(0, effect); // play effect
drv.setWaveform(1, 0);      // end waveform

// play the effect!
drv.go();

// wait a bit
delay(500);

effect++;
if (effect > 117) effect = 1;
}

```

Anexo B

Código para casos de prueba de los efectos hápticos

```

#include <Wire.h>
#include "Adafruit_DRV26xx.h"

Adafruit_DRV26XX drv;

void setup() {
  Serial.begin(9600);
  Serial.println("Basic test for each effect: Please enter a number between 1 and 123 or 's' to stop.");
  if (!drv.begin()) {
    Serial.println("Failed communication!");
    while (1) delay(10);
  }

  drv.selectLibrary(1);
  drv.setMode(DRV26XX_MODE_INTTRIG);
}

void loop() {
  static int effect = 0; // Variable estática para mantener el efecto actual
  static unsigned long lastTime = 0; // Variable para el temporizador

  // Espera hasta que se reciba un número a través del Serial
  if (Serial.available() > 0) {
    String input = Serial.readStringUntil('\n'); // Lee la entrada hasta el salto de línea

```

```

if (input.equals("s")) { // Verifica si la entrada es 's'
    effect = 0; // Detiene el efecto
    Serial.println("Effect stopped.");
} else {
    int newEffect = input.toInt(); // Convierte la entrada a un número

    if (newEffect >= 1 && newEffect <= 123) { // Verifica si el ID del efecto está en
el rango válido
        effect = newEffect; // Actualiza el efecto si es válido
        Serial.print("Playing Effect #");
        Serial.println(effect);
    } else {
        Serial.println("Invalid effect ID. Please enter a number between 1 and 123.");
        effect = 0;
    }
}

// Limpia el buffer de entrada
while (Serial.available() > 0) {
    Serial.read();
}

// Reproduce el efecto seleccionado indefinidamente
if (effect > 0) { // se asegura de que hay un efecto válido
    if (millis() - lastTime > 500) { // Reproduce cada 500 ms
        drv.setWaveform(0, effect);
        drv.setWaveform(1, 0); // Fin de la forma de onda
        drv.go();
        lastTime = millis(); // Actualiza el tiempo
    }
}
}
}

```

Anexo C

Código para el microcontrolador

```
#include <Wire.h>
#include "Adafruit_DRV26xx.h"

Adafruit_DRV26xx drv; // Establecer la dirección I2C del DRV26xx y el controlador

int effects[8];

void setup() {
  Serial.begin(9600);
  Serial.println("INITIALIZING DEVICE");
  delay(1000);

  if (!drv.begin()) {
    Serial.println("Failed communication!");
    while (1); delay (10);
  }

  drv.setMode(DRV2605_MODE_INTTRIG);

  Serial.println("Choose a type of actuator (1 o 2):");
  Serial.println("1) LRA");
  Serial.println("2) ERM");

  int actuatorType = -1;
  while (actuatorType < 1 || actuatorType > 2) {
    while (!Serial.available());
    actuatorType = Serial.parseInt();

    switch (actuatorType) {
      case 1:
        Actuator1();
        break;
      case 2:
        Actuator2();
        break;

      default:
        Serial.println("Ups! actuator not identified. Try again");
    }
  }
}
```

```

        actuatorType = 0; // Reset the value
        break;
    }
}

Serial.println("Choose a library between 1 and 7:");

int libraryType = 0;
while (libraryType < 1 || libraryType > 7) {
    while (!Serial.available());
    libraryType = Serial.parseInt();

    // Verificar si la entrada es válida
    if (libraryType < 1 || libraryType > 7) {
        Serial.println("Sorry, library not defined. Try again");
        libraryType = -1; // Reiniciar el valor
        continue; // Volver al inicio del bucle
    }

    switch (libraryType) {
        case 1:
            drv.selectLibrary(1);
            break;
        case 2:
            drv.selectLibrary(2);
            break;
        case 3:
            drv.selectLibrary(3);
            break;
        case 4:
            drv.selectLibrary(4);
            break;
        case 5:
            drv.selectLibrary(5);
            break;
        case 6:
            drv.selectLibrary(6);
            break;
        case 7:
            drv.selectLibrary(7);
            break;
    }
}
}

```

```

/*
*/

void loop() {
    drv.writeRegister8(DRV26xx_REG_MODE, 0x00);

    Serial.println("Please Select Mode");
    Serial.println("1)Only one effect");
    Serial.println("2)Combined effects");

    int mode = 0;
    while (mode < 1 || mode > 2) {
        while (!Serial.available());
        mode = Serial.parseInt();

        switch (mode) {
            case 1:
                Only1();
                break;
            case 2:
                Combined2();
                break;
        }
    }
}

void Actuator1() {
    Serial.println("Calibrating FOR LRA...");
    drv.writeRegister8(DRV26xx_REG_MODE, 0x07); // Setear en modo y poner autocalibración

    drv.writeRegister8(0x16, 0x53); // RATED VOLTAGE
    drv.writeRegister8(0x17, 0xA0); // OVERDRIVE CLAMP VOLTAGE
    drv.writeRegister8(0x1A, 0xB6); // FEEDBACK CONTROL
    drv.writeRegister8(0x1B, 0x93); // CONTROL 1
    drv.writeRegister8(0x1C, 0xAA); // CONTROL 2
    drv.writeRegister8(0x1D, 0x80); // CONTROL 3
    drv.writeRegister8(0x1E, 0xB0); // CONTROL 4
    drv.writeRegister8(0x1F, 0xAF); // Control 5
    drv.writeRegister8(0x01, 0x07); // MODO
    drv.writeRegister8(0x1E, 0x20); //AUTO- CALIBRATION MEMORY INTERFACE
    drv.go();
    delay(1000);
}

```

```

}

void Actuator2() {
  Serial.println("Calibrating FOR ERM");
  drv.writeRegister8(DRV26xx_REG_MODE, 0x07); // Setear en modo y poner autocalibración

  drv.writeRegister8(0x16, 0x90); // RATED VOLTAGE
  drv.writeRegister8(0x17, 0xA4); // OVERDRIVE CLAMP VOLTAGE
  drv.writeRegister8(0x1A, 0x36); // FEEDBACK CONTROL
  drv.writeRegister8(0x1B, 0x13); // CONTROL 1
  drv.writeRegister8(0x1C, 0xAA); // CONTROL 2
  drv.writeRegister8(0x1D, 0xA0); // CONTROL 3
  drv.writeRegister8(0x1F, 0xAF); // Control 5
  drv.writeRegister8(0x1E, 0xB0); // CONTROL 4
  drv.writeRegister8(0x01, 0x07); // MOD0
  drv.writeRegister8(0x1E, 0x20); //AUTO- CALIBRATION MEMORY INTERFACE

  drv.go();
  delay(1000);
}

void Combined2(){
  Serial.println("How many registers to save? (Max 8)");

  int numRegs = -1;

  while (numRegs < 1 || numRegs > 8) {
    while (!Serial.available());
    numRegs = Serial.parseInt();
    reset(numRegs);
    if (numRegs < 1 || numRegs > 8) {
      Serial.println("Choose 1 to 8:");
    }
  }
}

for (int i = 0; i < numRegs; i++) {
  Serial.print("You have selected the Register # "); Serial.println(i + 1);
  int numEffect = 0;
  while (numEffect < 1 || numEffect > 123) {
    while (!Serial.available());
    numEffect = Serial.parseInt();
    if (numEffect < 1 || numEffect > 123) {
      Serial.println("Choose 1 to 123:");
    }
  }
}
}

```

```

    effects[i] = numEffect;
    drv.writeRegister8(DRV26xx_REG_WAVESEQ1 + i, effects[i]);
}

RepeatCount();
delay(2000);
}

void RepeatCount() {
    Serial.println("How many times to repeat?");
    int repeatCount = 0;
    while (repeatCount < 1) {
        while (!Serial.available());
        repeatCount = Serial.parseInt();
    }

    for (int j = 0; j < repeatCount; j++) {
        drv.go();
        delay(1000);
    }
}

void Only1() {

Serial.println("Basic test for each effect: Please enter a number between 1 and 123 or
's' to stop.");
static int effect = 0; // Variable estática para mantener el efecto actual
static unsigned long lastTime = 0; // Variable para el temporizador

// Espera hasta que se reciba un número a través del Serial
if (Serial.available() > 0) {
    String input = Serial.readStringUntil('\n'); // Lee la entrada hasta el salto de
línea

    if (input.equals("s")) { // Verifica si la entrada es 's'
        effect = 0; // Detiene el efecto
        Serial.println("Effect stopped.");
        return;
    } else {
        int newEffect = input.toInt(); // Convierte la entrada a un número

        if (newEffect >= 1 && newEffect <= 123) { // Verifica si el ID del efecto está en
el rango válido
            effect = newEffect; // Actualiza el efecto si es válido
            Serial.print("Playing Effect #");

```

```

        Serial.println(effect);
    } else {
        Serial.println("Invalid effect ID. Please enter a number between 1 and 123.");
        effect = 0;
    }
}

// Limpia el buffer de entrada
while (Serial.available() > 0) {
    Serial.read();
}

// Reproduce el efecto seleccionado indefinidamente
if (effect > 0) { // se asegura de que hay un efecto válido
    if (millis() - lastTime > 500) { // Reproduce cada 500 ms
        drv.setWaveform(0, effect);
        drv.setWaveform(1, 0); // Fin de la forma de onda
        drv.go();
        lastTime = millis(); // Actualiza el tiempo
    }
}

}

void reset(int value) {
    if (value == 7) {
        Serial.println("Resetting...");
        setup();
    }
}

```

Anexo D

Código en visual studio

```
# -*- coding: utf-8 -*-

# Form implementation generated from reading ui file 'TEST_1.ui'

#

# Created by: PyQt5 UI code generator 5.15.9

#

# WARNING: Any manual changes made to this file will be lost when pyuic5 is
# run again. Do not edit this file unless you know what you are doing.

from PyQt5 import QtCore, QtGui, QtWidgets

from PyQt5.QtSerialPort import QSerialPort, QSerialPortInfo

import time

class Ui_Dialog(object):

    def setupUi(self, Dialog):

        Dialog.setObjectName("Dialog")

        Dialog.resize(1096, 577)

        self.txtEffect1 = QtWidgets.QLabel(Dialog)

        self.txtEffect1.setGeometry(QtCore.QRect(330, 340, 41, 16))

        font = QtGui.QFont()
```

```
font.setPointSize(9)

font.setBold(True)

font.setItalic(True)

font.setWeight(75)

self.txtEffect1.setFont(font)

self.txtEffect1.setFrameShadow(QtWidgets.QFrame.Plain)

self.txtEffect1.setMidLineWidth(0)

self.txtEffect1.setObjectName("txtEffect1")

self.txtEffect2 = QtWidgets.QLabel(Dialog)

self.txtEffect2.setGeometry(QtCore.QRect(410, 340, 12, 18))

font = QtGui.QFont()

font.setPointSize(9)

font.setBold(True)

font.setItalic(True)

font.setWeight(75)

self.txtEffect2.setFont(font)

self.txtEffect2.setObjectName("txtEffect2")

self.txtEffect3 = QtWidgets.QLabel(Dialog)

self.txtEffect3.setGeometry(QtCore.QRect(490, 340, 31, 16))
```

```
font = QtGui.QFont()

font.setPointSize(9)

font.setBold(True)

font.setItalic(True)

font.setWeight(75)

self.txtEffect3.setFont(font)

self.txtEffect3.setLayoutDirection(QtCore.Qt.LeftToRight)

self.txtEffect3.setObjectName("txtEffect3")

self.txtEffect4 = QtWidgets.QLabel(Dialog)

self.txtEffect4.setGeometry(QtCore.QRect(570, 340, 31, 16))

font = QtGui.QFont()

font.setPointSize(9)

font.setBold(True)

font.setItalic(True)

font.setWeight(75)

self.txtEffect4.setFont(font)

self.txtEffect4.setLayoutDirection(QtCore.Qt.LeftToRight)

self.txtEffect4.setObjectName("txtEffect4")

self.txtEffect5 = QtWidgets.QLabel(Dialog)
```

```
self.txtEffect5.setGeometry(QRect(650, 340, 31, 20))

font = QtGui.QFont()

font.setPointSize(9)

font.setBold(True)

font.setItalic(True)

font.setWeight(75)

self.txtEffect5.setFont(font)

self.txtEffect5.setLayoutDirection(QtCore.Qt.LeftToRight)

self.txtEffect5.setObjectName("txtEffect5")

self.txtEffect5_2 = QtWidgets.QLabel(Dialog)

self.txtEffect5_2.setGeometry(QRect(730, 340, 31, 20))

font = QtGui.QFont()

font.setPointSize(9)

font.setBold(True)

font.setItalic(True)

font.setWeight(75)

self.txtEffect5_2.setFont(font)

self.txtEffect5_2.setLayoutDirection(QtCore.Qt.LeftToRight)

self.txtEffect5_2.setObjectName("txtEffect5_2")
```

```
self.txtEffect5_3 = QtWidgets.QLabel(Dialog)

self.txtEffect5_3.setGeometry(QtCore.QRect(810, 340, 21, 20))

font = QtGui.QFont()

font.setPointSize(9)

font.setBold(True)

font.setItalic(True)

font.setWeight(75)

self.txtEffect5_3.setFont(font)

self.txtEffect5_3.setLayoutDirection(QtCore.Qt.LeftToRight)

self.txtEffect5_3.setObjectName("txtEffect5_3")

self.txtEffect5_4 = QtWidgets.QLabel(Dialog)

self.txtEffect5_4.setGeometry(QtCore.QRect(890, 340, 31, 20))

font = QtGui.QFont()

font.setPointSize(9)

font.setBold(True)

font.setItalic(True)

font.setWeight(75)

self.txtEffect5_4.setFont(font)

self.txtEffect5_4.setLayoutDirection(QtCore.Qt.LeftToRight)
```

```
self.txtEffect5_4.setObjectName("txtEffect5_4")

self.label = QtWidgets.QLabel(Dialog)

self.label.setGeometry(QtCore.QRect(540, 40, 171, 31))

font = QtGui.QFont()

font.setPointSize(15)

font.setBold(True)

font.setWeight(75)

self.label.setFont(font)

self.label.setObjectName("label")

self.label_2 = QtWidgets.QLabel(Dialog)

self.label_2.setGeometry(QtCore.QRect(150, 210, 181, 41))

font = QtGui.QFont()

font.setPointSize(11)

font.setBold(True)

font.setWeight(75)

self.label_2.setFont(font)

self.label_2.setTextFormat(QtCore.Qt.MarkdownText)

self.label_2.setAlignment(QtCore.Qt.AlignCenter)

self.label_2.setWordWrap(True)
```

```
self.label_2.setObjectName("label_2")

self.CONNECT = QtWidgets.QPushButton(Dialog)

self.CONNECT.setGeometry(QtCore.QRect(20, 170, 91, 51))

self.CONNECT.setObjectName("CONNECT")

self.DISCONNECT = QtWidgets.QPushButton(Dialog)

self.DISCONNECT.setGeometry(QtCore.QRect(20, 220, 91, 51))

self.DISCONNECT.setObjectName("DISCONNECT")

self.EXIT = QtWidgets.QPushButton(Dialog)

self.EXIT.setGeometry(QtCore.QRect(20, 270, 91, 51))

self.EXIT.setObjectName("EXIT")

self.RUN = QtWidgets.QPushButton(Dialog)

self.RUN.setGeometry(QtCore.QRect(490, 480, 93, 91))

self.RUN.setObjectName("RUN")

self.Repetitions = QtWidgets.QSlider(Dialog)

self.Repetitions.setGeometry(QtCore.QRect(550, 440, 160, 22))

self.Repetitions.setMaximum(10)

self.Repetitions.setOrientation(QtCore.Qt.Horizontal)

self.Repetitions.setObjectName("Repetitions")

self.label_3 = QtWidgets.QLabel(Dialog)
```

```
self.label_3.setGeometry(QtCore.QRect(580, 380, 171, 31))

font = QtGui.QFont()

font.setPointSize(15)

font.setBold(True)

font.setWeight(75)

self.label_3.setFont(font)

self.label_3.setObjectName("label_3")

self.numEffects = QtWidgets.QFrame(Dialog)

self.numEffects.setGeometry(QtCore.QRect(280, 160, 681, 181))

self.numEffects.setFrameShape(QtWidgets.QFrame.StyledPanel)

self.numEffects.setFrameShadow(QtWidgets.QFrame.Raised)

self.numEffects.setObjectName("numEffects")

self.EffeReg1 = QtWidgets.QSlider(self.numEffects)

self.EffeReg1.setGeometry(QtCore.QRect(50, 0, 22, 160))

self.EffeReg1.setLayoutDirection(QtCore.Qt.RightToLeft)

self.EffeReg1.setMaximum(123)

self.EffeReg1.setOrientation(QtCore.Qt.Vertical)

self.EffeReg1.setObjectName("EffeReg1")

self.EffeReg2 = QtWidgets.QSlider(self.numEffects)
```

```
self.EffeReg2.setGeometry(QRect(130, 0, 22, 160))
```

```
self.EffeReg2.setMaximum(123)
```

```
self.EffeReg2.setOrientation(QtCore.Qt.Vertical)
```

```
self.EffeReg2.setObjectName("EffeReg2")
```

```
self.EffeReg3 = QtWidgets.QSlider(self.numEffects)
```

```
self.EffeReg3.setGeometry(QRect(210, 0, 22, 160))
```

```
self.EffeReg3.setMaximum(123)
```

```
self.EffeReg3.setOrientation(QtCore.Qt.Vertical)
```

```
self.EffeReg3.setObjectName("EffeReg3")
```

```
self.EffeReg4 = QtWidgets.QSlider(self.numEffects)
```

```
self.EffeReg4.setGeometry(QRect(290, 0, 22, 160))
```

```
self.EffeReg4.setMaximum(123)
```

```
self.EffeReg4.setOrientation(QtCore.Qt.Vertical)
```

```
self.EffeReg4.setObjectName("EffeReg4")
```

```
self.EffeReg5 = QtWidgets.QSlider(self.numEffects)
```

```
self.EffeReg5.setGeometry(QRect(370, 0, 22, 160))
```

```
self.EffeReg5.setMaximum(123)
```

```
self.EffeReg5.setOrientation(QtCore.Qt.Vertical)
```

```
self.EffeReg5.setObjectName("EffeReg5")
```

```
self.EffeReg6 = QtWidgets.QSlider(self.numEffects)

self.EffeReg6.setGeometry(QtCore.QRect(440, 0, 22, 160))

self.EffeReg6.setMaximum(123)

self.EffeReg6.setOrientation(QtCore.Qt.Vertical)

self.EffeReg6.setObjectName("EffeReg6")

self.EffeReg7 = QtWidgets.QSlider(self.numEffects)

self.EffeReg7.setGeometry(QtCore.QRect(530, 0, 22, 160))

self.EffeReg7.setMaximum(123)

self.EffeReg7.setOrientation(QtCore.Qt.Vertical)

self.EffeReg7.setObjectName("EffeReg7")

self.EffeReg8 = QtWidgets.QSlider(self.numEffects)

self.EffeReg8.setGeometry(QtCore.QRect(610, 0, 22, 160))

self.EffeReg8.setMaximum(123)

self.EffeReg8.setOrientation(QtCore.Qt.Vertical)

self.EffeReg8.setObjectName("EffeReg8")

self.CALIBRA1 = QtWidgets.QPushButton(Dialog)

self.CALIBRA1.setGeometry(QtCore.QRect(20, 340, 111, 28))

self.CALIBRA1.setObjectName("CALIBRA1")

self.CALIBRA2 = QtWidgets.QPushButton(Dialog)
```

```
self.CALIBRA2.setGeometry(QtCore.QRect(20, 380, 111, 28))

self.CALIBRA2.setObjectName("CALIBRA2")

self.CALIBRA3 = QtWidgets.QPushButton(Dialog)

self.CALIBRA3.setGeometry(QtCore.QRect(20, 420, 111, 28))

self.CALIBRA3.setObjectName("CALIBRA3")

self.Reg1Val = QtWidgets.QLabel(Dialog)

self.Reg1Val.setGeometry(QtCore.QRect(290, 100, 91, 51))

self.Reg1Val.setText("")

self.Reg1Val.setObjectName("Reg1Val")

self.RESET = QtWidgets.QPushButton(Dialog)

self.RESET.setGeometry(QtCore.QRect(660, 480, 93, 91))

self.RESET.setObjectName("RESET")

self.Reg2Val = QtWidgets.QLabel(Dialog)

self.Reg2Val.setGeometry(QtCore.QRect(380, 99, 81, 51))

self.Reg2Val.setText("")

self.Reg2Val.setObjectName("Reg2Val")

self.Reg3Val = QtWidgets.QLabel(Dialog)

self.Reg3Val.setGeometry(QtCore.QRect(460, 100, 71, 51))

self.Reg3Val.setText("")
```

```
self.Reg3Val.setObjectName("Reg3Val")

self.Reg6Val = QtWidgets.QLabel(Dialog)

self.Reg6Val.setGeometry(QtCore.QRect(694, 100, 71, 51))

self.Reg6Val.setText("")

self.Reg6Val.setObjectName("Reg6Val")

self.Reg4Val = QtWidgets.QLabel(Dialog)

self.Reg4Val.setGeometry(QtCore.QRect(530, 100, 81, 51))

self.Reg4Val.setText("")

self.Reg4Val.setObjectName("Reg4Val")

self.Reg5Val = QtWidgets.QLabel(Dialog)

self.Reg5Val.setGeometry(QtCore.QRect(610, 100, 81, 51))

self.Reg5Val.setText("")

self.Reg5Val.setObjectName("Reg5Val")

self.Reg7Val = QtWidgets.QLabel(Dialog)

self.Reg7Val.setGeometry(QtCore.QRect(774, 105, 81, 51))

self.Reg7Val.setText("")

self.Reg7Val.setObjectName("Reg7Val")

self.Reg8Val = QtWidgets.QLabel(Dialog)

self.Reg8Val.setGeometry(QtCore.QRect(854, 105, 91, 51))
```

```
self.Reg8Val.setText("")

self.Reg8Val.setObjectName("Reg8Val")

self.RepeatsVal = QtWidgets.QLabel(Dialog)

self.RepeatsVal.setGeometry(QtCore.QRect(630, 420, 55, 16))

self.RepeatsVal.setObjectName("RepeatsVal")

self.frame = QtWidgets.QFrame(Dialog)

self.frame.setGeometry(QtCore.QRect(90, 60, 120, 80))

self.frame.setFrameShape(QtWidgets.QFrame.StyledPanel)

self.frame.setFrameShadow(QtWidgets.QFrame.Raised)

self.frame.setObjectName("frame")

self.COM_PORT = QtWidgets.QComboBox(self.frame)

self.COM_PORT.setGeometry(QtCore.QRect(10, 30, 73, 22))

self.COM_PORT.setObjectName("COM_PORT")

self.label_4 = QtWidgets.QLabel(self.frame)

self.label_4.setGeometry(QtCore.QRect(0, 0, 91, 16))

font = QtGui.QFont()

font.setPointSize(10)

font.setBold(True)

font.setWeight(75)
```

```
self.label_4.setFont(font)

self.label_4.setObjectName("label_4")

self.txtEffect1.raise_()

self.txtEffect2.raise_()

self.txtEffect3.raise_()

self.txtEffect4.raise_()

self.txtEffect5.raise_()

self.txtEffect5_2.raise_()

self.txtEffect5_3.raise_()

self.txtEffect5_4.raise_()

self.label.raise_()

self.label_2.raise_()

self.CONNECT.raise_()

self.DISCONNECT.raise_()

self.EXIT.raise_()

self.RUN.raise_()

self.Repetitions.raise_()

self.label_3.raise_()

self.numEffects.raise_()
```

```
self.CALIBRA1.raise_()
```

```
self.CALIBRA2.raise_()
```

```
self.CALIBRA3.raise_()
```

```
self.Reg1Val.raise_()
```

```
self.RESET.raise_()
```

```
self.Reg2Val.raise_()
```

```
self.Reg3Val.raise_()
```

```
self.Reg6Val.raise_()
```

```
self.Reg4Val.raise_()
```

```
self.Reg5Val.raise_()
```

```
self.Reg7Val.raise_()
```

```
self.RepeatsVal.raise_()
```

```
self.Reg8Val.raise_()
```

```
self.frame.raise_()
```

```
self.retranslateUi(Dialog)
```

```
QtCore.QMetaObject.connectSlotsByName(Dialog)
```

```
def retranslateUi(self, Dialog):
```

```

_translate = QtCore.QCoreApplication.translate

Dialog.setWindowTitle(_translate("Dialog", "Dialog"))

self.txtEffect1.setText(_translate("Dialog", "1"))

self.txtEffect2.setText(_translate("Dialog", "2"))

self.txtEffect3.setText(_translate("Dialog", "3"))

self.txtEffect4.setText(_translate("Dialog", "4"))

self.txtEffect5.setText(_translate("Dialog", "5"))

self.txtEffect5_2.setText(_translate("Dialog", "6"))

self.txtEffect5_3.setText(_translate("Dialog", "7"))

self.txtEffect5_4.setText(_translate("Dialog", "8"))

self.label.setText(_translate("Dialog", "NO. REGS"))

self.label_2.setText(_translate("Dialog", "EFFECT (0 to 123 )"))

self.CONNECT.setText(_translate("Dialog", "Connect"))

self.DISCONNECT.setText(_translate("Dialog", "Disconnect"))

self.EXIT.setText(_translate("Dialog", "Exit"))

self.RUN.setText(_translate("Dialog", "Run"))

self.label_3.setText(_translate("Dialog", "REPEATS"))

self.CALIBRA1.setText(_translate("Dialog", "CALIBRATION 1"))

self.CALIBRA2.setText(_translate("Dialog", "CALIBRATION 2"))

```

```
self.CALIBRA3.setText(_translate("Dialog", "CALIBRATION 3"))
```

```
self.RESET.setText(_translate("Dialog", "Reset"))
```

```
self.RepeatsVal.setText(_translate("Dialog", "0"))
```

```
self.label_4.setText(_translate("Dialog", "Serial COM"))
```

```
# Inicial Setup
```

```
self.CONNECT.setEnabled(True)
```

```
self.DISCONNECT.setEnabled(False)
```

```
self.EXIT.setEnabled(False)
```

```
self.RUN.setEnabled(False)
```

```
self.numEffects.setEnabled(False)
```

```
self.Repetitions.setEnabled(False)
```

```
self.CALIBRA1.setEnabled(False)
```

```
self.CALIBRA2.setEnabled(False)
```

```
self.CALIBRA3.setEnabled(False)
```

```
self.populate_com_ports()
```

```
# Serial Setup
```

```
self.serial = QSerialPort()
```

```
self.serial.setDataTerminalReady(True)
```

```
# Button Setup
```

```
self.CONNECT.clicked.connect(self.fn_Connect)
```

```
self.DISCONNECT.clicked.connect(self.fn_Disconnect)
```

```
self.RUN.clicked.connect(self.fn_Run)
```

```
self.EXIT.clicked.connect(self.fn_Exit)
```

```
self.RESET.clicked.connect(self.fn_Reset)
```

```
self.CALIBRA1.clicked.connect(self.fn_Calibration1)
```

```
self.CALIBRA2.clicked.connect(self.fn_Calibration2)
```

```
self.CALIBRA3.clicked.connect(self.fn_Calibration3)
```

```
# Conectar sliders a sus funciones
```

```
self.EffeReg1.valueChanged.connect(self.update_reg1)
```

```
self.EffeReg2.valueChanged.connect(self.update_reg2)
```

```
self.EffeReg3.valueChanged.connect(self.update_reg3)
```

```
self.EffeReg4.valueChanged.connect(self.update_reg4)
```

```
self.EffeReg5.valueChanged.connect(self.update_reg5)
```

```
self.EffeReg6.valueChanged.connect(self.update_reg6)
```

```
self.EffeReg7.valueChanged.connect(self.update_reg7)
```

```
self.EffeReg8.valueChanged.connect(self.update_reg8)
```

```
self.Repetitions.valueChanged.connect(self.update_reps)
```

```
self.sliders = [self.EffeReg1, self.EffeReg2, self.EffeReg3, self.EffeReg4,
```

```
self.EffeReg5, self.EffeReg6, self.EffeReg7, self.EffeReg8]
```

```
self.slider_texts = {
```

```
    0: "0",
```

```
    1: "1 Strong \n Click- 100%",
```

```
    2: "2 Strong \n Click - 60%",
```

```
    3: "6 Sharp \n Click - 30%%",
```

```
    4: "4 Sharp \n Click - 100%",
```

```
    5: "5 Sharp \n Click - 60%",
```

```
    6: "6 Sharp \n Click - 30%",
```

```
    7: "7 Soft \n Bump - 100%",
```

```
    8: "8 Soft \n Bump - 60%",
```

```
    9: "9 Soft \n Bump - 30%",
```

```
    10:"10 Double \n Click - 100%",
```

```
    11:"11 Double \n Click - 60%",
```

```
    12:"12 Triple \n Click - 100%",
```

```
    13:"13 Soft \n Fuzz - 60%",
```

14: "14 Strong \n Buzz - 100%",

15: "15 750 ms \n Alert - 100%",

16: "16 1000 ms \n Alert - 100%",

17: "17 Strong \n Click 1 - 100%",

18: "18 Strong \n Click 2 - 80%",

19: "19 Strong \n Click 3 - 60%",

20: "20 Strong \n Click 4 - 30%",

21: "21 Medium \n Click 1 - 100%",

22: "22 Medium \n Click 2 - 80%",

23: "23 Medium \n Click 3 - 60%",

24: "24 Sharp \n Tick 1 - 100%",

25: "25 Sharp \n Tick 2 - 80%",

26: "26 Sharp \n Tick 3 - 60%",

27: "27 Short \n Double Click Strong 1 - 100%",

28: "28 Short \n Double Click Strong 2 - 80%",

29: "29 Short \n Double Click \n Strong 3 - 60%",

30: "30 Short \n Double Click \n Strong 4 - 30%",

31: "31 Short \n Double Click \n Medium 1 - 100%",

32: "32 Short \n Double Click \n Medium 2 - 80%",

33: "33 Short \n Double Click \n Medium 3 - 60%",

34: "34 Short \n Double Sharp Tick 1 - 100%",

35: "35 Short \n Sharp Tick 2 - 80%",

36: "36 Short \n Sharp Tick 3 - 60%",

37: "37 Long \n Double Sharp Click \n Strong 1 - 100%",

38: "38 Long \n Sharp Click \n Strong 2 - 80%",

39: "39 Long \n Sharp Click \n Strong 3 - 60%",

40: "40 Long \n Sharp Click \n Strong 4 - 30%",

41: "41 Long \n Sharp Click \n Medium 1 - 100%",

42: "42 Long \n Double Sharp Click \n Medium 2 - 80%",

43: "43 Long \n Double Sharp Click \n Medium 3 - 60%",

44: "44 Long \n Double Sharp \n Tick 1 - 100%",

45: "45 Long \n Double Sharp \n Tick 2 - 80%",

46: "46 Long \n Double Sharp \n Tick 3 - 60%",

47: "47 Buzz 1 - 100%",

48: "48 Buzz 2 - 80%",

49: "49 Buzz 3 - 60%",

50: "50 Buzz 4 - 40%",

51: "51 Buzz 5 - 20%",

52: "52 Buzz 6 - 10%",

53: "53 Pulsing Strong \n1 - 100%",

54: "54 Pulsing Strong \n2 - 60%",

55: "55 Pulsing Medium \n1 - 100%",

56: "56 Pulsing Medium \n2 - 60%",

57: "57 Pulsing Sharp \n1 - 100%",

58: "58 Pulsing Sharp \n2 - 60%",

59: "59 Transition Click \n1 - 100%",

60: "60 Transition Click \n2 - 80%",

61: "61 Transition Click \n3 - 60%",

62: "62 Transition Click \n4 - 30%",

63: "63 Transition Click \n5 - 20%",

64: "64 Transition Click \n6 - 10%",

65: "65 Transition Hum \n 1 - 100%",

66: "66 Transition Hum \n 2 - 80%",

67: "67 Transition Hum \n 3 - 60%",

68: "68 Transition Hum \n 4 - 40%",

69: "69 Transition Hum \n 5 - 20%",

70: "70 Transition Hum \n 6 - 10%",

71: "71 Transition Ramp Down \n Long Smooth \n 1 - 100 to 0%",
72: "72 Transition Ramp Down \n Long Smooth \n 2 - 100 to 0%",
73: "73 Transition Ramp Down \n Medium Smooth \n 1 - 50 to 0%",
74: "74 Transition Ramp Down \n Medium Smooth \n 2 - 50 to 0%",
75: "75 Transition \n Ramp Down Short Smooth \n 2 - 100 to 0%",
76: "76 Transition \n Ramp Down Long Sharp \n 1 - 100 to 0%",
77: "77 Transition \n Ramp Down Long Sharp \n 2 - 100 to 0%",
78: "78 Transition \n Ramp Down Medium Sharp \n 1 - 100 to 0%",
79: "79 Transition \n Ramp Down Medium Sharp 2 - 100 to 0%",
80: "80 Transition Ramp Down Short Sharp 1 - 100 to 0%",
81: "81 Transition Ramp Down Short Sharp 2 - 100 to 0%",
82: "82 Transition Ramp Up Long Smooth 1 - 0 to 100%",
83: "83 Transition Ramp Up \n Long Smooth \n 1 - 0 to 100%",
84: "84 Transition Ramp Up \n Long Smooth \n 2 - 0 to 100%",
85: "85 Transition Ramp Up \n Medium Smooth \n 1 - 0 to 100%",
86: "86 Transition Ramp Up \n Medium Smooth \n 2 - 0 to 100%",
87: "87 Transition Ramp Up \n Short Smooth \n 1 - 0 to 100%",
88: "88 Transition Ramp Up \n Short Smooth \n 2 - 0 to 100%",
89: "89 Transition Ramp Up \n Short Sharp \n 1 - 0 to 100%",

90: "90 Transition Ramp Up \n Short Sharp \n 2 - 0 to 100%",

91: "91 Transition Ramp Down \n Long Sharp \n 1 - 100 to 0%",

92: "92 Transition Ramp Down \n Long Sharp \n 2 - 100 to 0%",

93: "93 Transition Ramp Down \n Medium Sharp \n 1 - 50 to 0%",

94: "94 Transition Ramp Down \n Medium Sharp \n 2 - 50 to 0%",

95: "95 Transition Ramp Down \n Short Sharp \n 1 - 50 to 0%",

96: "96 Transition Ramp Down \n Short Sharp \n 2 - 50 to 0%",

97: "97 Transition Ramp Up \n Long Sharp \n 1 - 0 to 100%",

98: "98 Transition Ramp Up \n Long Sharp \n 2 - 0 to 100%",

99: "99 Transition Ramp Up \n Medium Sharp \n 1 - 0 to 100%",

100: "100 Transition Ramp Up \n Medium Sharp \n 2 - 0 to 100%",

101: "101 Transition Ramp Up \n Short Sharp \n 1 - 0 to 100%",

102: "102 Transition Ramp Up \n Short Sharp \n 2 - 0 to 100%",

103: "103 Transition Ramp Down \n Long Smooth \n 1 - 50 to 0%",

104: "104 Transition Ramp Down \n Long Smooth \n 2 - 50 to 0%",

105: "105 Transition Ramp Down \n Medium Smooth \n 1 - 50 to 0%",

106: "106 Transition Ramp Down \n Medium Smooth \n 2 - 50 to 0%",

107: "107 Transition Ramp Down \n Short Smooth \n 1 - 50 to 0%",

108: "108 Transition Ramp Down \n Short Smooth \n 2 - 50 to 0%",

109: "109 Transition Ramp Down \n Long Sharp \n 1 - 100 to 0%",
110: "110 Transition Ramp Down \n Long Sharp \n 2 - 100 to 0%",
111: "111 Transition Ramp Down \n Medium Sharp \n 1 - 50 to 0%",
112: "112 Transition Ramp Down \n Medium Sharp \n 2 - 50 to 0%",
113: "113 Transition Ramp Down \n Short Sharp \n 1 - 50 to 0%",
114: "114 Transition Ramp Down \n Short Sharp \n 2 - 50 to 0%",
115: "115 Transition Ramp Up \n Long Sharp \n 1 - 0 to 100%",
116: "116 Transition Ramp Up \n Short Sharp \n 1 - 0 to 50%",
117: "117 Transition Ramp Up \n Short Sharp \n 2 - 0 to 50%",
118: "118 Long Buzz for programatic \n stopping- 100%",
119: "119 Smooth Hum 1 \n (No kick or brake pulse) - 50%",
120: "120 Smooth Hum 2 \n (No kick or brake pulse) - 40%",
121: "121 Smooth Hum 3 \n (No kick or brake pulse) - 30%",
122: "122 Smooth Hum 4 \n (No kick or brake pulse) - 20%",
123: "123 Smooth Hum 5 \n (No kick or brake pulse) - 10%",

}

```

def populate_com_ports(self):

    ports = QSerialPortInfo.availablePorts()

    for port in ports:

        self.COM_PORT.addItem(port.portName(

def fn_Connect(self):

    self.serial.setBaudRate(QSerialPort.Baud9600)

    self.serial.setPortName(self.COM_PORT.currentText())

    self.serial.setDataBits(QSerialPort.Data8)

    self.serial.setParity(QSerialPort.NoParity)

    self.serial.setStopBits(QSerialPort.OneStop)


if self.serial.open(QtCore.QIODevice.ReadWrite):

    print("Connected")

    self.CONNECT.setEnabled(False)

    self.DISCONNECT.setEnabled(True)

    self.RUN.setEnabled(True)

    self.CONNECT.setEnabled(False)

    self.DISCONNECT.setEnabled(True)

    self.EXIT.setEnabled(True)

```

```

        self.numEffects.setEnabled(False)

        self.CALIBRA1.setEnabled(True)

        self.CALIBRA2.setEnabled(True)

        self.CALIBRA3.setEnabled(True)

    else:

        print("Failed to connect")

def fn_Disconnect(self):

    self.serial.close()

    self.CONNECT.setEnabled(True)

    self.DISCONNECT.setEnabled(False)

    self.EXIT.setEnabled(True)

    self.RUN.setEnabled(False)

    self.numEffects.setEnabled(False)

    self.Repetitions.setEnabled(False)

    print("Disconnected")

    data = "200\n"

    self.serial.write(data.encode())

    print("200")

```

```
self.reset_sliders()
```

```
def fn_Reset(self):
```

```
    print("Resetting")
```

```
    data = "200\n"
```

```
    self.serial.write(data.encode())
```

```
    print("200")
```

```
    self.CALIBRA1.setEnabled(True)
```

```
    self.CALIBRA2.setEnabled(True)
```

```
    self.CALIBRA3.setEnabled(True)
```

```
    self.numEffects.setEnabled(False)
```

```
    self.reset_sliders()
```

```
def fn_Run(self):
```

```
    non_zero_values = [slider.value() for slider in self.sliders if slider.value() != 0]
```

```
    count_non_zero = len(non_zero_values)
```

```
    # Enviar el número de sliders con valor diferente de cero
```

```
    data = str(count_non_zero) + "\n"
```

```
self.serial.write(data.encode())
```

```
print(data)
```

```
time.sleep(0.1)
```

```
# Enviar los valores de los sliders con valor diferente de cero
```

```
for value in non_zero_values:
```

```
    data = str(value) + "\n"
```

```
    self.serial.write(data.encode())
```

```
    print(data)
```

```
    time.sleep(0.2)
```

```
# Enviar el valor del slider de Repetitions
```

```
val = self.Repetitions.value()
```

```
dat = str(val) + "\n"
```

```
self.serial.write(dat.encode())
```

```
print(dat)
```

```
#Condiciones despues del segundo click
```

```
def fn_Secondclick(self):
```

```
self.CALIBRA1.setEnabled(False)
```

```
self.CALIBRA2.setEnabled(False)
```

```
self.CALIBRA3.setEnabled(False)
```

```
self.numEffects.setEnabled(True)
```

```
self.EffeReg1.setEnabled(True)
```

```
self.EffeReg2.setEnabled(False)
```

```
self.EffeReg3.setEnabled(False)
```

```
self.EffeReg4.setEnabled(False)
```

```
self.EffeReg5.setEnabled(False)
```

```
self.EffeReg6.setEnabled(False)
```

```
self.EffeReg7.setEnabled(False)
```

```
self.EffeReg8.setEnabled(False)
```

```
self.RESET.setEnabled(True)
```

```
def fn_Exit(self):
```

```
    self.serial.close()
```

```
    QtWidgets.QApplication.quit()
```

```
def fn_Calibration1(self):
```

```
data = "1\n"
```

```
self.serial.write(data.encode())
```

```
print("1")
```

```
self.fn_Secondclick()
```

```
def fn_Calibration2(self):
```

```
data = "2\n"
```

```
self.serial.write(data.encode())
```

```
print("2")
```

```
self.fn_Secondclick()
```

```
def fn_Calibration3(self):
```

```
data = "3\n"
```

```
self.serial.write(data.encode())
```

```
print("3")
```

```
self.fn_Secondclick()
```

```
# Valores de SliderVal en textos
```

```
def update_reg1(self, value):
```

```
self.Reg1Val.setText(self.slider_texts.get(value, ""))
```

```
if value > 0:
```

```
    self.RUN.setEnabled(True)
```

```
    self.Repetitions.setEnabled(True)
```

```
    self.EffeReg2.setEnabled(True)
```

```
else:
```

```
    self.RUN.setEnabled(False)
```

```
    self.Repetitions.setEnabled(False)
```

```
    self.EffeReg2.setEnabled(False)
```

```
    self.EffeReg3.setEnabled(False)
```

```
    self.EffeReg4.setEnabled(False)
```

```
    self.EffeReg5.setEnabled(False)
```

```
    self.EffeReg6.setEnabled(False)
```

```
    self.EffeReg7.setEnabled(False)
```

```
    self.EffeReg8.setEnabled(False)
```

```
def update_reg2(self, value):
```

```
self.Reg2Val.setText(self.slider_texts.get(value, ""))
```

```
if value > 0:
```

```
    self.EffeReg3.setEnabled(True)
```

```
else:
```

```
    self.EffeReg3.setEnabled(False)
```

```
    self.EffeReg4.setEnabled(False)
```

```
    self.EffeReg5.setEnabled(False)
```

```
    self.EffeReg6.setEnabled(False)
```

```
    self.EffeReg7.setEnabled(False)
```

```
    self.EffeReg8.setEnabled(False)
```

```
def update_reg3(self, value):
```

```
    self.Reg3Val.setText(self.slider_texts.get(value, ""))
```

```
if value > 0:
```

```
    self.EffeReg4.setEnabled(True)
```

```
else:
```

```
self.EffeReg4.setEnabled(False)
```

```
self.EffeReg5.setEnabled(False)
```

```
self.EffeReg6.setEnabled(False)
```

```
self.EffeReg7.setEnabled(False)
```

```
self.EffeReg8.setEnabled(False)
```

```
def update_reg4(self, value):
```

```
    self.Reg4Val.setText(self.slider_texts.get(value, ""))
```

```
    if value > 0:
```

```
        self.EffeReg5.setEnabled(True)
```

```
    else:
```

```
        self.EffeReg5.setEnabled(False)
```

```
        self.EffeReg6.setEnabled(False)
```

```
        self.EffeReg7.setEnabled(False)
```

```
        self.EffeReg8.setEnabled(False)
```

```
def update_reg5(self, value):
```

```
    self.Reg5Val.setText(self.slider_texts.get(value, ""))
```

```

if value > 0:

    self.EffeReg6.setEnabled(True)


else:

    self.EffeReg6.setEnabled(False)

    self.EffeReg7.setEnabled(False)

    self.EffeReg8.setEnabled(False)


def update_reg6(self, value):

    self.Reg6Val.setText(self.slider_texts.get(value, ""))

    if value > 0:

        self.EffeReg7.setEnabled(True)


    else:

        self.EffeReg7.setEnabled(False)

        self.EffeReg8.setEnabled(False)


def update_reg7(self, value):

    self.Reg7Val.setText(self.slider_texts.get(value, ""))

```

```
if value > 0:
```

```
    self.EffeReg8.setEnabled(True)
```

```
else:
```

```
    self.EffeReg8.setEnabled(False)
```

```
def update_reg8(self, value):
```

```
    self.Reg8Val.setText(self.slider_texts.get(value, ""))
```

```
def update_reps(self, value):
```

```
    self.RepeatsVal.setText(str(value))
```

```
def reset_sliders(self):
```

```
    self.EffeReg1.setValue(0)
```

```
    self.EffeReg2.setValue(0)
```

```
    self.EffeReg3.setValue(0)
```

```
    self.EffeReg4.setValue(0)
```

```
    self.EffeReg5.setValue(0)
```

```
self.EffeReg6.setValue(0)
```

```
self.EffeReg7.setValue(0)
```

```
self.EffeReg8.setValue(0)
```

```
self.Repetitions.setValue(0)
```

```
if __name__ == "__main__":
```

```
    import sys
```

```
    app = QtWidgets.QApplication(sys.argv)
```

```
    Dialog = QtWidgets.QDialog()
```

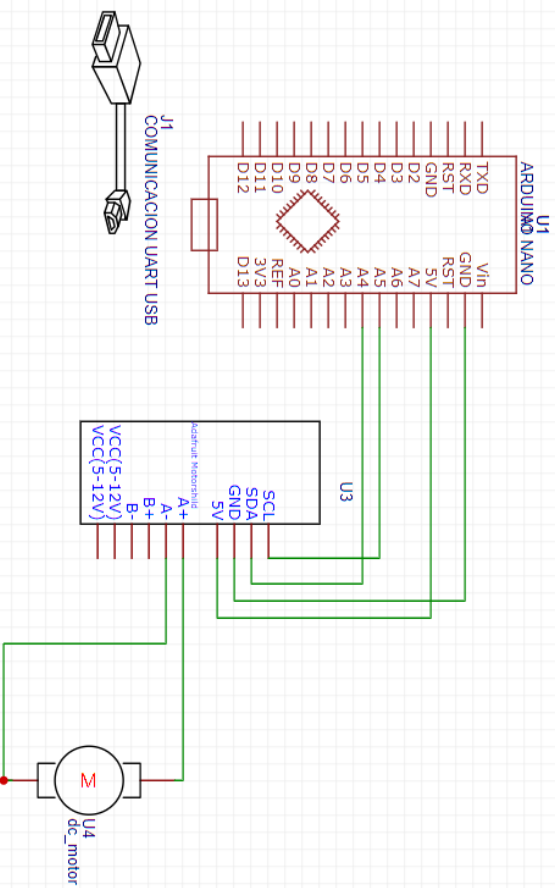
```
    ui = Ui_Dialog()
```

```
    ui.setupUi(Dialog)
```

```
    Dialog.show()
```

```
    sys.exit(app.exec_())
```

Anexo 1

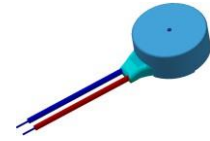


TITLE:		REV: 1.0
Circuito de prueba para LRA'S		
Date:	2018-03-29	Sheet: 1/2
EasyEDA V5.3.14	Drawn By: Azucena Tlacumilco	2018-03-29

ANEXO 2



PUIaudio



Data Sheet

HD-LA0803-LW10-R

PUI Audio's 8mm diameter by 3mm high **HD-LA0803-LW10-R** is a coin-style linear resonant actuator haptic device that employs a spring and mass system for fast response times and strong vibration in a small package.

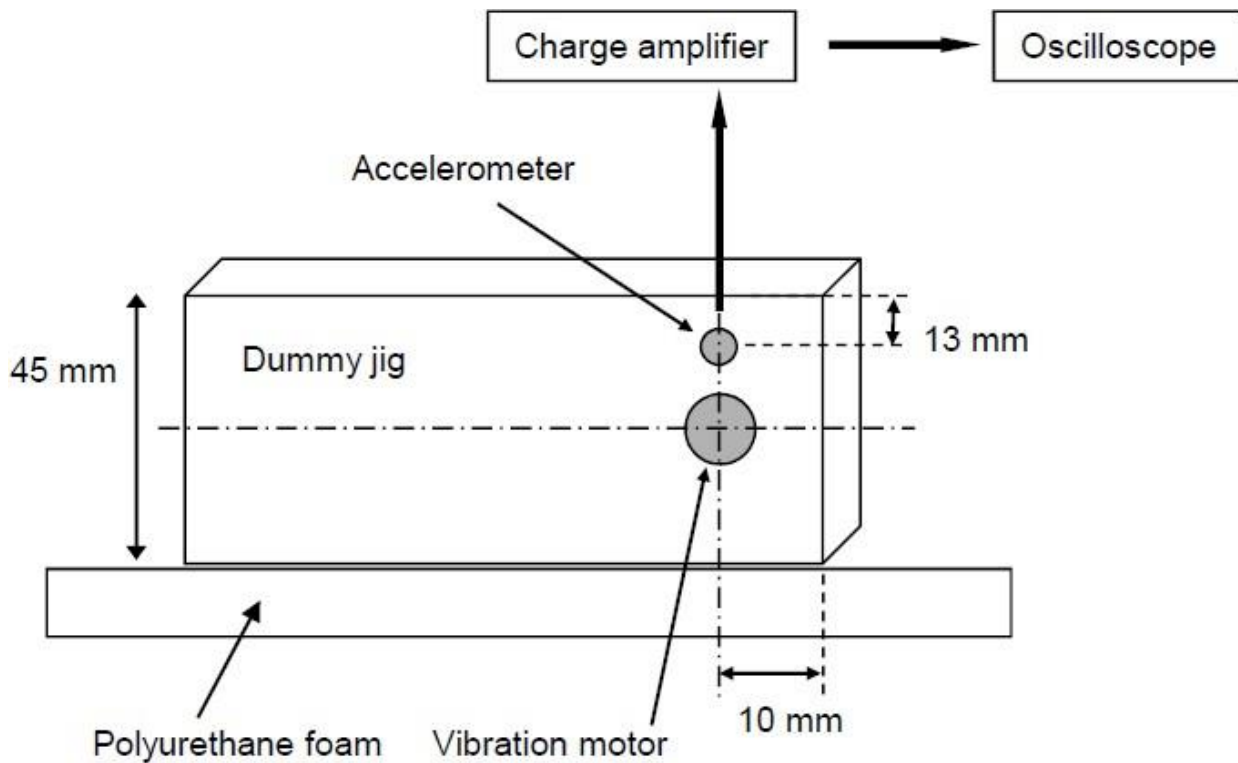
Features:

- Designed to be driven with a 235 Hz sinewave
- Convenient lead wires and double-sided tape for easy implementation
- Generates 1.5Gs with no more than 45 dBA acoustic noise

Specifications

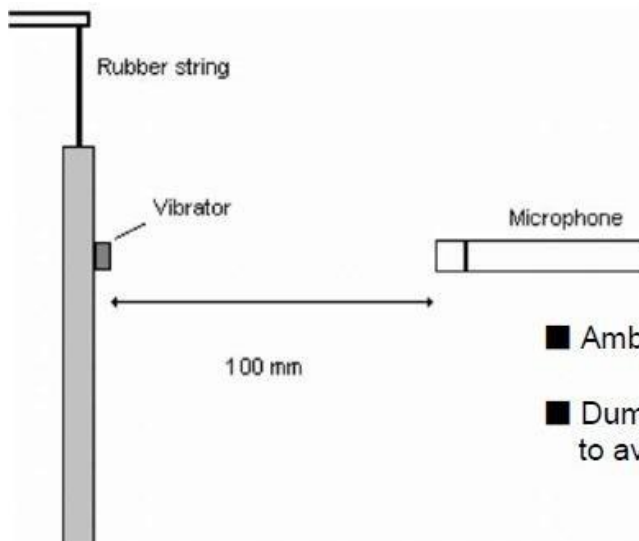
Parameters	Values	Units
Input Voltage (AC Sinewave)	2	Vrms
Operational Frequency	235	Hz
Coil Resistance	25 ±15%	Ohms
Vibration	1.5 ±0.3	Gs RMS
Max Current Draw (at 2 VRMS)	90	mA
Rise Time (50% of steady state)	50	mS
Fall Time (50% of steady state)	80	mS
Magnet Material	NdFeB	-
Acoustic Noise (Max)	45	dBA
Housing Material	SPC	-
Acceptable Soldering Methods	Hand Solder	-
Environmental Compliances	RoHS/REACH	-
Weight	1	Grams
Storage Temperature	-40 ~ +80	°C
Operating Temperature	-20 ~ +70	°C

Measurement Methods



■ Specification of the dummy Jig

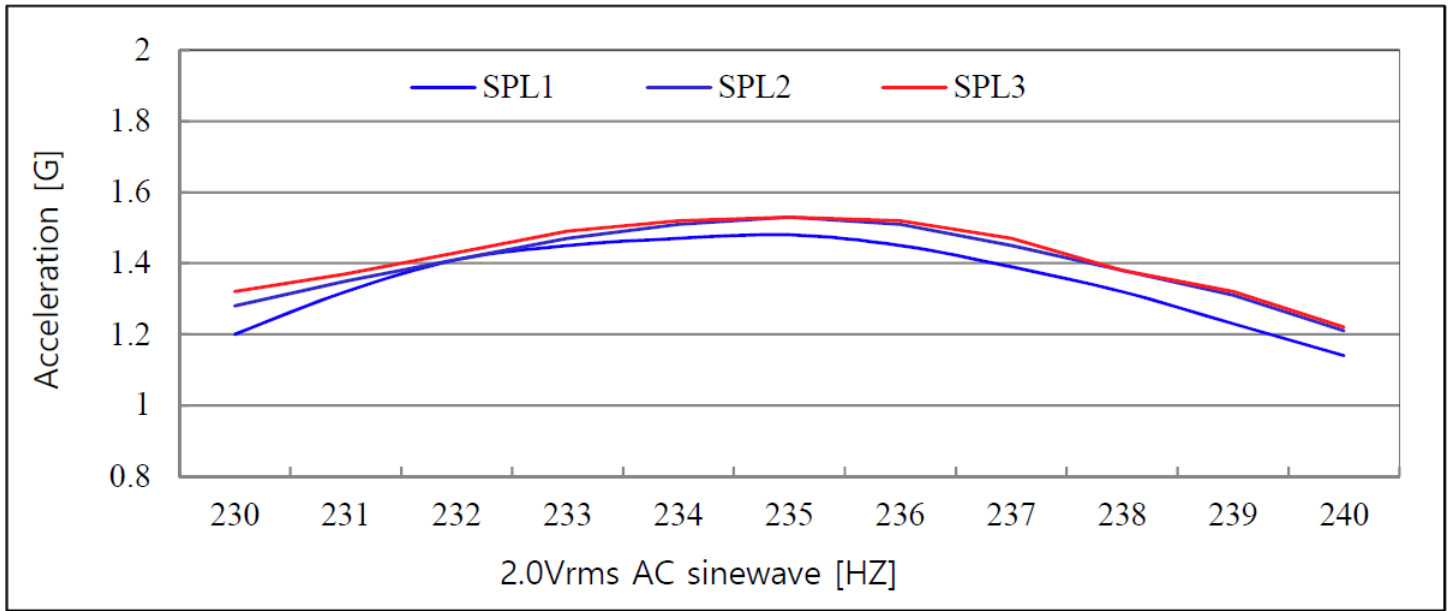
- Size : 45mm x 105mm x 15mm
- Weight : 100gr
- Material : Bakelite
- Vibration motor should be attached by using a double-sided tape.



■ Ambient noise level : 23dB(A)

■ Dummy jig must be suspended by a rubber string to avoid disturbance.

Typical Frequency Response Characteristics

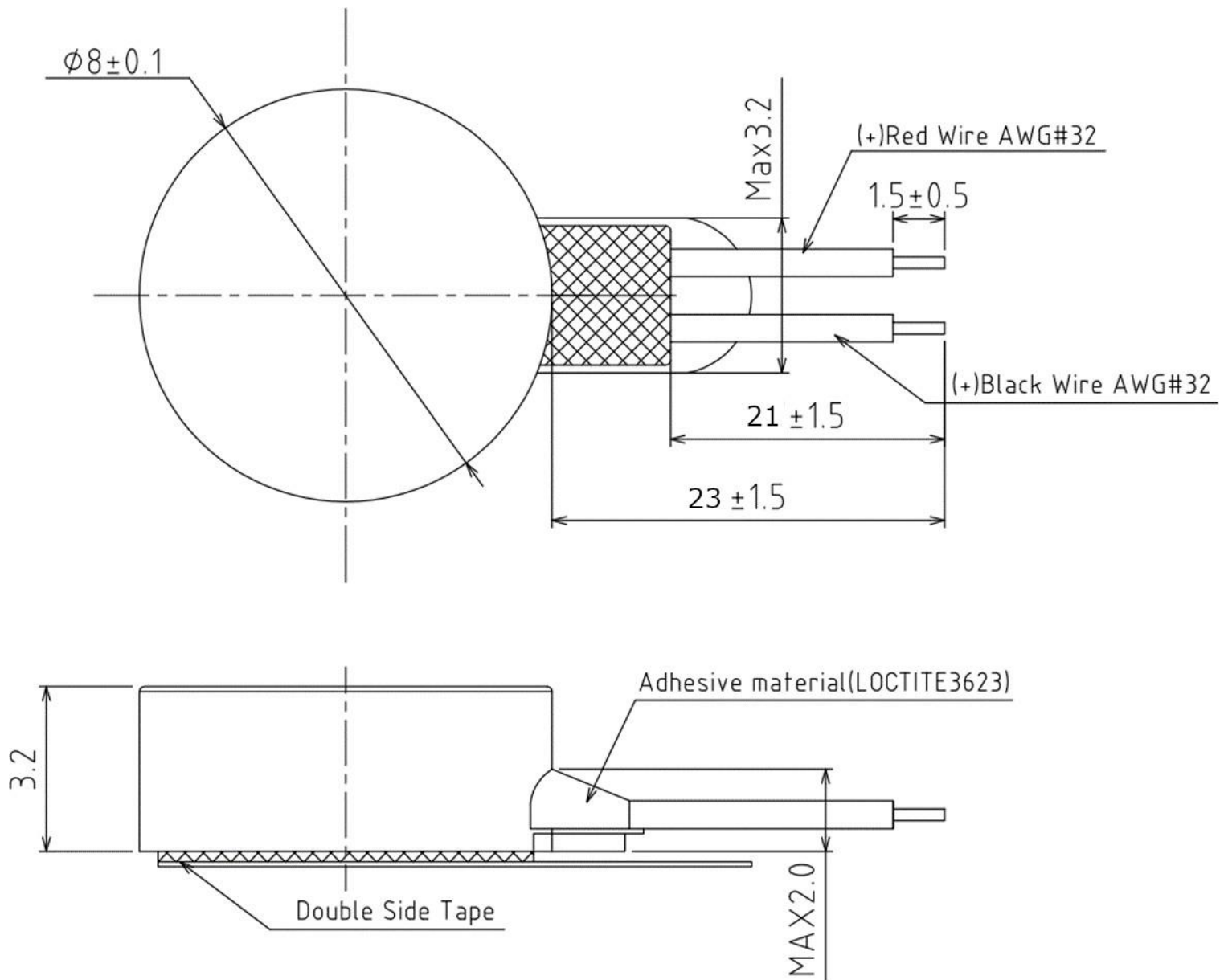


Reliability Testing

Type of Test	Test Specifications
High Temperature Test	96 hours at 80°C
Low Temperature Test	96 hours at -40°C
H2S Corrosion Resistance Test	Concentration: 3 ±1 PPM Test Environment: 40°C, humidity 80% RH Test Duration Time: 240 hours
Temperature Cycle Testing	Temperature: -40°C to +80°C Test Duration: 30 minutes Test Cycles: 15 cycles
Vibration Test	3mm deflection at 1000 cycles per minute, for 10 minutes in each axis (X, Y, Z).
Drop Test	Weight: Set the linear vibration motor to the around 100g (include the linear vibration motor) weight of block. Drop Height: 150cm Floor Material: Steel Direction: ±X, ±Y, ±Z; total 6 faces Number of Times: Each face 3 Times
Load Test	100,000 operating cycles in room temperature Cycle test condition: 2 seconds on, 1 second off

Acceleration, resistance, rated current and noise must meet specifications after allowing the device to rest for 4 hours after test.

Dimensions (Red wire is positive)



Specifications Revisions

Revision	Description	Date
A	Released from Engineering	4/6/2021
B	Updated Lead Wire Length	7/12/2022

Note:

1. Unless otherwise specified:
 - A. All dimensions are in millimeters.
 - B. Default tolerances are $\pm 0.5\text{mm}$ and angles are $\pm 3^\circ$.
2. Specifications subject to change or withdrawal without notice.



ANEXO 3

PUIaudio



Data Sheet

HD-EM0802-LW30-R

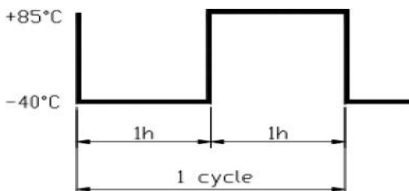
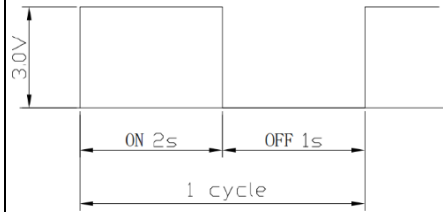
Features:

- Designed to be driven with 3 VDC
- Brushless motor with a built-in IC driver
- High Reliability
- Lead wires attached for easy assembly
- Generates 13,000 RPMs with no more than 50 dBA acoustic noise

Specifications

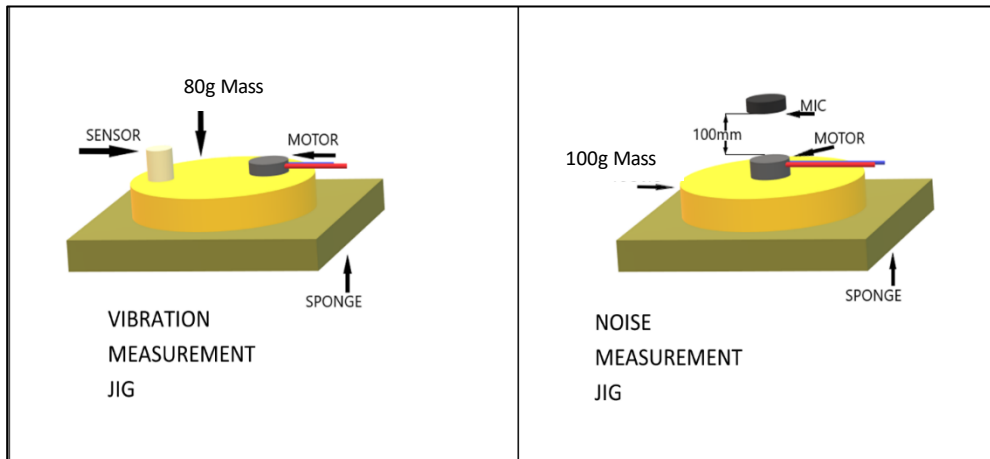
Parameters	Values	Units
Rated Voltage	3.0	V _{DC}
Starting Voltage	≤ 2.5	V _{DC}
Operating Voltage Range	2.7 ~ 3.3	V _{DC}
Direction of Rotation	CW	-
Vibration Strength	0.4	G
Insulation Resistance	> 1M	Ohms
Rated Speed	13,000 ± 3,000	RPM (@ 3 V _{DC})
Rated Current (Max)	80	mA (@ 3 V _{DC})
Starting Current (Max)	175	mA (@ 3 V _{DC})
Mechanical Noise (Max)	50	dBA
Environmental Compliances	RoHS/REACH	-
Actuator Mass	0.33	Grams
Storage Temperature	-30 ~ +70	°C
Operating Temperature	-20 ~ +60	°C

Reliability Testing

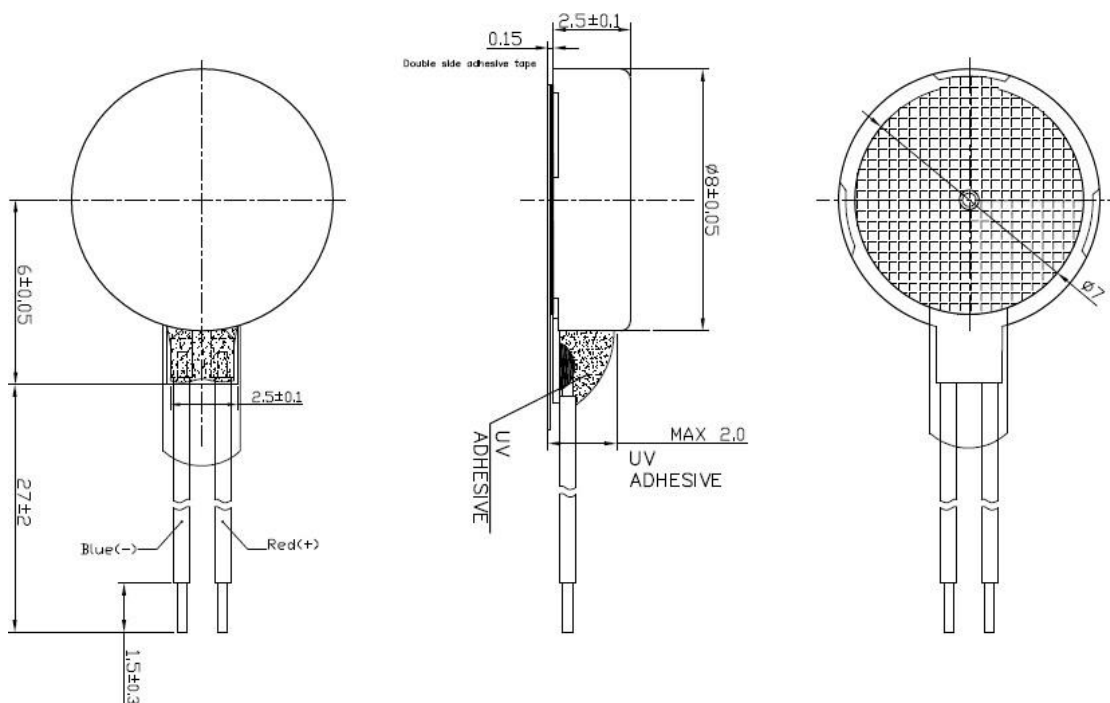
Type of Test	Test Specifications	Pass/Fail Criteria
High Temperature	Temperature: $+70 \pm 2^{\circ}\text{C}$ Test Duration: 96 hours	<i>After each test, motors rest for 4 hours at room temperature. After rest, the motor shall meet the following initial values:</i> 1) Rated Speed: $\pm 30\%$ initial 2) Rated Current: $\pm 30\%$ initial 3) Starting Voltage: $2.5 \text{ V}_{\text{DC}}$
Low Temperature	Temperature: $-20 \pm 2^{\circ}\text{C}$ Test Duration: 96 hours	
High Humidity	Temperature: $+40 \pm 2^{\circ}\text{C}$ Humidity: 90 ~ 95 RH Test Duration: 96 hours	
Thermal Shock	 Test Cycles: 32	
Free Drop	Drop from listed drop height three times in $\pm X$, $\pm Y$, $\pm Z$ directions Jig Mass: 100 grams (including motor) Drop Height: 1.5 meters Test Cycles: 3 per drop axis	<i>After test, motors rest for 4 hours at room temperature. After rest, the motor shall meet the following initial values:</i> 1) Rated Speed: $-20\%/+40\%$ initial 2) Rated Current: $\pm 30\%$ initial 3) Starting Voltage: $2.5 \text{ V}_{\text{DC}}$
Vibration	Acceleration: 22m/s^2 Frequency: 10 ~ 55 Hz Cycle Time: 10 min log sweep (1 G) Cycle: 10 Hz ~ 55 Hz ~ 10 Hz Test Orientation: X, Y, Z Test Duration: 10 minutes	
Lifetime	 Test Cycles: 500,000 (@ 3V_{DC})	

After testing, each motor must achieve the listed Pass/Fail Criteria for the appropriate test.

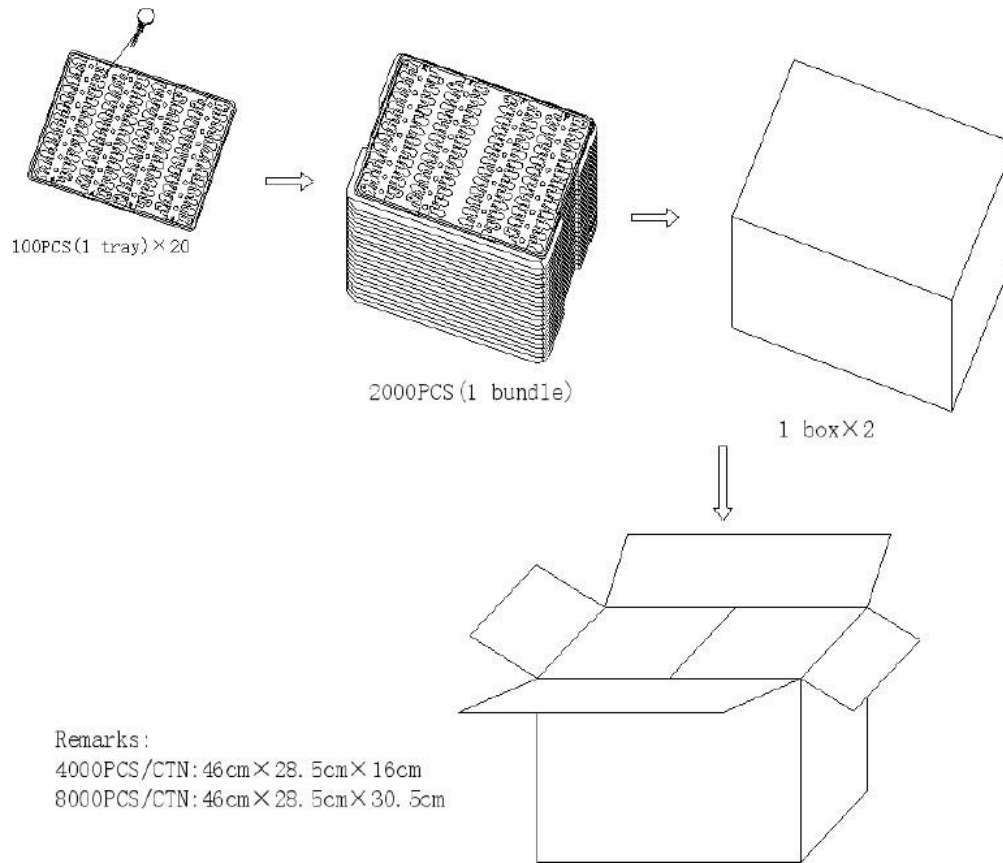
Measurement Methods



Dimensions



Packaging



Specifications Revisions

Revision	Description	Date
A	RELEASED FROM ENGINEERING	2/24/2023

Note:

- Unless otherwise specified:
 - All dimensions are in millimeters.
 - Default tolerances are $\pm 0.5\text{mm}$ and angles are $\pm 3^\circ$.
- Specifications subject to change or withdrawal without notice.

7.6 Register Map

Table 3. Register Map Overview

REG NO.	DEFAULT	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
0x00	0xE0	DEVICE_ID[2:0]			Reserved	DIAG_RESULT	Reserved	OVER_TEMP	OC_DETECT
0x01	0x40	DEV_RESET	STANDBY	MODE[2:0]					
0x02	0x00	RTP_INPUT[7:0]							
0x03	0x01	Reserved			HI_Z	Reserved	LIBRARY_SEL[2]	LIBRARY_SEL[1]	LIBRARY_SEL[0]
0x04	0x01	WAIT1	Reserved			WAV_FRM_SEQ[16:0]			
0x05	0x00	WAIT2	Reserved			WAV_FRM_SEQ[16:0]			
0x06	0x00	WAIT3	Reserved			WAV_FRM_SEQ[16:0]			
0x07	0x00	WAIT4	Reserved			WAV_FRM_SEQ[16:0]			
0x08	0x00	WAIT5	Reserved			WAV_FRM_SEQ[16:0]			
0x09	0x00	WAIT6	Reserved			WAV_FRM_SEQ[16:0]			
0x0A	0x00	WAIT7	Reserved			WAV_FRM_SEQ[16:0]			
0x0B	0x00	WAIT8	Reserved			WAV_FRM_SEQ[16:0]			
0x0C	0x00	Reserved							
0x0D	0x00	Overdrive time				ODT[7:0]			
0x0E	0x00	Sustain positive time				SPT[7:0]			
0x0F	0x00	Sustain negative time				SNT[7:0]			
0x10	0x00	Brake time				BR[7:0]			
0x11	0x05	Reserved			ATH_MIN_INPUT[7:0]			ATH_FILTER[1:0]	
0x12	0x19	Reserved			ATH_MIN_INPUT[7:0]			ATH_FILTER[1:0]	
0x13	0xFF	Reserved			ATH_MAX_INPUT[7:0]			ATH_FILTER[1:0]	
0x14	0x19	Reserved			ATH_MIN_DRIVER[7:0]			ATH_FILTER[1:0]	
0x15	0xFF	Reserved			ATH_MAX_DRIVER[7:0]			ATH_FILTER[1:0]	
0x16	0x3E	Reserved			RATED_VOLTAGE[7:0]			ATH_FILTER[1:0]	
0x17	0x8C	Reserved			OD_CLAMP[7:0]			ATH_FILTER[1:0]	
0x18	0x0C	Reserved			A_CAL_COMP[7:0]			ATH_FILTER[1:0]	
0x19	0x6C	Reserved			A_CAL_BEMF[7:0]			ATH_FILTER[1:0]	
0x1A	0x36	N_ERM_LRA	FB_BRAKE_FACTOR[2:0]		LOOP_GAIN[1:0]		BEMF_GAIN[1:0]		
0x1B	0x93	STARTUP_BOOST	Reserved	AC_COUPLE	DRIVE_TIME[4:0]		BEMF_GAIN[1:0]		
0x1C	0xF5	BIDR_INPUT	BRAKE_STABILIZER	SAMPLE_TIME[1:0]	BLANKING_TIME[1:0]		IDSS_TIME[1:0]		
0x1D	0xA0	NG_THRESH[1:0]		ERM_OPEN_LOOP	SUPPLY_COMP_DIS	DATA_FORMAT_RTP	LRA_DRIVE_MODE	N_PWM_ANALOG	LRA_OPEN_LOOP
0x1E	0x20	ZC_DET_TIME[1:0]		AUTO_CAL_TIME[1:0]	Reserved				
0x1F	0x80	AUTO_CL_CNT[1:0]		LRA_AUTO_OPEN_LOOP	PLAYBACK_INTERVAL	BLANKING_TIME[3:2]		IDSS_TIME[3:2]	
0x20	0x33	Reserved		OL_LRA_PERIOD[6:0]					
0x21	0x00	VBATT[7:0]							
0x22	0x00	LRA_PERIOD[7:0]							