UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

Diseño y realización de una micro bomba peristáltica para aplicaciones de sensado en tiempo real

Trabajo Fin de Grado

Grado en Ingeniería Electrónica Industrial y Automática

AUTOR/A: Sesma Jarauta, Jorge

Tutor/a: Ponce Alcántara, Salvador

Cotutor/a: García Rupérez, Jaime

CURSO ACADÉMICO: 2022/2023

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

DISEÑO Y REALIZACIÓN DE UNA MICRO BOMBA PERISTÁLTICA PARA APLICACIONES DE SENSADO EN TIEMPO REAL

Trabajo final de grado

Grado en Ingeniería Electrónica Industrial y Automática

Autor: Sesma Jarauta, Jorge

Tutor: Ponce Alcántara, Salvador

Cotutor: García Rupérez, Jaime

Curso académico: 2022/2023

Resumen

Una bomba peristáltica es un tipo de bomba de desplazamiento positivo que utiliza la acción de compresión y relajación de un tubo flexible para generar el flujo de líquido. Consiste en un rotor con rodillos que giran sobre el tubo, aplicando presión que impulsa el fluido de manera controlada y sin contaminación. Este diseño hace que la bomba peristáltica sea adecuada para aplicaciones que requieren un flujo preciso, continuo, y sin contacto directo con los componentes internos de la bomba.

Los laboratorios del Centro de Tecnología Nanofotónica de Valencia disponen de diversos modelos de bombas utilizadas para fluir distintas disoluciones sobre sensores fotónicos, y en tiempo real. A pesar de que las bombas presentes en los laboratorios ofrecen una alta precisión y repetibilidad, su elevado coste y tiempos de espera en la reparación representan desventajas considerables.

En el presente proyecto se plantea el diseño e implementación de una micro bomba peristáltica de bajo coste, fácil operación, con reducido tamaño y consumo eléctrico, con el fin de ser utilizada en setups de medida portables. Además, al ser diseñada y realizada en el Centro de Tecnología, su reparación será más rápida y sencilla en caso de ser necesaria. Este trabajo aborda el diseño integral de la bomba, desde la selección del motor y el diseño de las partes mecánicas, hasta el sistema de control y la programación de la interfaz.

Se expondrá detalladamente la elección de cada uno de los componentes, tales como el motor, el controlador, el uso de encoders, la pantalla LCD y los rodillos del cabezal. Además, se describirá la integración final con el sensor y la puesta en funcionamiento.

Finalmente, se presentarán y analizarán los resultados obtenidos de las pruebas experimentales, y se llevará a cabo una evaluación técnico-económica de la micro bomba peristáltica para aplicaciones prácticas en contextos reales.

Palabras clave

Bomba peristáltica, sensor fotónico, motor paso a paso, fluídica, velocidad de muestreo, encoder, Arduino.

Abstract

A peristaltic pump is a type of positive displacement pump that uses the compression and relaxation action of a flexible tube to generate fluid flow. It consists of a rotor with rollers that rotate on the tube, applying pressure that drives the fluid in a controlled and contamination-free manner. This design makes the peristaltic pump suitable for applications requiring precise, continuous flow without direct contact with the pump's internal components.

The laboratories of the Nanophotonics Technology Centre in Valencia have several models of pumps used to flow different solutions over photonic sensors, and in real time. Although the pumps present in the laboratories offer high precision and repeatability, their high cost and repair waiting times represent considerable disadvantages.

This project proposes the design and implementation of a low-cost, easy-to-operate peristaltic micro pump, with reduced size and power consumption, in order to be used in portable measurement setups. Furthermore, as it is designed and built at the Technology Centre, its repair will be quicker and simpler if necessary. This work deals with the integral design of the pump, from the selection of the motor and the design of the mechanical parts to the control system and the programming of the interface.

The choice of the individual components, such as the motor, the controller, the use of encoders, the LCD display and the head rollers, will be discussed in detail. Furthermore, the final integration with the sensor and the commissioning will be described.

Finally, the results obtained from the experimental tests will be presented and analysed, and a technical-economic evaluation of the peristaltic micro pump for practical applications in real contexts will be carried out.

Keywords

Peristaltic pump, photonic sensor, stepper motor, fluidics, sampling rate, encoder, Arduino.

Resum

Una bomba peristàltica és un tipus de bomba de desplaçament positiu que utilitza l'acció de compressió i relaxació d'un tub flexible per generar flux de fluid. Consisteix en un rotor amb rodets que giren sobre el tub, aplicant pressió que impulsa el fluid de manera controlada i sense contaminació. Aquest disseny fa que la bomba peristàltica sigua adequada per a aplicacions que requereixen un flux precís, continu, i sense contacte directe amb els components interns de la bomba.

Els laboratoris del Centre Tecnològic de Nanofotònica de València disposen de diversos models de bombes que permeten fluir diferents solucions per sensors, i en temps real. Tot i que les bombes presents als laboratoris ofereixen una gran precisió i repetibilitat, el seu elevat cost i temps d'espera de reparació representen desavantatges considerables.

Aquest projecte proposa el disseny i la implementació d'una microbomba peristàltica de baix cost i fàcil d'operar amb el fi de ser utilitzada en dispositius de mesura portables. A més, com que està dissenyat i construït al Centre Tecnològic, serà més ràpid i fàcil de reparar si cal. Aquest treball tracta del disseny integral de la bomba, des de la selecció del motor i el disseny de les peces mecàniques fins al sistema de control i la programació de la interfície.

L'elecció dels components individuals, com ara el motor, el controlador, el codificador rotatiu, la pantalla LCD i els rodets de capçal, es discutirà en detall. A més, es descriurà la integració final amb el sensor i la posada en marxa.

Finalment, es presentaran i analitzaran els resultats obtinguts de les proves experimentals i es realitzarà una avaluació tècnicoeconòmica de la micro bomba peristàltica per a aplicacions pràctiques en contextos reals.

Paraules clau

Bomba peristàltica, sensor fotònic, motor pas a pas, flux, velocitat de mostreig, encoder, Arduino.

Gracias a mis padres y a mi hermana Laura por su apoyo incondicional.

A David, Antonio y Alain por acompañarme hasta aquí.

Y mi más sincero agradecimiento a Salva, tutor de este TFG.

CONTENIDO

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

DISEÑO Y REALIZACIÓN DE UNA MICRO BOMBA PERISTÁLTICA PARA APLICACIONES DE SENSADO EN TIEMPO REAL

DOCUMENTO Nº1: MEMORIA TÉCNICA

Trabajo final de grado

Grado en Ingeniería Electrónica Industrial y Automática

Autor: Sesma Jarauta, Jorge

Tutor: Ponce Alcántara, Salvador

Cotutor: García Rupérez, Jaime

Curso académico: 2022

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO 1. OBJETO

El presente Trabajo de Fin de Grado tiene como objeto el diseño e implementación de una micro bomba peristáltica de bajo coste y fácil operación, que satisfaga las necesidades de los laboratorios del Centro de Tecnología Nanofotónica de Valencia en aplicaciones de sensado en tiempo real. En particular, la bomba será utilizada para un sistema de caracterización de sensores fotónicos portátil. Por ello requiere disponer de reducidas dimensiones y consumo eléctrico.

Los componentes escogidos están pensados en este sentido. La alimentación, tanto del microcontrolador como del motor, vendrá proporcionada por el propio dispositivo en el que se integrará la micro bomba, por lo que no se requerirá de una fuente adicional de alimentación. Esto resulta en un ahorro de espacio y una reducción de costes adicionales.

Por otra parte, se ha diseñado una interfaz sencilla y fácil de operar. Está compuesta por una pequeña pantalla LCD de dos líneas y 16 caracteres, en la que se verán los parámetros de configuración posibles tales como la velocidad de flujo deseada, el sentido de bombeo, o el diámetro del canal por el que se fluyen las disoluciones. La selección de parámetros se realizará mediante un encoder rotatorio y un botón de inicio y parada.

Además, el proyecto emplea herramientas de software y hardware libre, lo que permite que sea accesible a cualquier usuario interesado en utilizarlo, modificarlo o adaptarlo según sus necesidades específicas.

Por último, este trabajo se alinea con varios Objetivos de Desarrollo Sostenible (ODS) de la ONU. El ODS 9 (Industria, innovación e infraestructura) se relaciona con el desarrollo de tecnología e innovación en la fabricación de dispositivos microfluídicos. El ODS 3 (Salud y bienestar) se vincula al monitoreo de parámetros de salud y la administración controlada de medicamentos. Además, alinea con el ODS 4 (Educación de calidad) al promover la investigación, el desarrollo de tecnologías innovadoras y las posibles aplicaciones en la formación de profesionales en salud y ciencia.

Estos vínculos evidencian el impacto multidimensional y sostenible de la investigación propuesta.

CAPÍTULO 2. SITUACIÓN DE PARTIDA Y ESTUDIO DE NECESIDADES

2.1 Situación de partida

El empleo de micro bombas peristálticas es habitual en diversos campos de investigación, especialmente en aquellos vinculados a aplicaciones biomédicas y químicas [1]. Pese a que estos dispositivos suelen ser de alta calidad, su elevado coste de entre 1000 y 3500 euros [2], y su plazo de entrega (que puede superar los 30 días), imponen restricciones considerables en términos de su empleo y accesibilidad. Además, problemas menores pueden resultar en la inoperatividad completa, lo que implica costes adicionales en reparaciones y, primordialmente, posibles retrasos en un proyecto de investigación.

En el Centro de Tecnología Nanofotónica de Valencia se están llevando a cabo medidas de sensado empleando sensores fotónicos. Actualmente los laboratorios emplean dos clases de bombas para hacer fluir la disolución deseada por el sensor: bombas peristálticas y bombas de jeringa. Dichas bombas pueden apreciarse en las Figuras 1 y 2.

Las bombas de jeringa tienen dos inconvenientes notables respecto a las peristálticas: la imposibilidad de miniaturizarlas, y un volumen de bombeo limitado al tamaño de la jeringa. Por tanto, este tipo de bomba quedó descartada como opción válida para este proyecto en particular.

Por consiguiente, este TFG tiene como finalidad el diseño y realización de una bomba peristáltica de gran precisión, que sea capaz de fluir líquidos de manera continua, y con flujos reducidos, del orden de microlitros por minuto (uL/min). Además, ha de ser de bajo coste y consumo eléctrico, y tener dimensiones reducidas.

Figura 1. Bomba de jeringa SyringePumpPro

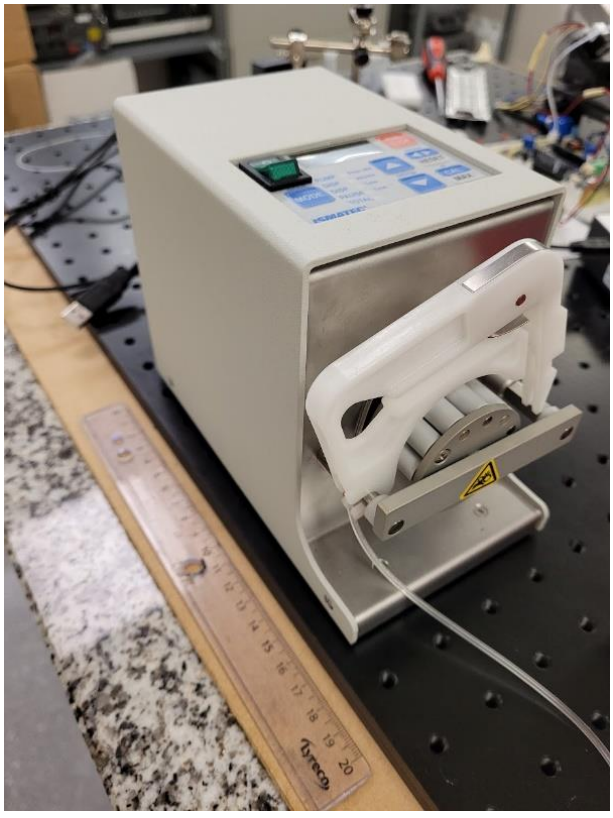


Figura 2. Bomba peristáltica ISMATEC

2.2 Requerimientos

Para llevar a cabo el proyecto de manera adecuada, es necesario establecer unos requisitos iniciales que deben cumplirse con el objetivo de ofrecer una herramienta completamente funcional y fácil de usar para el usuario. Se propone evitar soluciones que requieran una manipulación excesiva por parte del operador (cableado, configuración inicial, etc.)

Además, el sistema está concebido para su integración en un sistema más amplio, que es el setup de medida de sensores fotónicos portable. Por ello, su tamaño e interoperabilidad con dicho sistema serán cruciales.

A continuación, se detallan los requerimientos a cumplir:

- Capacidad de bombeo: entre 10 uL/min y 120 uL/min.
- Tamaño reducido. El equipo en el que se va a integrar es portable, por tanto, este aspecto es clave.
- Bajo consumo de electricidad. El equipo de medición funciona con baterías de 12 V. Es importante que el consumo de la bomba sea bajo para asegurar su autonomía. Este será un aspecto primordial a la hora de escoger los componentes.
- Bajo coste. La idea es ofrecer una alternativa viable con un costo que represente sólo una fracción del precio de las bombas existentes en el mercado.
- Diseño simple. Facilitará futuras modificaciones tanto del hardware como del software, así como eventuales reparaciones.
- Manejo intuitivo. Cualquier usuario podrá utilizar la micro bomba y modificar sus parámetros de ajuste sin necesidad de ningún conocimiento previo sobre ella.
- Alta continuidad en el flujo, de cara a realizar medidas de sensado fluyendo en tiempo real y de forma estable.

CAPÍTULO 3. DESCRIPCIÓN DETALLADA DE LA SOLUCIÓN ADOPTADA

3.1 Visión general del sistema

La bomba diseñada en este proyecto se basa en un motor paso a paso que proporciona un control preciso y una alta resolución en la velocidad de rotación. El motor es controlado por un *driver* adecuado para asegurar un rendimiento óptimo y eficiente. La lógica y el control del sistema se llevan a cabo mediante una placa de Arduino Nano. Se trata de una plataforma de hardware libre y fácilmente programable, lo que permite una gran flexibilidad en el desarrollo y adaptación del sistema.

Para lograr un control preciso de la velocidad del motor, se emplea además un encoder óptico de 1000 ciclos por revolución (CPR), que permite un monitoreo preciso y en tiempo real de la posición y velocidad del motor. Esta información es utilizada por el sistema de control para ajustar y mantener la velocidad de la bomba de acuerdo con los requerimientos de la aplicación.

El sistema cuenta con una pantalla LCD que proporciona una interfaz gráfica de usuario para visualizar y navegar por un menú de opciones. La navegación y la interacción con el menú se realiza mediante un encoder rotativo y un pulsador, lo que permite el ajuste de parámetros y configuración del sistema.

El cabezal de la bomba, componente crítico en funcionamiento de una bomba peristáltica, ha sido diseñado específicamente para este proyecto, y se ha fabricado mediante impresión 3D. Esto permite un diseño personalizado y adaptable, así como una reducción en los costes de producción.

3.2 Dimensionamiento

El principio de funcionamiento de una bomba peristáltica se basa en la compresión y relajación alternada de un tubo flexible, que genera un movimiento del fluido contenido en su interior. Este proceso permite el desplazamiento del fluido en una dirección determinada.

El mecanismo de compresión peristáltica es accionado por un motor y se compone de un conjunto de rodillos montados sobre un cabezal rotativo. A medida que el cabezal gira, los rodillos comprimen el tubo flexible contra una superficie fija, creando una zona de baja presión tras cada rodillo y una zona de alta presión delante de él. Este gradiente de presión genera un flujo unidireccional del fluido a través del tubo, cuyo caudal y velocidad de bombeo pueden ajustarse controlando la velocidad del motor y el diámetro del tubo[3].

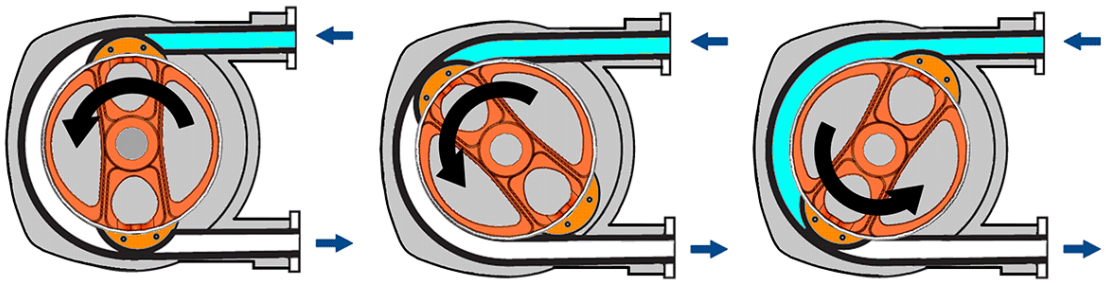


Figura 3.Principio de funcionamiento de una bomba [4]

Por tanto, el volumen total desplazado por la bomba o flujo (Q) es el resultado del producto del volumen atrapado entre los rodillos (V_T) por la velocidad de giro (n) [3]. En el diseño abordado en este trabajo, el número de rodillos es 8.

Esto se expresa:

$$Q = V_T \times n$$

Siendo V_T:

$$V_T = \pi r^2 \times L \times 8$$

Donde r es el radio interno de tubo, en este caso 0,255 mm, y L la longitud entre dos puntos de presión de los rodillos, en este caso 23 mm (Ver plano 2).

De acuerdo con las ecuaciones anteriores, la tabla 1 muestra la velocidad de rotación para cada valor de flujo deseado.

vel. flujo(uL/min)	vel. Rotacion (rpm)
10	0,266
15	0,399
20	0,532
25	0,665
30	0,798
35	0,931
40	1,064
45	1,197
50	1,330
55	1,463
60	1,596
65	1,729
70	1,862
75	1,995
80	2,128
85	2,261
90	2,394
95	2,527
100	2,660
105	2,793
110	2,926
115	3,059
120	3,193

Tabla 1.Velocidad de flujo y de rotación para un canal con un radio interno de 0,255 mm

Viendo esta información de manera gráfica, se aprecia claramente la proporcionalidad directa que existe entre la velocidad de rotación y de bombeo.

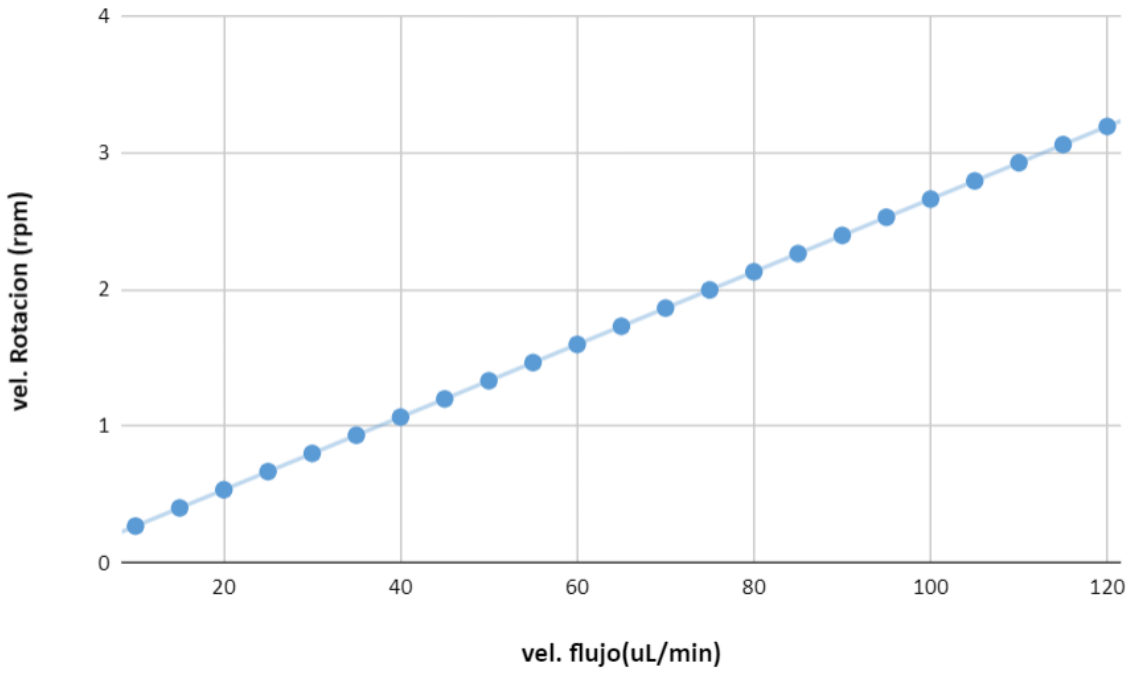


Figura 4.Velocidad de rotación frente a velocidad de flujo

El flujo real siempre será menor al teórico ya que existen pérdidas por diversos motivos, como pueden ser las pérdidas por cavitación en el tubo, o por rozamiento. La relación entre el flujo real y el teórico se conoce como eficiencia volumétrica y suele situarse entre el 92% y el 99%[5]. Solo se podrá conocer la eficiencia volumétrica de esta bomba una vez esté en funcionamiento y se puedan hacer pruebas de rendimiento.

En este punto cabe destacar que, aunque la exactitud en la velocidad de flujo es importante, lo que tiene más relevancia es que sea repetitiva, y que el flujo sea completamente constante y continuo.

3.3 Elección del motor

Dentro de los motores rotativos destacan dos tipos principales para esta clase de aplicaciones: los motores de corriente continua y los motores paso a paso. Cada uno presenta características propias que supondrán ventajas o inconvenientes para esta aplicación en concreto.

Motores de corriente continua (CC)

Los motores de corriente continua son uno de los actuadores electromecánicos más comunes, y existe una gran variedad en cuanto a potencia, voltaje y tamaño. Estos motores tienen altas velocidades de giro y un par relativamente bajo [6]. Algunos cuentan con un reductor interno de engranajes para aumentar el par del motor y reducir su velocidad. Sin embargo, estos motores tienen un mal control de la velocidad y nulo control de posicionamiento, lo que suele traducirse en un comportamiento no lineal. Además, dependen mucho de la carga que soportan.

Motores paso a paso

Los motores paso a paso (también conocidos como *steppers*), son muy comunes especialmente en aplicaciones de robótica. A diferencia de los motores de corriente continua, los motores *stepper* tienen un control de posición y velocidad total. Presentan un mayor par a velocidades bajas, y a menudo incorporan también cajas reductoras para disminuir aún más la velocidad [6]. Sin embargo, debido a la naturaleza paso a paso de estos motores, pueden presentar problemas de vibración [6]. La tabla 2 presenta los principales puntos a favor y en contra de los motores comentados.

	Características		**Control**	
	Velocidad	**Par**	**Posición**	**Velocidad**
Motor CC	▲Alto	▼Bajo	▼Malo	▼Malo
Motor CC con reducciones	– Medio	▲Alto	▼Malo	▼Malo
Stepper	- Medio	- Medio	▲Absoluto	▲Absoluto

Tabla 2. Características de los distintos motores [6]

Considerando los requerimientos fundamentales de este proyecto, que incluyen una baja velocidad de flujo (o de rotación) y un control preciso, se ha seleccionado el motor paso a paso como la opción más adecuada para satisfacer estas condiciones.

Como se ha visto anteriormente en el apartado de dimensionamiento, las velocidades de rotación requeridas son realmente bajas, y a velocidades bajas los motores stepper pueden perder par. Por ello, se ha optado por un motor paso a paso con reducciones mecánicas, caracterizado por conseguir esas bajas velocidades a la vez que se mantiene un par elevado [6].

Para hallar el par o la carga que ha de soportar el motor, se ha realizado una prueba experimental con la bomba peristáltica presente en el laboratorio, y mostrada en la Figura 2. La prueba consistió en colgar una masa conocida de manera perpendicular al eje del motor, e ir aumentando esta masa con el fin de hallar la fuerza máxima con la que el motor se mueve de manera continuada. La masa con la que se alcanzó el límite del motor fue de 5 Kg. Multiplicando esta masa por la aceleración de la gravedad, se obtiene la carga radial máxima del motor.

$$F_R = m \times g = 5 \times 9{,}8 = 49\ N$$

Por tanto, el motor ideal para esta bomba es un motor paso a paso con caja reductora y una carga radial máxima mayor a 49 N, un tamaño reducido (preferiblemente del calibre NEMA 17 o inferior), y un consumo de energía bajo. El estándar NEMA define una serie de tamaños estándar, que van desde NEMA 8 hasta el NEMA 42. Cada calibre NEMA tiene dimensiones específicas para la carcasa de motor, el diámetro del eje y los patrones de montaje.

Después de una exhaustiva búsqueda se opta por el motor NEMA 14 Bipolar 14HS13-0804S-PG5[7]. Sus características son:

- Ángulo de paso: 1,8 grados
- Corriente nominal/fase: 1 A
- Relación reductora: 5,18:1
- Carga radial máxima del eje: 100 N
- Tamaño: 35x35x81 mm
- Tiene un paso de 1,8 grados (200 pasos por vuelta). El reductor interno tiene una relación de 1/5. Combinados, un total de 1036 pasos por vuelta, equivalente a un paso de solo 0,35 grados.

3.4 Control del motor

3.4.1 Driver

Los controladores, o *drivers*, son fundamentales en el funcionamiento de los motores paso a paso. Proporcionan la secuencia específica de activación de las bobinas del motor, regulan la corriente necesaria para su funcionamiento, permiten técnicas de micro paso para un movimiento más silencioso y suave, y ofrecen protecciones como limitación de corriente, sobretensión y sobrecalentamiento. En esencia, los drivers sirven como una interfaz vital entre el sistema de control de bajo voltaje y el motor de alta potencia, garantizando un control seguro y sencillo.

Los controladores A4988 y DRV8825 son dos opciones populares para controlar motores paso a paso en proyectos de electrónica y robótica. Aunque ambos son capaces de manejar motores paso a paso de forma eficiente, existen algunas diferencias clave entre ellos.

A4988

- Puede proporcionar hasta 2 A por bobina.
- Soporta reducciones de hasta 1/16 pasos (lo que permite un movimiento más suave en comparación con los pasos completos).
- Voltaje de operación de hasta 35V.
- Dimensión 15 x 20 mm

DRV8825

- Puede proporcionar hasta 2,5 A por bobina (aunque necesita un disipador de calor para evitar el sobrecalentamiento a esta corriente).
- Soporta hasta 1/32 pasos, lo que proporciona un movimiento aún más suave que el A4988.
- Voltaje de operación de hasta 45V.
- Dimensiones: 15,5 x 20,5 mm

Todos los detalles se pueden encontrar en el Anexo II.

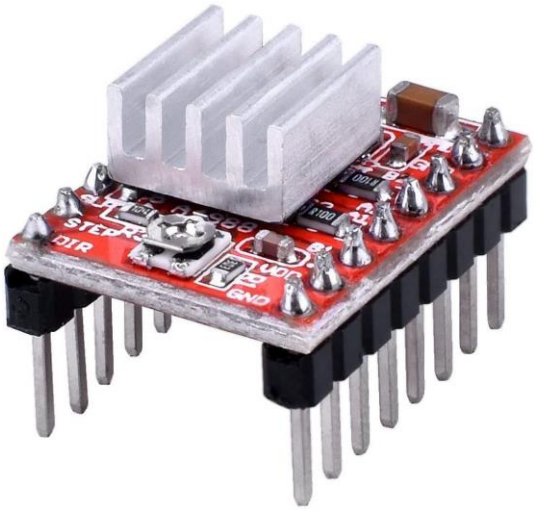


Figura 5. Driver A4988 con disipador

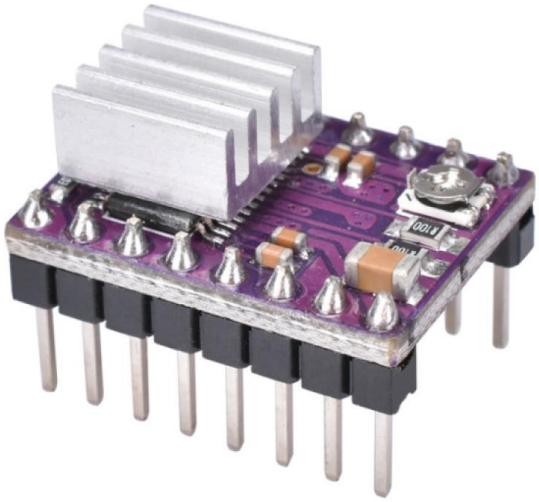


Figura 6. Driver DRV8825 con disipador

En resumen, aunque ambos drivers son capaces de controlar motores paso a paso de manera eficiente, se escoge el DRV8825 ya que presenta unas características técnicas superiores y el precio, aunque algo superior, sigue siendo muy competitivo [8]. El precio de estos dispositivos oscila entre los 80 céntimos y los 3 euros [9].

3.4.2 Encoder

Si bien es cierto que los motores paso a paso pueden funcionar correctamente sin necesidad de una retroalimentación, en este caso se ha optado por añadir un encoder para asegurar un funcionamiento óptimo y absolutamente repetitivo. Como se viene diciendo a lo largo de este trabajo, la precisión y, sobre todo, la continuidad del flujo es vital para el correcto funcionamiento de la medida de sensado, por lo que añadir un encoder proporcionará más robustez y fiabilidad al diseño y desempeño de la bomba.

El principio de funcionamiento de la retroalimentación es que el encoder mida la velocidad de giro del motor. A continuación, el programa la comparará con la velocidad que debería llevar y, en el caso de que haya una diferencia, ajustará la velocidad del motor. Esto se explicará de manera más exhaustiva en el apartado 3.6, dedicado al programa.

Se ha optado por el uso de un encoder óptico, ya que los encoders magnéticos proporcionan menor resolución y se pueden ver afectados por el campo magnético del motor.

Un codificador óptico funciona emitiendo un haz de luz a través de un disco codificador con ranuras transparentes y opacas. A medida que este disco o cinta se mueve, las ranuras intercalan la luz hacia un fotodetector, generando pulsos de luz. Estos pulsos se convierten en señales eléctricas que se interpretan para determinar la posición, velocidad y dirección del objeto en movimiento.

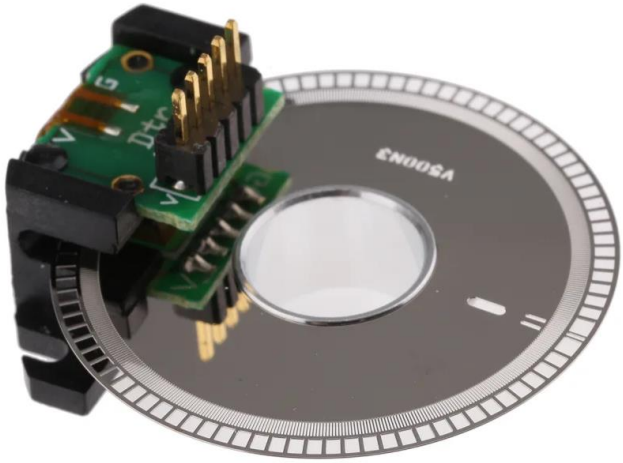


Figura 7.Codificador óptico Broadcom 5V [10]

Los codificadores ópticos pueden ser absolutos o incrementales. Los codificadores absolutos proporcionan una posición única para cada posición del disco codificador, mientras que los codificadores incrementales proporcionan una cantidad de pulsos por revolución o por distancia recorrida, y se utilizan para determinar cuánto ha girado o se ha movido el objeto desde la última vez que se verificó.

La elección de un disco de codificador con más o menos cuentas por revolución (CPR) depende del nivel de precisión y resolución que se requiera en la aplicación específica. En este caso, al tratarse de una aplicación de gran precisión, se escoge un disco de 1000 CPR. De esta manera la relación entre pasos del encoder y pasos del motor será prácticamente de 1 a 1 lo que proporciona una lectura absoluta.

Finalmente, el modelo escogido ha sido el codificador óptico incremental AEDB-9140 de 1000 CPR [10] cuyo precio ronda los 40 euros. Sus características aparecen de una forma más extensa en el Anexo II.

3.5 Interfaz de usuario

El principal requisito a la hora de implementar una interfaz de usuario es que fuese sencilla y muy intuitiva, es decir, que cualquier persona fuese capaz de manejar la bomba, aunque no hubiese trabajado con ella anteriormente, y sin necesidad de ningún manual de instrucciones.

La interfaz de usuario está compuesta por una pantalla LCD 16x2 para visualizar y navegar por un menú de opciones. La navegación y la interacción con el menú se realiza mediante un encoder rotatorio y un pulsador, lo que permite el ajuste de parámetros y configuración del sistema. El menú de opciones muestra los parámetros a modificar que son: sentido de giro de la bomba, velocidad de flujo, y el diámetro del tubo. Adicionalmente, una cuarta opción permitirá ver los ajustes de la configuración.

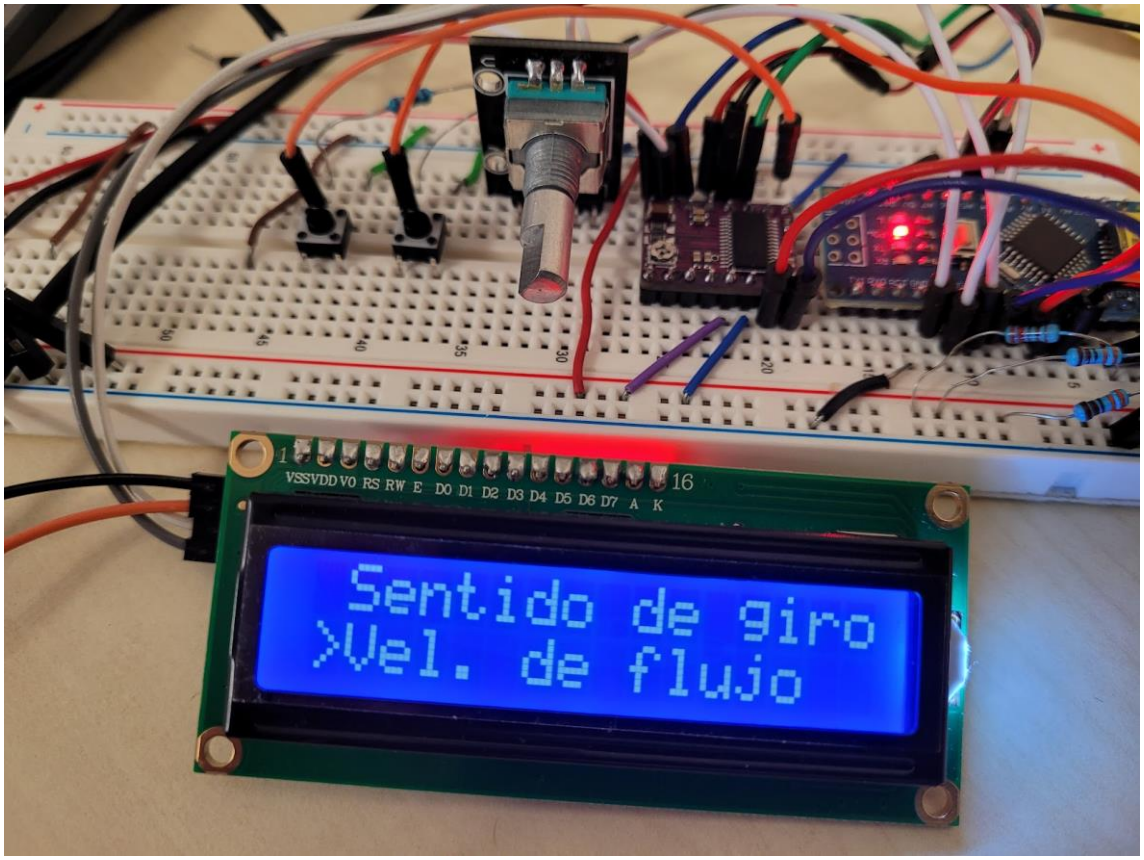


Figura 8. Montaje de la interfaz

Aunque tanto el sentido de giro de la bomba como el diámetro del tubo se puedan configurar, vendrán predeterminados por el sistema ya que son variables que presumiblemente no cambiarán, o por lo menos no habitualmente. De esta manera, el usuario solo tendrá que introducir la velocidad de flujo deseada y pulsar el botón de *start/stop* para que la bomba se ponga en funcionamiento de forma inmediata. Una vez en funcionamiento, la pantalla mostrará un mensaje indicando que está fluyendo y con qué flujo. A través del encoder rotativo se podrá aumentar o disminuir la velocidad de flujo en tiempo real, sin necesidad de parar la bomba ni de volver al menú principal. Cuando se desee parar la bomba, únicamente habrá que pulsar sobre el botón *start/stop*. Adicionalmente, se incluye un botón para poner a funcionar la bomba a máxima velocidad, con el fin de poder vaciar rápidamente el circuito hidráulico al finalizar la medida de sensado.

Todo esto viene gobernado por la placa Arduino Nano basada en el controlador ATmega328P. Esta placa es muy compacta, de apenas 43x15 mm, y cuenta con el número de entradas y salidas necesarias para este proyecto.

3.6 Programa

El programa se encarga de manejar el menú, las opciones seleccionadas, la interacción con el encoder y botones, el movimiento del motor y del funcionamiento del codificador óptico.

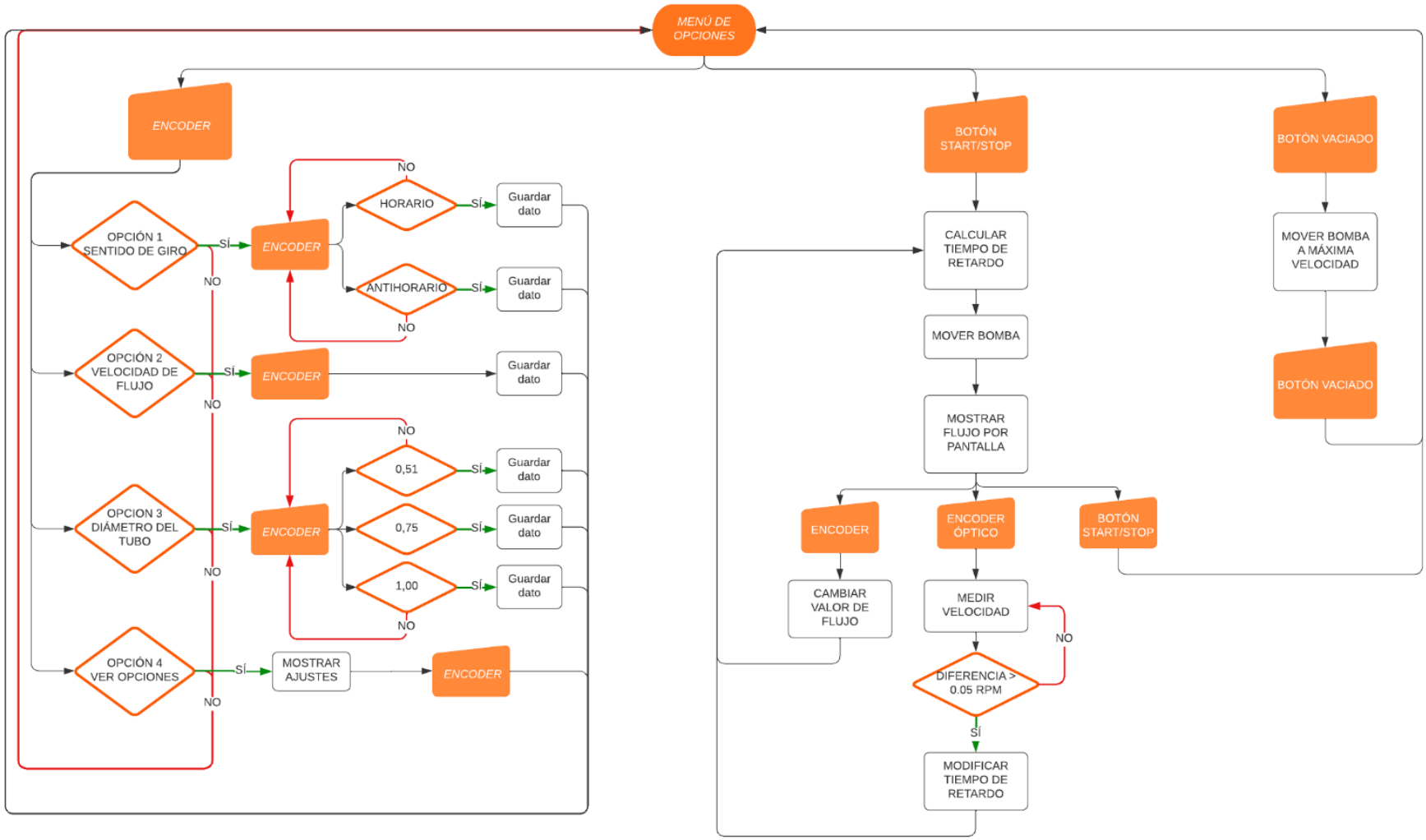


Figura 9. Diagrama de flujo del funcionamiento general del programa

Para la programación de un Arduino, el lenguaje convencionalmente utilizado es C++.

En este programa se incluyen además dos librerías, <Encoder.h> para el manejo del encoder y <LiquidCrystal_I2C.h> para la pantalla LCD.

```
1   #include <LiquidCrystal_I2C.h>
2   #include <Encoder.h>
```

Figura 10. Librerías utilizadas

El primer paso es definir los pines utilizados por el display LCD, los botones, los encoders y el driver del motor.

```
6   // Inicializacion display LCD
7   LiquidCrystal_I2C lcd(0x27,16,2);
8
9   // establecemos el botón start/stop
10  const int buttonStart=7;
11  // establecemos el botón vaciado
12  const int buttonVacio=8;
13
14  // Pines del encoder rotativo y botón
15  const int encoderPinA = 3;
16  const int encoderPinB =4;
17  const int buttonPin = 2;
18
19  //Pines encoder optico
20   int codificadorA=11;
21   int codificadorB=12;
22
23  //variables cuentas del encoder
24  int counter;
25  int aLast;
26  int aState;
27  int bState;

37   //pines stepper driver
38   int dirPin=5;
39   int stepPin=6;
```

Figura 11. Definición de pines

El menú de configuración consta de cuatro opciones que también se definen:

```
// Menú
const char *menuItems[] = {
    "Sentido de giro",
    "Vel. de flujo",
    "Diametro tubo",
    "Ver ajustes"
};
```

Figura 12. Menú de configuración

Cada elemento del menú se representa como una cadena de caracteres.

El menú está compuesto por los siguientes elementos:

1. “Sentido de giro”: Esta opción permite al usuario seleccionar el sentido de giro del motor, ya sea horario o antihorario.
2. “Vel.flujo”: Esta opción permite al usuario ajustar la velocidad de flujo deseada en unidades de microlitros por minuto (uL/min).
3. “Diametro tubo”: Esta opción permite al usuario seleccionar el diámetro del tubo entre tres opciones: 0.51, 0,75 y 1,00 mm.
4. “Ver ajustes”: Esta opción muestra los ajustes del sistema seleccionados mediante las opciones anteriores.

Las opciones para sentido de giro también se definen como una cadena de caracteres que en este caso se almacenan en **submenuItems.*

```
// Submenú
const char *submenuItems[] = {
  "Horario",
  "Antihorario"
};
```

Figura 13. Submenú para el sentido de giro

Después, se encuentra la función *setup()* donde se configuran los pines como entradas o salidas, se inicializa la comunicación serie, el LCD, y se establecen los valores predeterminados para las variables referentes al sentido de giro y el diámetro de tubo.

```ino
void setup() {

  Serial.begin(9600);

//Inicializar pines
  pinMode(buttonPin, INPUT_PULLUP);
  pinMode(buttonStart, INPUT_PULLUP);
  pinMode(codificadorA, INPUT);
  pinMode(codificadorB, INPUT);
  pinMode(dirPin, OUTPUT);
  pinMode(stepPin, OUTPUT);

  aLast=digitalRead(codificadorA);

  lcd.init();
  lcd.backlight();
  lcd.setCursor(1,0);
  displayMenuItems(0);
  updateMenu(0);

  // Establecer los valores predeterminados de rotationDirection y dtubo
  rotationDirection = "horario";
  dtubo = 0.51;

}
```

Figura 14. Función Setup ()

Aunque tanto el sentido de giro como el diámetro del tubo se pueden variar en la configuración, se establecen esas opciones como predeterminadas ya que normalmente no variarán entre uso y uso de la bomba.

A continuación, aparece la función void *displayMenuitems()* para mostrar las opciones del menú por pantalla y la función void *updateMenu()* para actualizar la posición del cursor en el menú.

```ino
// Muestra las opciones del menú en la pantalla LCD
void displayMenuItems(int startIndex) {
  int baseIndex = startIndex > 0 ? startIndex - 1 : startIndex;

  for (int i = 0; i < 2; i++) {
    int currentItemIndex = baseIndex + i;
    if (currentItemIndex < 0 || currentItemIndex >= menuSize) {
      continue;
    }
    lcd.setCursor(1, i);
    lcd.print(menuItems[currentItemIndex]);
    lcd.print("                ");
  }
}

// Actualiza la posición del signo '>' en el menú
void updateMenu(int menuIndex) {
  int cursorPos = (menuIndex > 0) ? 1 : 0;

  if (menuIndex != lastMenuIndex && menuIndex < menuSize) {
    lcd.setCursor(0, lastMenuIndex % 2);
    lcd.print(" ");
    lcd.setCursor(0, cursorPos);
    lcd.print(">");
    displayMenuItems(menuIndex);
    lastMenuIndex = menuIndex;
  }
}

```

Figura 15.Funciones displayMenuitems() y updateMenu()

Cada opción del menú tiene su propia función llamada *void optionxFunction()* siendo x el número de cada opción. Es decir, la función que se ejecutará al pulsar sobre "Sentido de giro" es *void option1Function()* que contiene la lógica para seleccionar el sentido de giro. Al pulsar sobre el sentido de giro deseado, se almacenará en la variable previamente declarada como "rotationDirection" y se volverá automáticamente al menú principal. Al pulsar sobre "Vel. flujo" se llama a la función *void option2Function()* que contiene la lógica necesaria para escoger la velocidad de flujo deseada y almacenarla en la variable velFlujo. Al pulsar sobre "Diametro tubo" se llama a la función *void option3Function()* que mostrará otro submenú con las tres opciones de tubo disponible. Al pulsar sobre alguna de estas opciones, se almacenará el valor en la variable "dtubo" y se volverá al menú principal. Por último, al pulsar sobre "Ver ajustes" se ejecutará la función *void option4Function()* que mostrará los ajustes por pantalla hasta que el usuario vuelva a pulsar sobre el encoder. Todas estas funciones se pueden ver en detalle en el Anexo I.

Por último queda la función loop(), es decir, el bucle principal del programa. Esta es la función más extensa del programa y tiene varios cometidos. Por un lado contiene la lógica para navegar por el menú, detecta los cambios de posición del encoder rotatorio, y llama a las funciones *displayMenuitems()* y *updateMenu()* comentadas anteriormente a la vez que verifica si se pulsa sobre el comentado encoder.

```
  long newPosition = encoder.read() / 4;

  // Restringe el movimiento del encoder a un rango específico
  if (newPosition < 0) {
    newPosition = 0;
    encoder.write(newPosition * 4);
  } else if (newPosition > menuSize - 1) {
    newPosition = menuSize - 1;
    encoder.write(newPosition * 4);
  }

 // Actualiza el menú si la posición del encoder ha cambiado
  if (newPosition != encoderPosition) {
    encoderPosition = newPosition;
    displayMenuItems(encoderPosition);
    updateMenu(encoderPosition);
  }

   // Verifica si el botón está presionado
buttonPressed = digitalRead(buttonPin) == LOW;
```

Figura 16. Navegación en el bucle principal

Si se pulsa sobre el encoder, se entra en el bucle que contiene una estructura tipo *switch(),* encargada de llamar a la función correspondiente para cada opción.

```
if (!buttonPressed && lastButtonState == LOW) {
lastButtonState = HIGH;
unsigned long currentMillis = millis();

if (currentMillis - lastButtonPressTime > debounceDelay && !returningToMenu) {
  switch (lastMenuIndex) {
    case 0:
      option1Function();
      break;
    case 1:
      option2Function();
      break;
    case 2:
      option3Function();
      break;
    case 3:
      option4Function();
      break;
    default:
      break;
  }
}
}
 // Si se está volviendo al menú principal , esperamos a que se libere el botón
  if (returningToMenu && !buttonPressed) {
    returningToMenu = false;
  }
```

Figura 17. Estructura switch() para llamar a las distintas opciones

Dentro de la función loop() se encuentra también el movimiento del motor. Por un lado está el botón de vaciado que al ser pulsado entrará en un bucle tipo while(). En este bucle, la velocidad de flujo será máxima y de sentido opuesto al sentido de bombo escogido. La bomba parará cuando el usuario vuelva a presionar sobre el botón de vaciado.

```
398   if(digitalRead(buttonVacio)==HIGH&&lastButtonState1==LOW){ //si se pulsa el botón de vaciado
399
400     delay(1000); // esperamos un segundo para evitar detectar múltiples pulsaciones
401     vacio=true; //se activa el bucle de vaciado
402
403      while(vacio){
404
405       if(rotationDirection=="horario"){
406         digitalWrite(dirPin, LOW); //girar sentido antihorario
407       }else if(rotationDirection=="antihorario"){
408         digitalWrite(dirPin, HIGH);//girar sentido horario
409       }
410
411         digitalWrite(stepPin,HIGH);
412         delayMicroseconds(1044);// retardo para flujo de 120 uL/min
413         digitalWrite(stepPin,LOW);
414
415         lastButtonState1 = HIGH ;
416
417       if (digitalRead(buttonVacio) == HIGH && lastButtonState1 == HIGH) { // si se detecta otra pulsación
418         vacio = false; // desactivamos el bucle de vaciado
419           }
420       }
421   delay(300); //evitar rebotes
422 }
```

Figura 18. Bucle para el vaciado de la bomba

Por otra parte, está el bucle para el bombeo. En este bucle se entrará cuando se pulse sobre el botón de *START/STOP.* Al detectar una pulsación sobre el botón START/STOP, se calculará el tiempo de retardo entre pasos de acuerdo con la configuración previamente escogida por el usuario y se activará el bucle while() para el movimiento del motor.

```
424   if (digitalRead(buttonStart)== HIGH && lastButtonState1 == LOW) { // si el botón está pulsado
425
426         float pasosmin=0.00;
427         float pasosseg=0.00;
428
429         velrot=velFlujo/(578.08*pow((dtubo/2),2));
430         pasosmin=velrot*stepsperrev;
431         pasosseg=pasosmin/60.00;
432
433         retardo= 1000000.00/pasosseg;
434         delay(1000); // esperamos un segundo para evitar detectar múltiples pulsaciones
435
436
437         startTime=micros(); //iniciar temporizador para codific.
438         counter=0;
439         updatingFlow = true; // activamos el bucle de actualización de velFlujo
440         lcd.clear();
441
```

Figura 19. Cálculo del tiempo de retardo y activación del bombeo

Una vez en este bucle, el motor comenzará a girar de acuerdo con el retardo calculado. Además, por pantalla se mostrará la velocidad de flujo, la cual podrá ser actualizada en tiempo real sin necesidad de volver al menú principal, ni parar la bomba.

```
while(updatingFlow){

  //establecer sentido de giro del motor
if(rotationDirection=="horario"){
    digitalWrite(dirPin, HIGH); //girar sentido horario
  }else if(rotationDirection=="antihorario"){
    digitalWrite(dirPin, LOW);//girar sentido antihorario
  }

   //GIRO DEL MOTOR
   digitalWrite(stepPin,HIGH);
   delayMicroseconds(retardo);
   digitalWrite(stepPin,LOW);

  int newPosition = encoder.read() / 4;
  if (newPosition < 10) {

    lcd.setCursor(0, 0);
    lcd.print("Flowing... ");
    lcd.setCursor(0, 1);
    lcd.print("Flujo: ");


    newPosition = velFlujo;
    lcd.setCursor(7, 1);
    lcd.print(velFlujo);
    lcd.print(" uL/min    ");
    encoder.write(newPosition * 4);

   } else if (newPosition > 120) {
  newPosition = 120;
  encoder.write(newPosition * 4);}

```

Figura 20. Bucle para el movimiento del motor

Si el usuario modifica la velocidad de flujo, el tiempo de retardo se volverá a calcular de acuerdo con esta.

```
475         if (newPosition != velFlujo) {
476          velFlujo = newPosition;
477
478           lcd.setCursor(7, 1);
479           lcd.print(velFlujo);
480
481          float pasosmin=0.00;
482          float pasosseg=0.00;
483
484        velrot=velFlujo/(578.08*pow((dtubo/2),2));
485        pasosmin=velrot*stepsperrev;
486        pasosseg=pasosmin/60.00;
487
488        retardo= 1000000.00/pasosseg;
489      }
```

Figura 21. Actualización del tiempo de retardo

Por último, en este bucle se encuentra también la retroalimentación del encoder óptico. Simplemente se hará una comparación entre la velocidad de rotación calculada que debe llevar el motor, y la velocidad real medida por el encoder óptico. En caso de que la velocidad medida sea inferior a la calculada significa que el motor está girando despacio, por lo que se disminuirá el tiempo de retardo entre los pasos. En caso de que la velocidad medida por el encoder sea mayor a la calculada significa que el motor está girando muy rápido, por lo que el tiempo de retardo entre pasos deberá aumentar. En ambos casos, el tiempo de retardo variará en 15 us. Si la diferencia entre la velocidad medida y la calculada es menor a 15 us, no se hará ninguna corrección, pues de lo contrario el bucle estaría sumando y restando 15 us al tiempo de retardo infinitamente, provocando así una velocidad de flujo no constante. El tiempo de 15 us fue calculado durante la validación experimental, tal y como consta en el apartado 5.1 relativo a las Pruebas de funcionamiento del presente trabajo.

```
    int diferencia = velrot - medida; //se calcula la diferencia entre velocida objetivo y medi

   if (abs(diferencia) > 15) { //si el valor absoluto es mayor a 15, se corrige
      if (diferencia > 0) { // si es positiva está yendo muy lento
       retardo -= 15; // Corrección hacia una velocidad más rápida
      } else {          // de lo contrario
       retardo += 15; // Corrección hacia una velocidad más lenta
   }
  }
```

Figura 22. Retroalimentación del encoder óptico

Para medir y calcular esa velocidad de rotación mediante el encoder óptico se ha implementado el código mostrado a continuación en la figura 23.

```
496         //Leer el estado del encoder óptico
497        aState=digitalRead(codificadorA);
498         if(aState !=aLast){
499           bState = digitalRead(codificadorB);
500          if (bState != aState) {
501          counter++;
502         } else {
503          counter--;
504        }
505         }
506
507     aLast = aState;
508
509    endTime = micros(); // Detener el temporizador
510    elapsedTime = endTime - startTime; // Calcular el tiempo transcurrido en microsegundos
511    speed = (float)counter / (float)elapsedTime * 1000000.0; // Calcular la velocidad en pasos por segundo
512    rpm = speed * 60.0 / 997.0; // Convertir la velocidad a RPM
513    velFlujo=rpm*37,59; //Calcular la velocidad de flujo leida
514
```

Figura 23. Mediciones del encoder óptico

3.7 Diseño mecánico

Todos los componentes mecánicos de este proyecto han sido diseñados mediante el software *Inventor Professional* de *Autodesk* con el propósito de fabricarlos mediante impresión 3D.

Los componentes se pueden dividir en tres partes principales que son: el extensor del eje del motor, el cabezal de la bomba y el soporte para el lector óptico del encoder.

Extensión del eje del motor

El cabezal de la bomba y el disco de ranuras del encoder han de ir unidos al eje del motor para girar solidariamente. Por ello, en la figura 24 se presenta el diseño de una extensión del eje para albergar estos dos elementos. De forma más detallada se muestra en el plano 1 del presente proyecto.

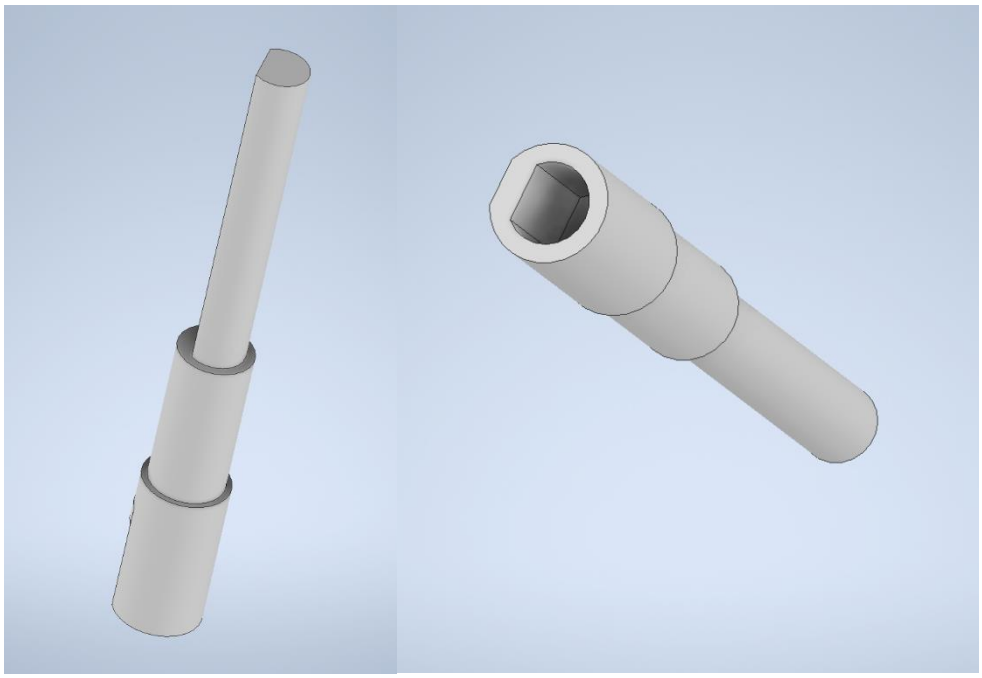


Figura 24. Extensión del eje del motor

El eje del motor va acoplado a la parte hembra de la pieza que se ve en la figura derecha. Seguidamente, en el cilindro intermedio se acoplará el disco del encoder óptico, y finalmente en la parte superior, el cabezal de la bomba.

Cabezal de la bomba

Esta pieza está inspirada en el cabezal de la bomba presente en el laboratorio.

Figura 25. Cabezal micro bomba NTC

El cabezal está formado a su vez por tres piezas distintas: las tapas, los pasadores y los rodillos. Ambas tapas son exactamente iguales.

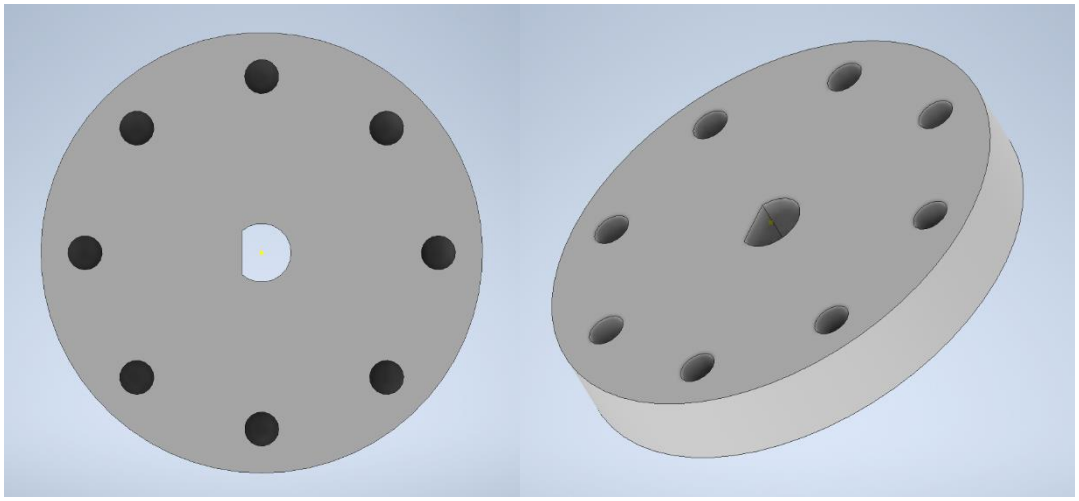


Figura 26. Tapa del cabezal

Como se puede apreciar en las figuras, se trata de una pieza completamente circular con un agujero pasante en el centro por el cual atravesará la extensión del eje del motor. Cuenta también con ocho agujeros no pasantes distribuidos de manera uniforme en los cuales se introducirán los pasadores de los ocho rodillos.

Los pasadores son simples cilindros de 3 mm de diámetro. Para el primer prototipo también están diseñados en el software 3D. Pero en el diseño final, serán varillas metálicas debido a que han de soportar un gran esfuerzo durante periodos notables de tiempo y, de estar impresas en PLA, se podrían partir.

Figura 27. Pasador

Los rodillos son cilindros con un agujero pasante en su centro de un diámetro ligeramente superior al de los pasadores para que puedan rodar con soltura.

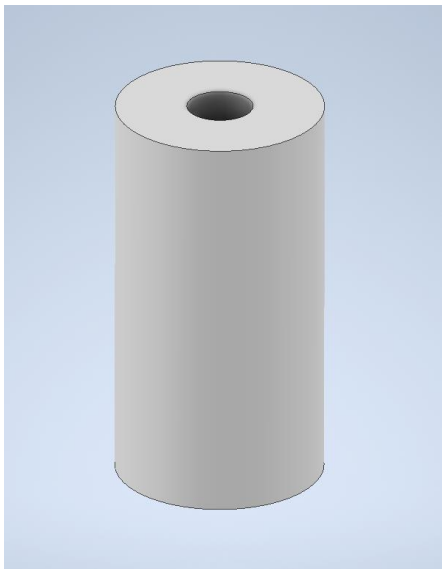


Figura 28. Rodillo

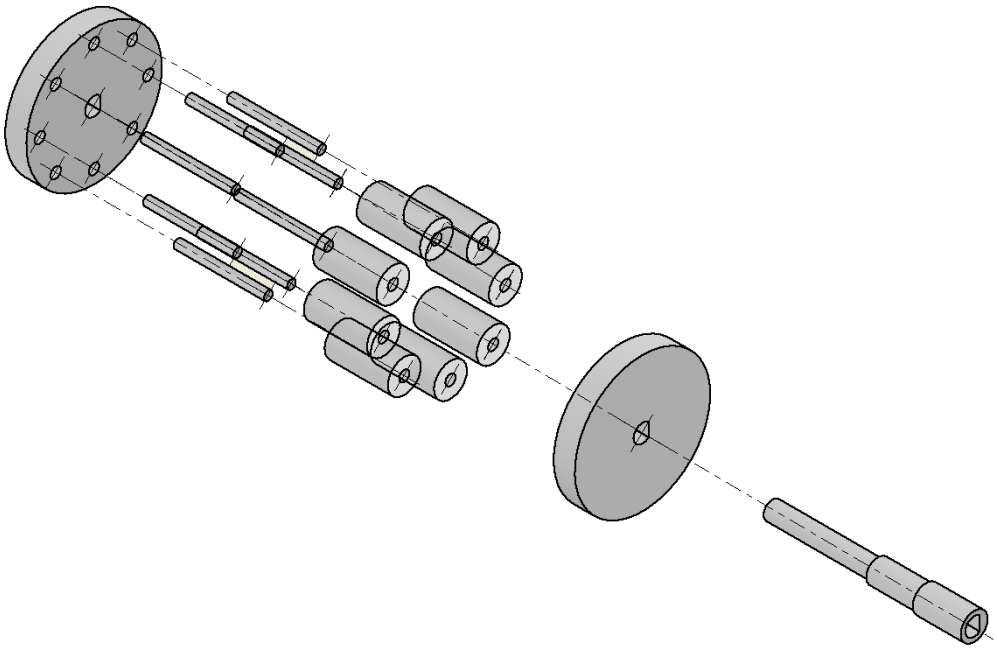


Figura 29. Explosionado del cabezal

Soporte para el lector óptico

El lector óptico del encoder no puede ir unido al eje del motor, debe permanecer estático. Por ello se ha diseñado un soporte adicional que encaja en la estructura del motor, y dará soporte al lector a la altura en la que se encuentra el disco ranurado.

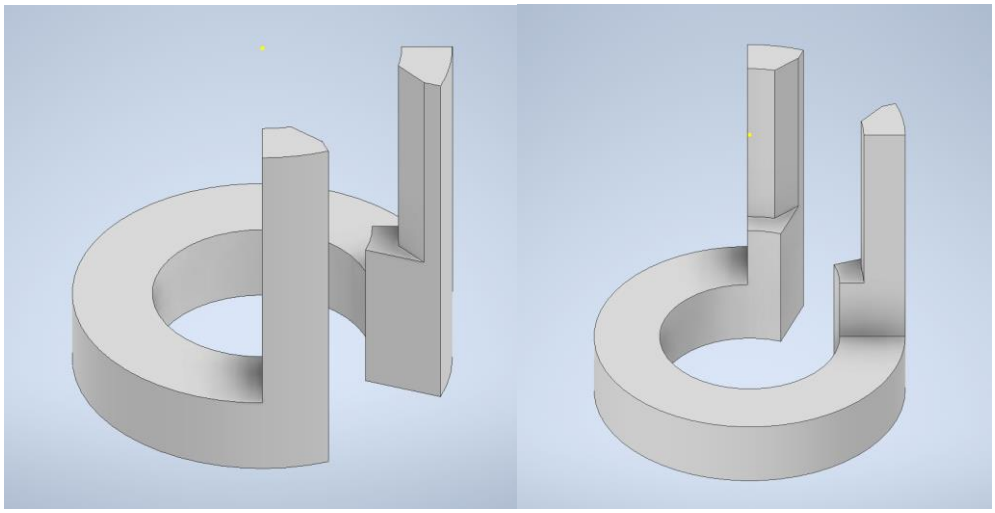


Figura 30. Soporte del lector óptico asociado al encoder

Los detalles de todas las piezas se pueden encontrar en el apartado de planos.

CAPÍTULO 4. MONTAJE DE LA BOMBA

4.1 Proceso de fabricación

La fabricación de los distintos componentes de la parte mecánica se ha llevado a cabo mediante impresión 3D.

El software de diseño *Inventor* permite preparar y exportar los archivos a formato STL. Los archivos .STL contienen la información geométrica necesaria para imprimir objetos en una impresora 3D, y son ampliamente utilizados en el campo de la fabricación aditiva. Estos archivos son utilizados por software de impresión 3D para generar las instrucciones necesarias para imprimir un objeto físico capa por capa. El software de impresión 3D traduce la geometría del archivo .STL en una serie de movimientos y comandos que la impresora 3D puede seguir para construir el objeto deseado. En este caso el software de impresión 3D utilizado ha sido el *Ultimaker Cura.*

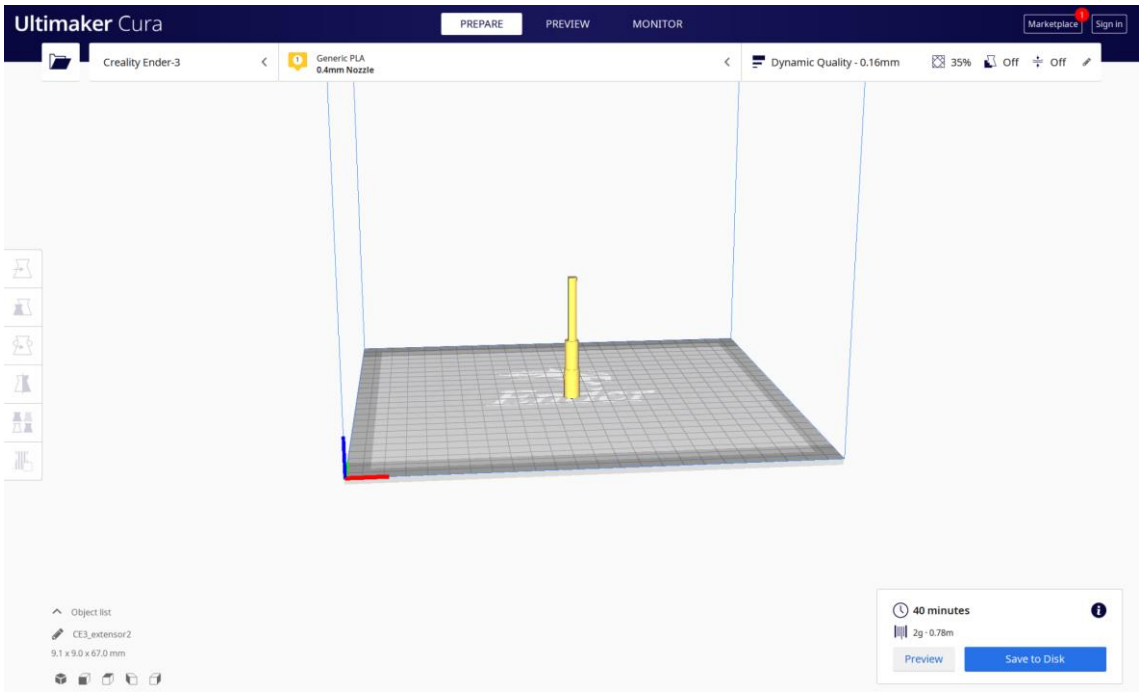


Figura 31. Entorno de Ultimaker Cura

Este software permite colocar la pieza en una posición concreta dentro del espacio de impresión, escoger la impresora 3D que se va utilizar, el material de impresión, la densidad de relleno, temperaturas, etc. Además, permite previsualizar capa a capa la pieza y proporciona información sobre el tiempo de impresión y la cantidad de material que se va a utilizar.

Una vez que todos los parámetros de impresión han sido establecidos, se genera el archivo .GCODE. Los archivos .GCODE contienen instrucciones específicas para controlar la impresión 3D, incluyendo movimientos, velocidades, temperaturas y otros parámetros, permitiendo a la impresora fabricar objetos de acuerdo con el modelo 3D especificado. En este caso la impresora 3D utilizada ha sido la *Creality Ender-3 Pro.* Una vez que se carga el archivo .GCODE en la impresora 3D, la impresora sigue las instrucciones paso a paso para imprimir el objeto en capas sucesivas, siguiendo el camino especificado y extruyendo el material según sea necesario.

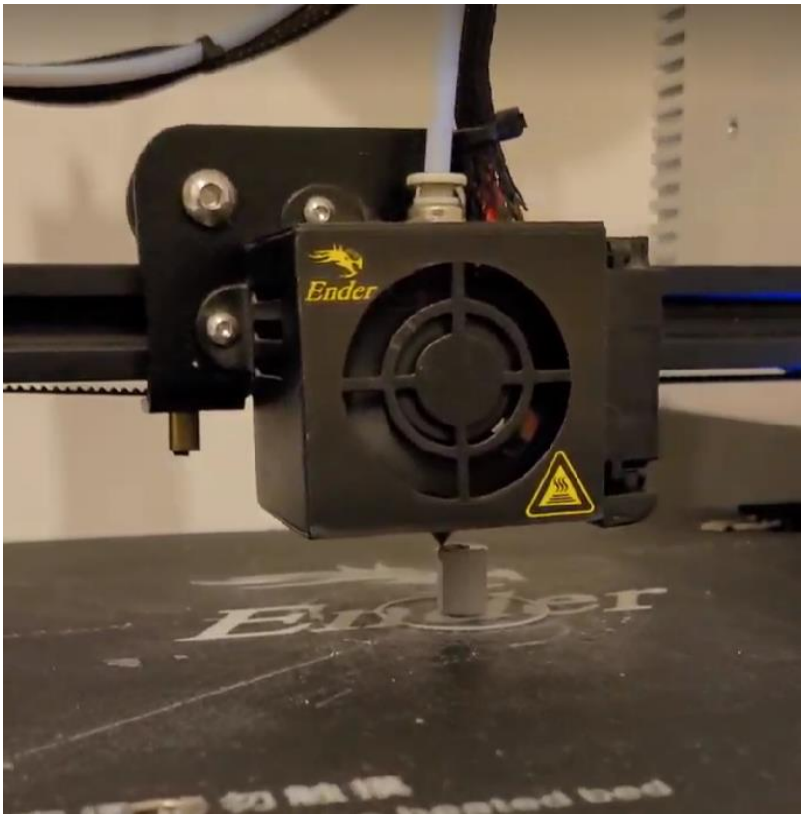


Figura 32. Impresión de pieza

4.2 Montaje y ensamblaje de los componentes

A continuación, se muestra en imágenes el montaje paso a paso de la micro bomba.

Montaje del cabezal

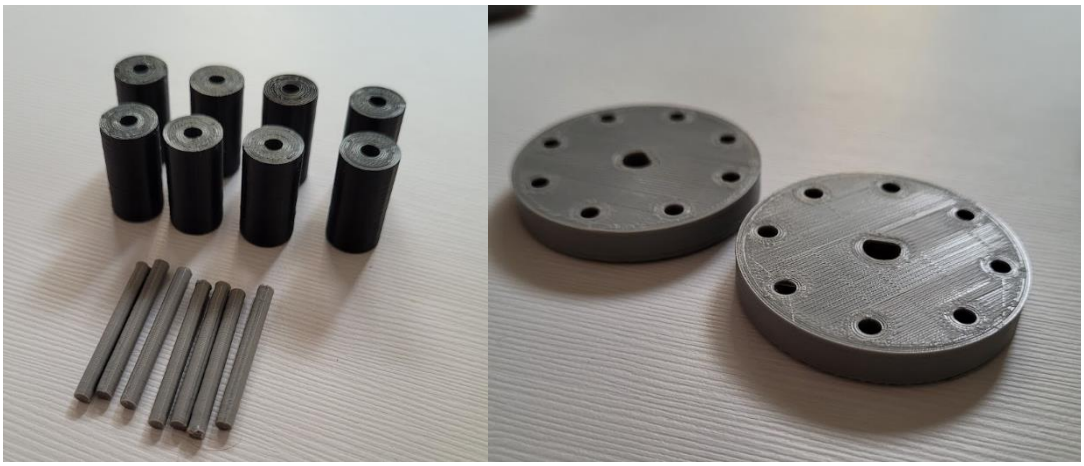


Figura 33. Partes del cabezal

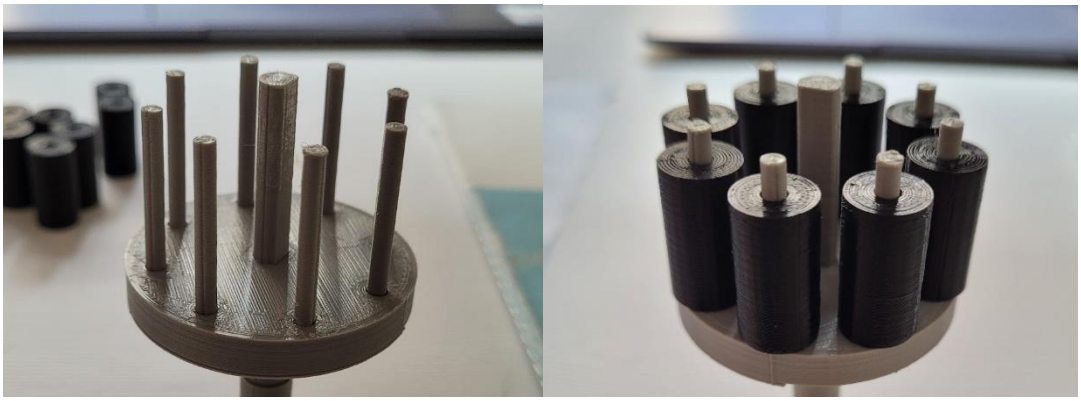


Figura 34. Montaje de las partes del cabezal

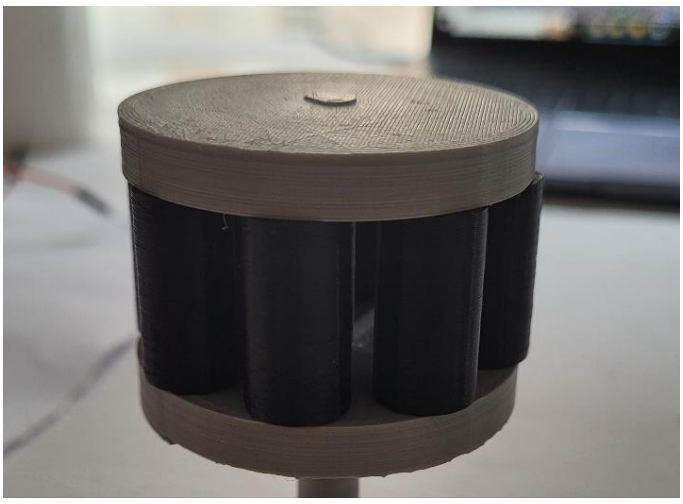


Figura 35. Cabezal de la bomba

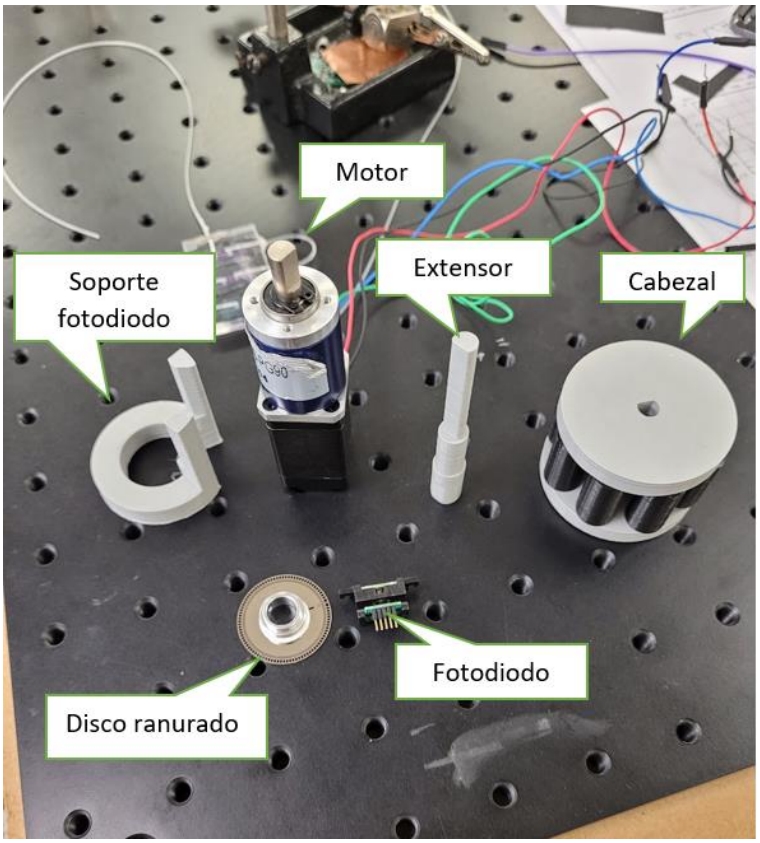


Figura 36. Todos los componentes de la parte mecánica de la micro bomba

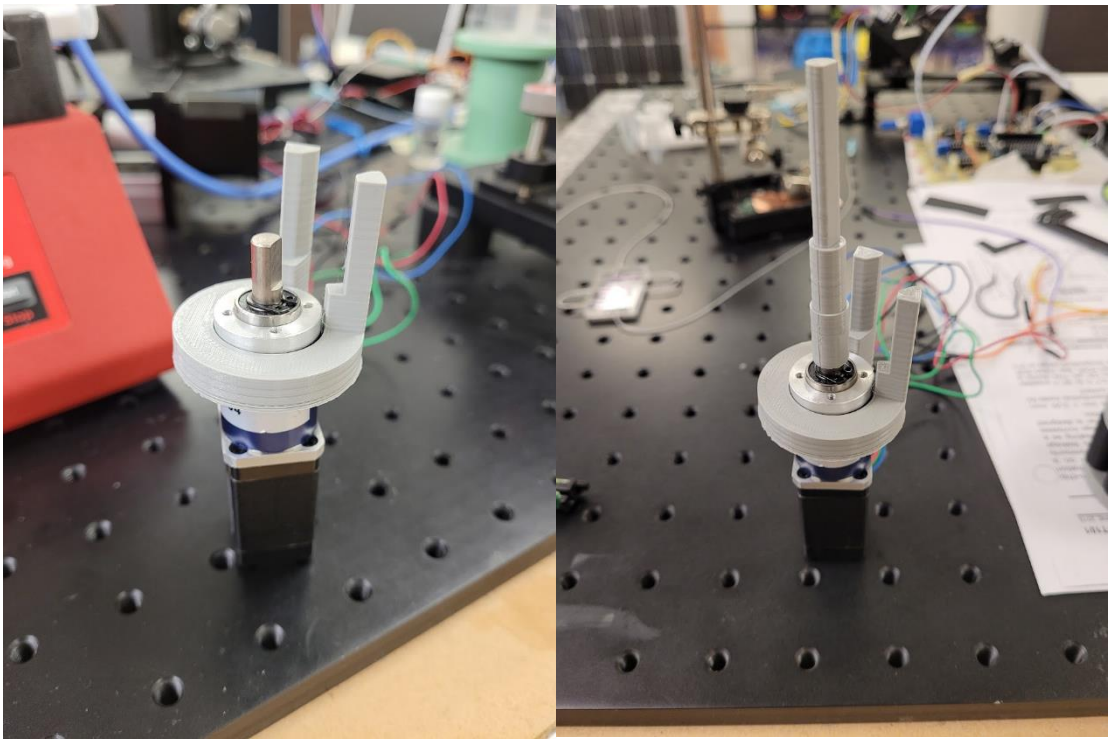


Figura 37.Motor con el soporte del lector óptico y extensor

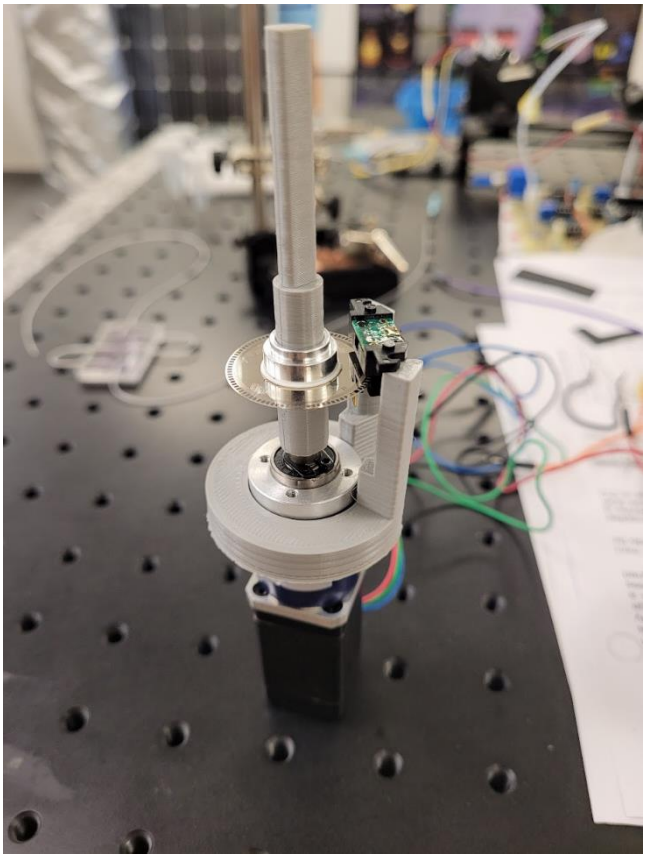


Figura 38.Encoder colocado

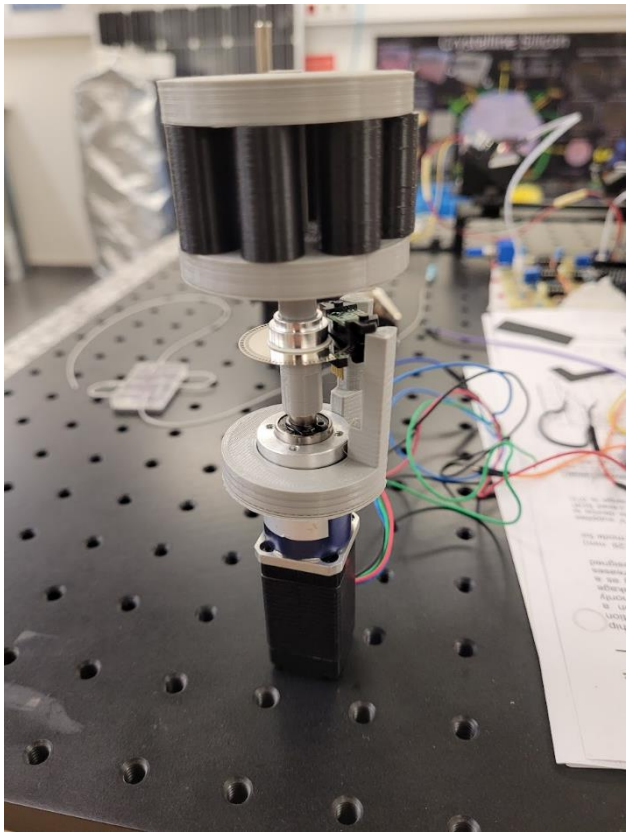


Figura 39. Montaje completo de la parte mecánica

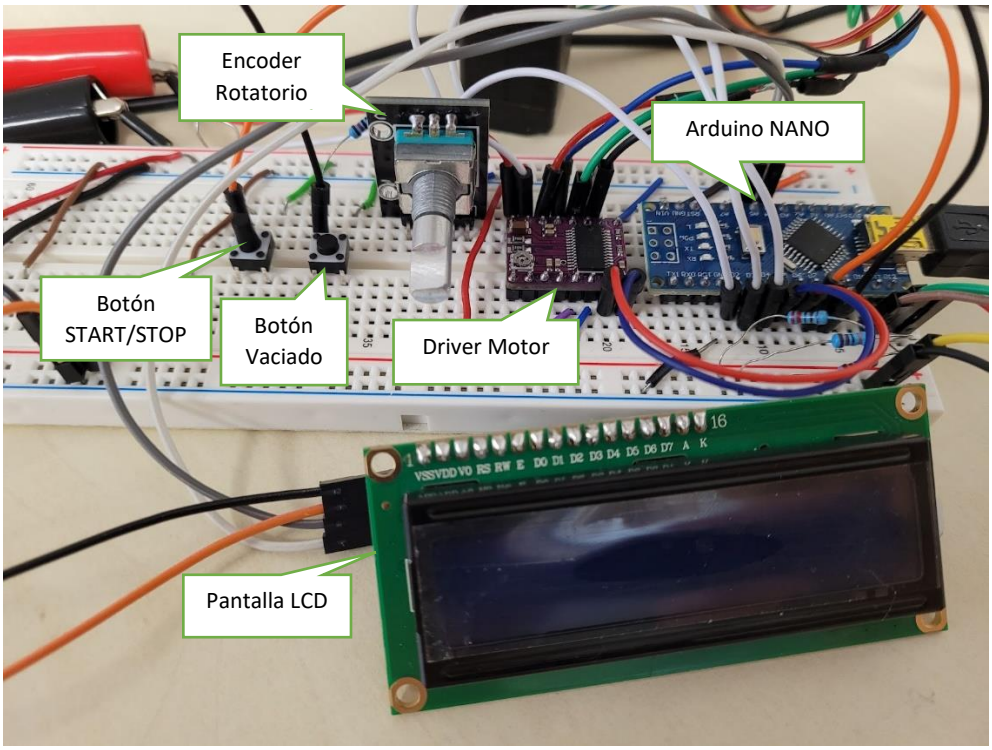


Figura 40. Montaje de la parte electrónica en una placa de prototipos

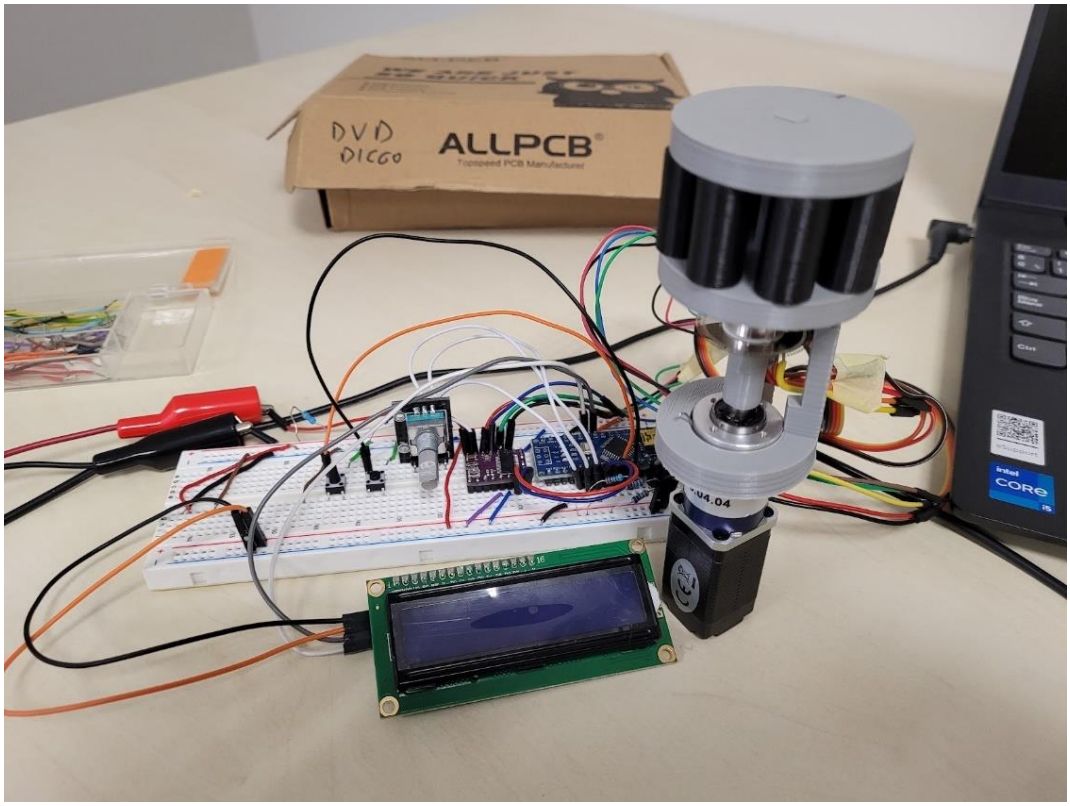


Figura 41. Montaje completo de la bom

CAPÍTULO 5. EVALUACIÓN Y VALIDACIÓN EXPERIMENTAL

5.1 Pruebas de funcionamiento

Para asegurar el correcto funcionamiento de la bomba peristáltica propuesta, se han llevado a cabo diversas pruebas y validaciones exhaustivas de sus diferentes componentes. Se han realizado pruebas tanto de las distintas partes del sistema por separado como de todo el conjunto. Estas pruebas se realizaron con el objetivo de verificar el funcionamiento adecuado del menú y su control, el giro del motor con la velocidad y calculada, la precisión del encoder óptico para medir y corregir la velocidad de giro, así como el funcionamiento general de todo el conjunto.

Se realizaron pruebas del menú implementado en la pantalla LCD y su correspondiente control a través del encoder rotatorio y el pulsador. Se verificó la correcta navegación por las diferentes opciones del menú y la respuesta adecuada a las acciones de selección y confirmación. Y también se comprobó que todas las funciones y parámetros del menú se ajustaran correctamente.

Así mismo, se llevó a cabo una prueba que engloba tanto el giro del motor como el funcionamiento del encoder óptico. Esta prueba se realizó con el objetivo de verificar el correcto funcionamiento del motor y del encoder, así como la precisión en la medición y corrección de la velocidad de giro.

Se ajustó la velocidad de giro del motor a través del tiempo de retardo entre los pasos, utilizando los cálculos previamente establecidos. Se midió la velocidad real de giro registrada por el encoder, y se comparó con la velocidad teórica calculada. Cualquier discrepancia entre ambas velocidades indicaría la necesidad de ajustes en el tiempo de retardo para corregir la velocidad de giro.

Durante la prueba, se registraron los resultados obtenidos y se analizaron estos resultados para determinar si se requerían ajustes en los parámetros de control.

Tal y como se presenta en la figura 40, se hicieron modificaciones en el código para hacer la medición y mostrar la información por el monitor serie de Arduino cada tres segundos.

```
void loop() {
  counter = 0;
  startTime = micros(); // Iniciar el temporizador

  for (i = 1; i <= 1036; i++) {
    digitalWrite(stepPin, HIGH);
    delayMicroseconds(stepDelay);
    digitalWrite(stepPin, LOW);
    //delayMicroseconds(stepDelay);

    // Leer el estado del encoder óptico
    aState = digitalRead(EncoderA);
    if (aState != aLast) {
      bState = digitalRead(EncoderB);
      if (bState != aState) {
        counter++;
      } else {
        counter--;
      }
    }
    aLast = aState;
  }

  endTime = micros(); // Detener el temporizador
  elapsedTime = endTime - startTime; // Calcular el tiempo transcurrido en microsegundos
  speed = (float)counter / (float)elapsedTime * 1000000.0; // Calcular la velocidad en pasos por segundo
  rpm = speed * 60.0 / 1036.0; // Convertir la velocidad a RPM
  velflujo=rpm*37.59;

  // Mostrar la cantidad de pasos en una vuelta del motor y la velocidad de giro
  Serial.print("Pasos en una vuelta: ");
  Serial.println(counter);
  Serial.print("Velocidad de giro: ");
  Serial.print(speed);
  Serial.println(" pasos/segundo");
   Serial.print(rpm);
  Serial.println(" RPM");
  Serial.print("Velocidad de flujo: ");
  Serial.println(velflujo);
  Serial.print("uL/min");
```

Figura 42. Medición con el encoder óptico y cálculo de velocidad

A continuación, en las figuras 41 y 42, se muestran las mediciones realizadas para una velocidad de flujo de 40 uL/min y de 90 uL/min.

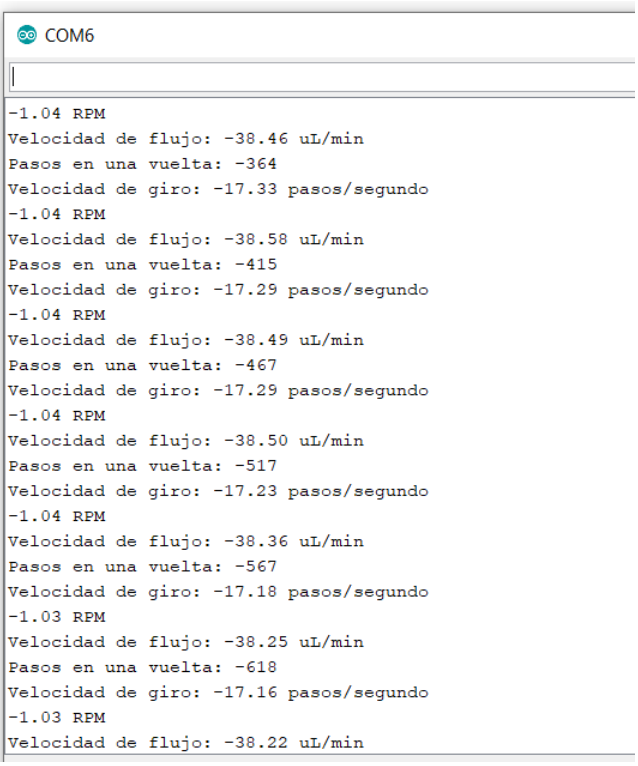


Figura 43. Resultados obtenidos para un flujo de 40 uL/min

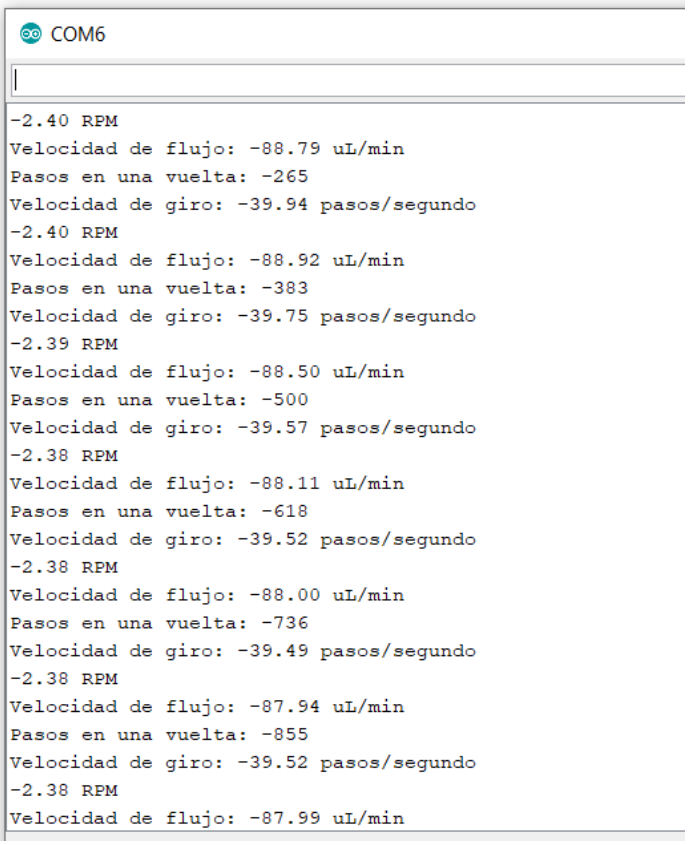


Figura 44. Resultados obtenidos para un flujo de 90 uL/min

Los resultados obtenidos en las pruebas pueden considerarse muy positivos ya que los valores tanto de flujo como de velocidad del motor son altamente repetitivos. Hay una pequeña diferencia entre el flujo medido y el requerido, pero es un error menor al 5%. Es decir, un error tolerable que además será corregido comparando la velocidad medida con la esperada y, de acuerdo con la diferencia, ajustando el tiempo de retardo entre pasos. Para poder analizar estos datos de una mejor manera y hacer los cálculos necesarios se elaboró un documento Excel.

	Flujo			Velocidad		
	Objetivo (uL/min)	Medido (uL/min)	Diferencia	Objetivo (rpm)	Medido (rpm)	Diferencia
Medición 1	15	13,86	1,14	0,399	0,41	0,011
Medición 2	15	13,72	1,28	0,399	0,41	0,011
Medición 3	15	13,6	1,4	0,399	0,42	0,021
Medición 4	15	13,44	1,56	0,399	0,42	0,021
Medición 5	15	13,77	1,23	0,399	0,42	0,021
Medición 6	15	13,64	1,36	0,399	0,42	0,021
Medición 7	15	13,28	1,72	0,399	0,42	0,021
Medición 8	15	12,97	2,03	0,399	0,42	0,021
Medición 9	15	13,88	1,12	0,399	0,42	0,021
		MEDIA	**1,38**		**MEDIA**	**0,02**
		PORCENTUAL	9,19%		PORCENTUAL	4,38%

Tabla 3. Valores medidos en las pruebas para un flujo de 15 Ul/min

	Flujo			Velocidad		
	Objetivo (uL/min)	Medido (uL/min)	Diferencia	Objetivo (rpm)	Medido (rpm)	Diferencia
Medición 1	40	38,46	1,54	1,064	1,04	0,024
Medición 2	40	38,58	1,42	1,064	1,04	0,024
Medición 3	40	38,49	1,51	1,064	1,04	0,024
Medición 4	40	38,5	1,5	1,064	1,04	0,024
Medición 5	40	38,36	1,64	1,064	1,04	0,024
Medición 6	40	38,25	1,75	1,064	1,04	0,024
Medición 7	40	38,22	1,78	1,064	1,04	0,024
Medición 8	40	38,34	1,66	1,064	1,03	0,034
Medición 9	40	38,42	1,58	1,064	1,03	0,034
		MEDIA	**1,59**		**MEDIA**	**0,026**
		PORCENTUAL	3,97%		PORCENTUAL	2,41%

Tabla 4. Valores medidos en las pruebas para un flujo de 40 uL/min

	Flujo			Velocidad		
	Objetivo (uL/min)	Medido (uL/min)	Diferencia	Objetivo (rpm)	Medido (rpm)	Diferencia
Medición 1	90	88,79	1,21	2,394	2,4	0,006
Medición 2	90	88,92	1,08	2,394	2,4	0,006
Medición 3	90	88,5	1,5	2,394	2,39	0,004
Medición 4	90	88,11	1,89	2,394	2,38	0,014
Medición 5	90	88	2	2,394	2,38	0,014
Medición 6	90	87,94	2,06	2,394	2,38	0,014
Medición 7	90	87,94	2,06	2,394	2,38	0,014
Medición 8	90	87,99	2,01	2,394	2,38	0,014
Medición 9	90	88,3	1,7	2,394	2,38	0,014
		MEDIA	**1,63**		**MEDIA**	**0,01**
		PORCENTUAL	1,82%		PORCENTUAL	0,37%

Tabla 5. Valores medidos en las pruebas para un flujo de 90 uL/min

El error medio de la velocidad da una idea de la corrección necesaria a aplicar. El error se va a corregir modificando en tiempo real el tiempo de retardo entre paso y paso. Si el tiempo de retardo es demasiado largo, la velocidad de giro será más lenta de lo esperado, mientras que, si el tiempo de retardo es demasiado corto, la velocidad de giro será más rápida de lo deseado. Para saber el tiempo de retardo que incrementar o decrementar es necesario fijarse en el porcentaje de error. Se propone que el error máximo sea del 1,5%. Para conseguir este ajuste, únicamente hace falta calcular el 1,5% del tiempo de retardo entre pasos para cada velocidad tal y como se muestra en la tabla 6.

vel. flujo(uL/min)	delay(us)	1,5% del delay(us)
10	12529,30	187,94
15	**8352,87**	**125,29**
20	6264,65	93,97
25	5011,72	75,18
30	4176,43	62,65
35	3579,80	53,70
40	**3132,32**	**46,98**
45	2784,29	41,76
50	2505,86	37,59
55	2278,05	34,17
60	2088,22	31,32
65	1927,58	28,91
70	1789,90	26,85
75	1670,57	25,06
80	1566,16	23,49
85	1474,04	22,11
90	**1392,14**	**20,88**
95	1318,87	19,78
100	1252,93	18,79
105	1193,27	17,90
110	1139,03	17,09
115	1089,50	16,34
120	1044,11	15,66

Tabla 6. Porcentaje de error para el tiempo de retardo

Por lo tanto, el incremento o decremento del tiempo de retardo entre pasos tendrá que ser de 15 us. Es decir, cada vez que se compare la velocidad medida con la objetivo y el error sea mayor del 1,5%, el tiempo de retardo entre pasos incrementará o decrementará, dependiendo de si la velocidad medida en menor o mayor a la objetivo, 15 us. Si el tiempo fuese mayor que ese, por ejemplo, de 50 us, para la velocidad máxima solo se podría corregir el error hasta el 4,8%. A continuación, en la tabla 7, se muestra el porcentaje de error máximo para cada velocidad haciendo un ajuste de 15 us en el tiempo de retardo.

vel. flujo(uL/min)	delay(us)	1,5% del delay(us)	% de error para 15 us
10	12529,30	187,94	0,12
15	**8352,87**	**125,29**	0,18
20	6264,65	93,97	0,24
25	5011,72	75,18	0,30
30	4176,43	62,65	0,36
35	3579,80	53,70	0,42
40	**3132,32**	**46,98**	0,48
45	2784,29	41,76	0,54
50	2505,86	37,59	0,60
55	2278,05	34,17	0,66
60	2088,22	31,32	0,72
65	1927,58	28,91	0,78
70	1789,90	26,85	0,84
75	1670,57	25,06	0,90
80	1566,16	23,49	0,96
85	1474,04	22,11	1,02
90	**1392,14**	**20,88**	1,08
95	1318,87	19,78	1,14
100	1252,93	18,79	1,20
105	1193,27	17,90	1,26
110	1139,03	17,09	1,32
115	1089,50	16,34	1,38
120	1044,11	15,66	1,44

Tabla 7. Porcentaje de error máximo para cada velocidad

Aplicando esto al programa el código queda tal y como se muestra en la figura 43.

```
    int diferencia = velrot - medida; //se calcula la diferencia entre velocida objetivo y medida

    if (abs(diferencia) > 15) { //si el valor absoluto es mayor a 15, se corrige
      if (diferencia > 0) { // si es positiva está yendo muy lento
       retardo -= 15; // Corrección hacia una velocidad más rápida
      } else {          // de lo contrario
       retardo += 15; // Corrección hacia una velocidad más lenta
    }
  }
```

Figura 45. Código para la corrección de la velocidad

Con esta prueba integral del giro del motor y del encoder óptico, se aseguró la adecuada funcionalidad y precisión en la velocidad de flujo de la bomba peristáltica.

CAPÍTULO 6. CONCLUSIONES

En este proyecto, se ha diseñado e implementado una micro bomba peristáltica de bajo coste y fácil operación para satisfacer las necesidades de los laboratorios del Centro de Tecnología Nanofotónica de Valencia en aplicaciones de sensado en tiempo real. La bomba ha sido diseñada para ser utilizada en un sistema de caracterización de sensores fotónicos portátil, por lo que se ha dado especial atención a reducir las dimensiones y el consumo eléctrico.

Los componentes seleccionados se han elegido cuidadosamente para cumplir con los requisitos del proyecto. El motor paso a paso proporciona un control preciso y una alta resolución en la velocidad de rotación, mientras que el driver asegura un rendimiento óptimo y eficiente. El uso de una placa de Arduino Nano como sistema de control permite una programación fácil y flexible.

Para lograr un control adecuado de la velocidad del motor, se ha incorporado un encoder óptico de 1000 CPR, que permite un monitoreo preciso y en tiempo real de la posición y velocidad del motor. Esta información es utilizada por el sistema de control para ajustar y mantener la velocidad de la bomba de acuerdo con los requerimientos de la aplicación.

La interfaz de usuario se ha implementado mediante una pantalla LCD y un encoder rotativo, lo que facilita la visualización y navegación por un menú de opciones. Esto permite el ajuste de parámetros y la configuración del sistema de manera intuitiva.

El cabezal de la bomba, un componente crítico para el funcionamiento de una bomba peristáltica ha sido diseñado específicamente para este proyecto utilizando impresión 3D. Esto ha permitido un diseño personalizado y adaptable, además de reducir los costes de producción.

Durante el desarrollo del proyecto, se realizaron diversas pruebas para validar el correcto funcionamiento de todas las partes del sistema. Se probaron el menú y su control, el giro del motor y la adecuación de su velocidad, así como el encoder óptico para medir y corregir la velocidad de rotación. Estas pruebas garantizaron el correcto rendimiento del sistema y su capacidad para cumplir con los requisitos de precisión y fiabilidad.

El presente trabajado de fin de grado ha servido como prototipo demostrador de una bomba peristáltica de bajo coste y gran precisión, utilizando herramientas de software y hardware libre, lo que lo hace accesible y adaptable a otros usuarios con necesidades similares.

Sin embargo, de cara a la elaboración del producto final para su uso en los laboratorios del NTC, quedan por resolver algunos temas que quedan fuera del alcance de este proyecto.

Para la integración final de la bomba será necesaria la fabricación de un circuito impreso con el fin de reducir al máximo posible el espacio ocupado por la electrónica y el cableado, así como para darle robustez al sistema. Además, será necesario fabricar una envoltura para la bomba en la que se encuentren todos los componentes. Finalmente, sería necesario hacer pruebas de sensado reales para comprobar el funcionamiento de la bomba y hacer correcciones en caso de que fuese necesario.

BIBLIOGRAFÍA

[1] Kunihiro Saito, Yoshihiro Hirayama, Yoshiki Kimura, and Taro Nakamura. Development of a peristaltic pump base on bowel peraltasis. Advanced Robotics. 2012.

[2] https://es.bimedis.com/catalog/ismatec-18097 Consultado el día 07/06/2023

[3]https://core.ac.uk/download/pdf/289974898.pdf Página 12.

[4] https://byeb.es/productos/abaque/

[5]https://glossary.slb.com/es/terms/v/volumetric_efficiency#:~:text=El%20cociente%20entre%20el%20volumen,la%20bomba%20en%20condiciones%20perfectas.

[6] https://www.luisllamas.es/tipos-motores-rotativos-proyectos-arduino/

[7] https://www.omc-stepperonline.com/es/motor-paso-a-paso-nema-14-bipolar-l-33-mm-con-relaci%C3%B3n-de-engranaje-5-1-caja-de-engranajes-planetarios-14hs13-0804s-pg5

[8] https://www.luisllamas.es/motores-paso-paso-arduino-driver-a4988-drv8825/

[9]https://www.google.com/search?rlz=1C1CHBF_esES916ES917&sxsrf=APwXEdcVAQBt5c7j4osHKoEVk7wbJP0IaA:1686910779444&q=driver+drv8825&tbm=shop&sa=X&ved=2ahUKEwiL7pquyMf_AhWZaqQEHfPdDrUQ0pQJegQIChAB&biw=1440&bih=817&dpr=1.5

[10] https://es.rs-online.com/web/p/encoders-opticos/7967806

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

DISEÑO Y REALIZACIÓN DE UNA MICRO BOMBA PERISTÁLTICA PARA APLICACIONES DE SENSADO EN TIEMPO REAL

DOCUMENTO Nº2: PLANOS

Trabajo final de grado

Grado en Ingeniería Electrónica Industrial y Automática

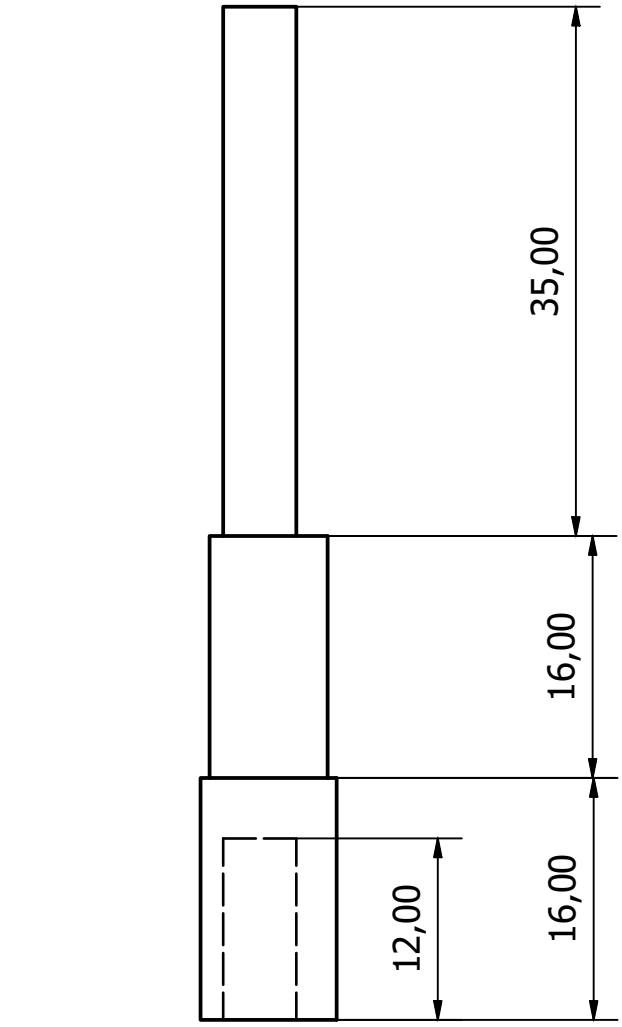
Autor: Sesma Jarauta, Jorge

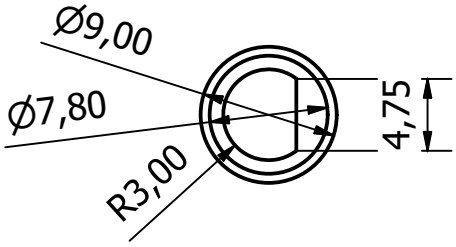
Tutor: Ponce Alcántara, Salvador

Cotutor: García Rupérez, Jaime

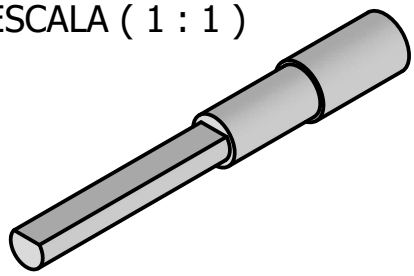
Curso académico: 2022/2023

Índice

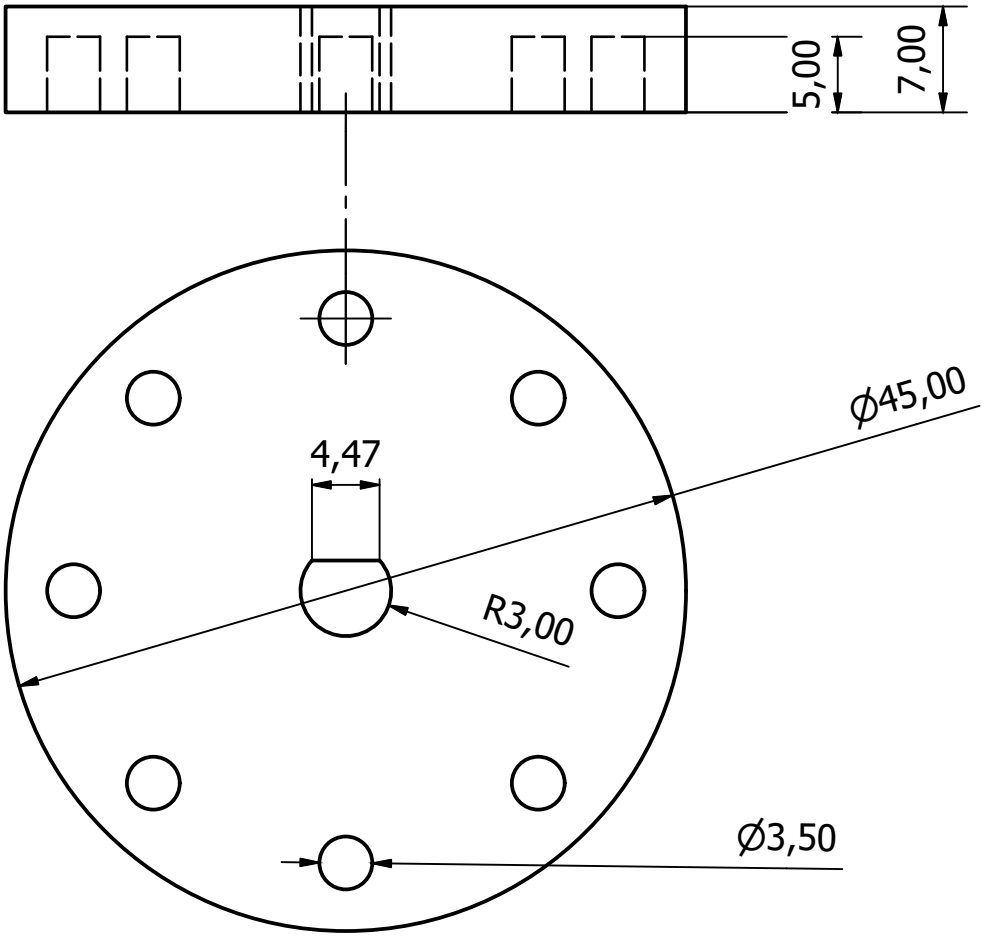




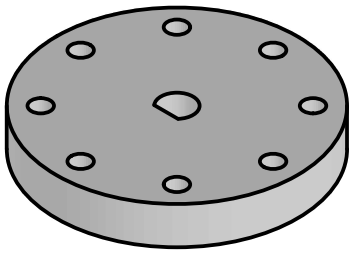
ESCALA (1 : 1)



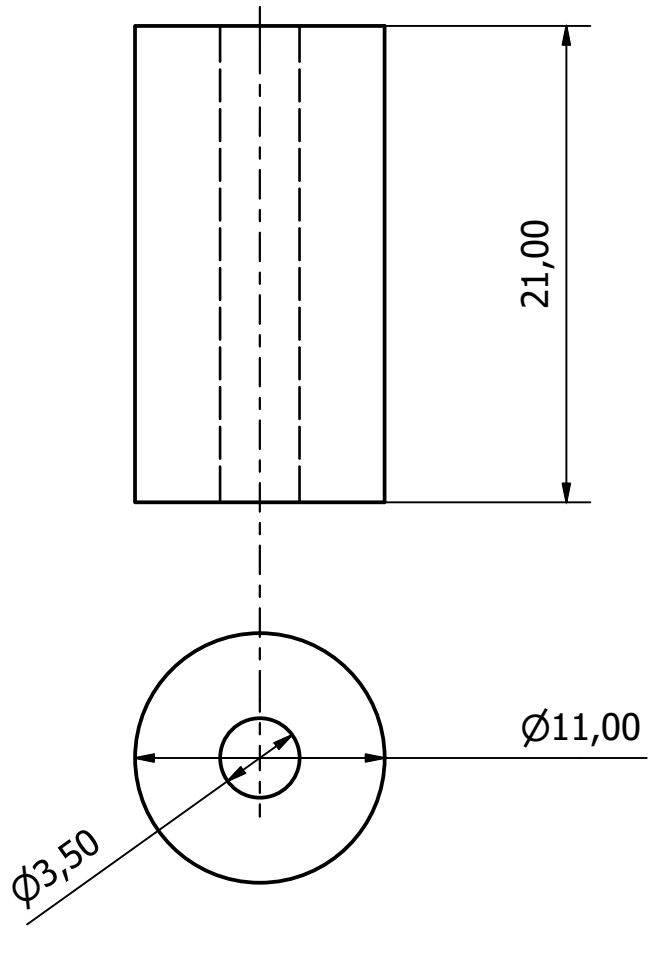
Autor Jorge Sesma Jarauta		Fecha 03/06/2023	
Trabajo de Fin de Grado	Extensor del eje		
	Escala 2:1		Pieza 1



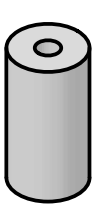
ESCALA (1 : 1)



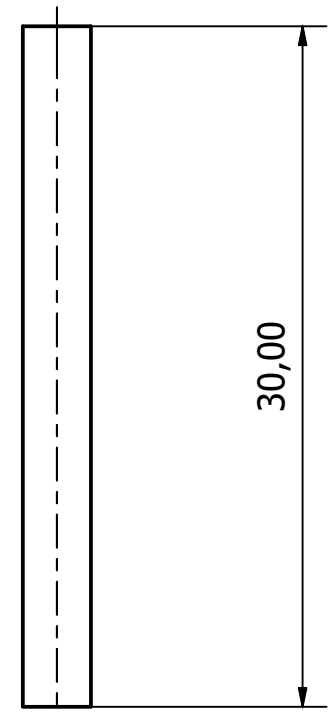
Autor: Jorge Sesma Jarauta		Date: 03/06/2023	
Trabajo de Fin de Grado	Tapa de cabezal		
	ESCALA 2:1	Piezas 2 Y 3	

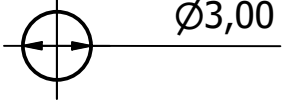


ESCALA (1 : 1)



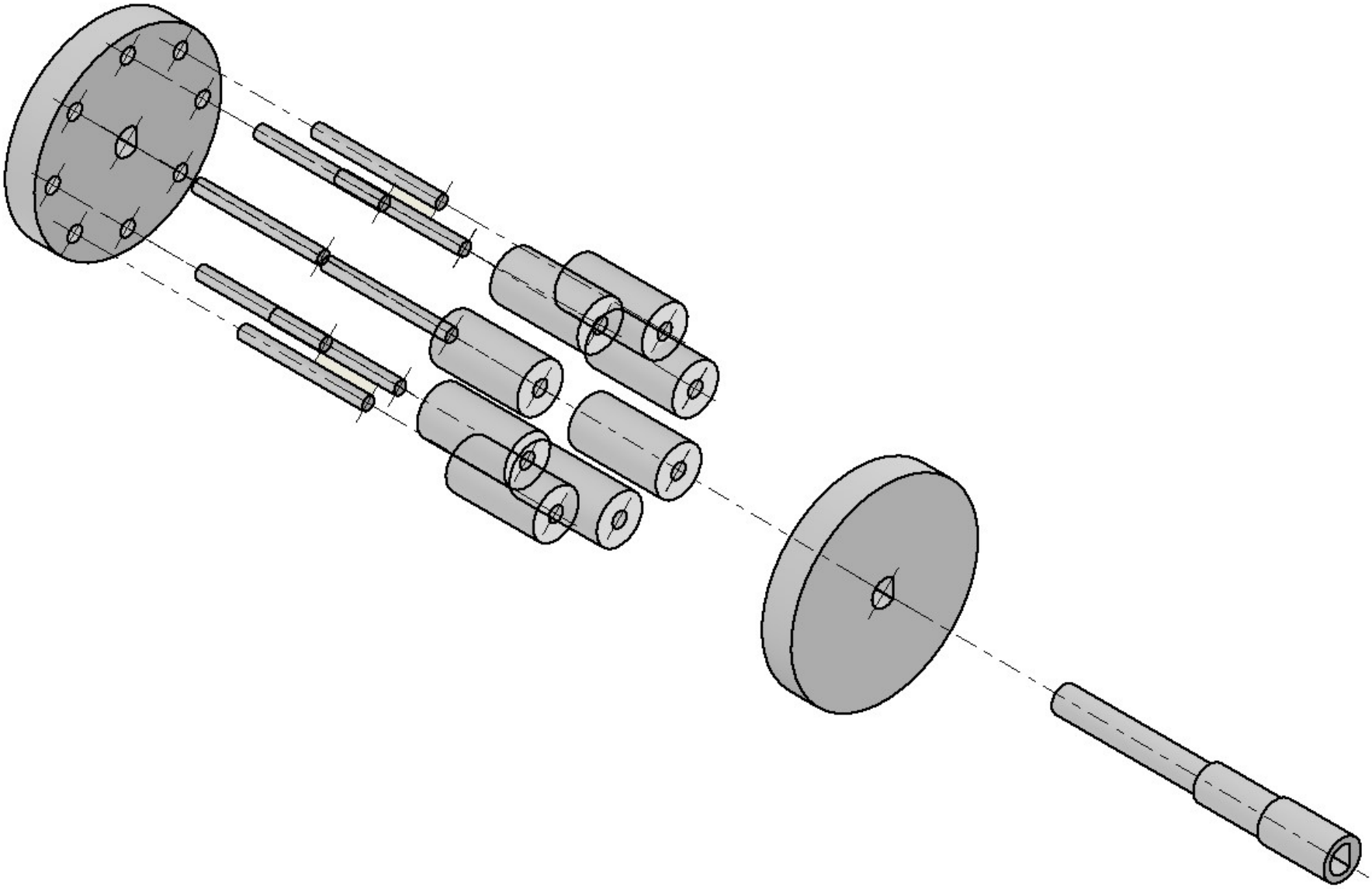
Autor		Date	
Jorge Sesma Jarauta		03/06/2023	
Trabajo de Fin de Grado	RODILLOS		
	ESCALA 3:1	Pieza 4	



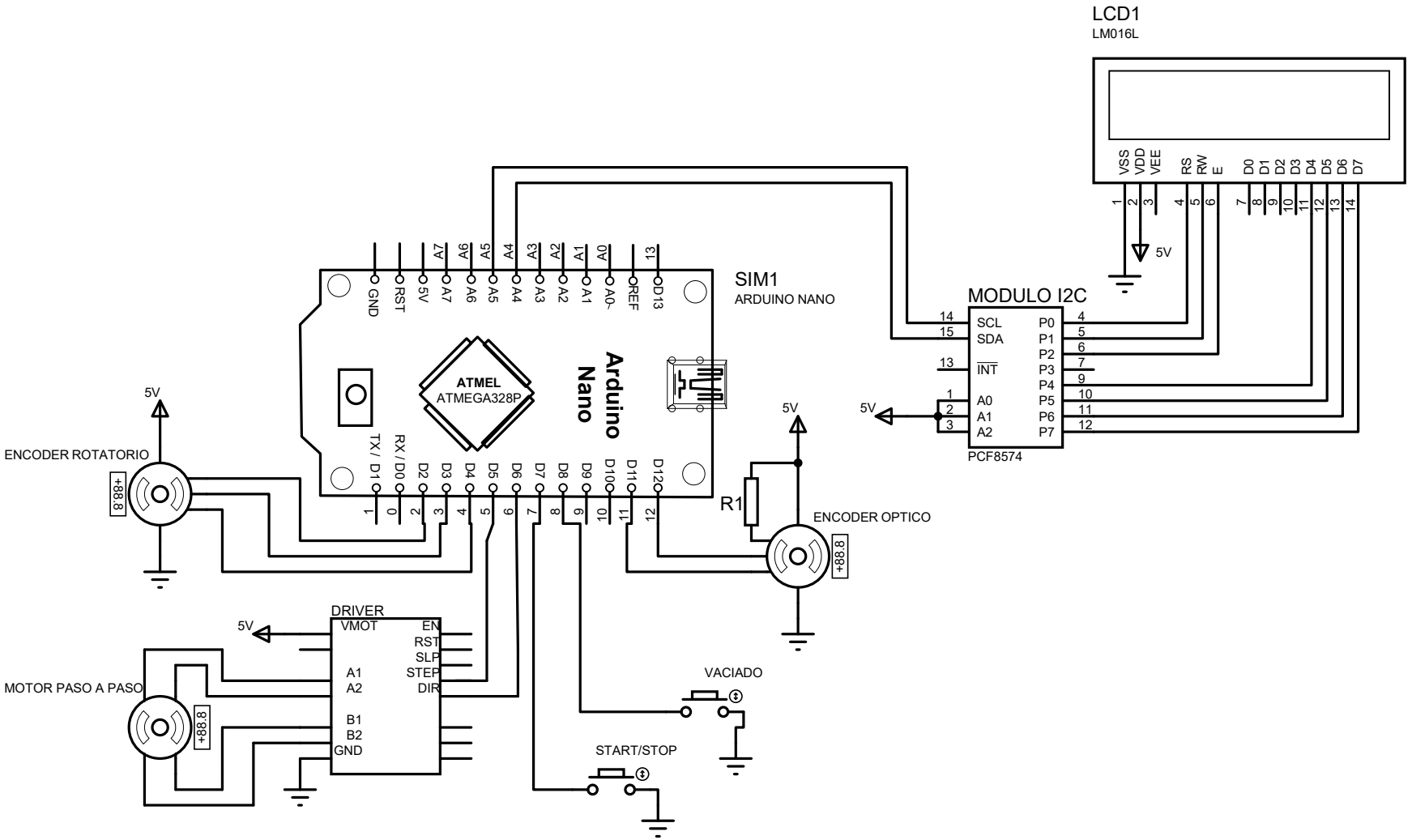


ESCALA (1 : 1)

Autor Jorge Sesma Jarauta		Date 03/06/2023	
Trabajo de Fin de Grado	PASADORES		
	ESCALA 3:1	Pieza 5	



Autor		Date	
Jorge Sesma Jarauta		03/06/2023	
Trabajo de Fin de Grado	Explosionado del cabezal		
	ESCALA 1:1		


LCD1
LM016L
VSS
VDD
VEE
RS
RW
E
D0
D1
D2
D3
D4
D5
D6
D7
5V
SIM1
ARDUINO NANO
MODULO I2C
SCL
SDA
INT
A0
A1
A2
P0
P1
P2
P3
P4
P5
P6
P7
PCF8574
ATMEL
ATMEGA328P
Arduino
Nano
GND
RST
5V
A7
A6
A5
A4
A3
A2
A1
A0
REF
D13
TX / D1
RX / D0
D2
D3
D4
D5
D6
D7
D8
D9
D10
D11
D12
ENCODER ROTATORIO
+88.8
R1
ENCODER OPTICO
DRIVER
VMOT
EN
RST
SLP
STEP
DIR
A1
A2
B1
B2
GND
MOTOR PASO A PASO
VACIADO
START/STOP

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

DISEÑO Y REALIZACIÓN DE UNA MICRO BOMBA PERISTÁLTICA PARA APLICACIONES DE SENSADO EN TIEMPO REAL

DOCUMENTO Nº3: PLIEGO DE CONDICIONES

Trabajo final de grado

Grado en Ingeniería Electrónica Industrial y Automática

Autor: Sesma Jarauta, Jorge

Tutor: Ponce Alcántara, Salvador

Cotutor: García Rupérez, Jaime

Curso académico: 2022/2023

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

Índice

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

1. CONDICIONES GENERALES

1.1 Vigencia

El presente pliego de condiciones técnicas tiene carácter de obligado cumplimiento una vez sellado y legalizado. Estará en vigor durante la producción del dispositivo electrónico. En caso de posibles discrepancias entre los distintos documentos que conforman el proyecto, el orden de prioridad es el siguiente:

1) Planos

2) Pliego de Condiciones

3) Presupuesto

4) Memoria

1.2 Descripción

El presente proyecto se refiere al diseño y realización de una micro bomba peristáltica para aplicaciones de sensado en tiempo real. El sistema está basado en un motor paso a paso, un microcontrolador y una interfaz de control. Sus características principales son la portabilidad y el bajo coste.

1.3 Pliegos oficiales

El contratista es responsable del cumplimiento de la Directiva de Baja Tensión(2014/35/UE) del Parlamento Europeo y del Consejo. Respecto a la gestión de residuos generados en la fabricación del dispositivo, es de obligado cumplimiento el Real Decreto 110/2015, de 20 de febrero, sobre residuos de aparatos eléctricos y electrónicos.

1.4 Modificaciones

Todas las modificaciones durante la ejecución del proyecto deben ser revisadas y aprobadas por el responsable de la dirección del proyecto.

1.5 Dirección e inspección

El responsable de la dirección del proyecto es el responsable de la correcta fabricación del dispositivo, pudiendo éste delegar en personal a cargo de la ejecución del proyecto.

2. CONDICIONES FACULTATIVAS

El director del proyecto debe revisar el trabajo realizado y los materiales empleados para ello. En caso de que los materiales no sean los indicados, estos deben estar aprobados por el director de proyecto. Todas las modificaciones o aclaraciones del proyecto deben constar en un documento redactado por la parte contratista. La fabricación del dispositivo se hará siguiendo estrictamente el proyecto que ha servido de base a la contratación y a las modificaciones que hayan sido aprobadas. En caso de dudas se formará un comité entre el director de la obra y el proyectista para aclararlas.

3. CONDICIONES DE LOS COMPONENTES

3.1 Descripción

3.1.1 Alimentación del sistema

La fuente de alimentación debe proporcionar un voltaje de hasta 12V y una corriente 2 A.

3.1.2 Motor

El tipo de motor debe ser paso a paso bipolar, y ser capaz de mover una carga radial de más de 50 N. Debe tener también una caja reductora para mantener un par elevado, y conseguir un ángulo de paso bajo. Además, la corriente por bobina ha de ser menor o igual a 1 A, y el motor de tamaño NEMA 17 o inferior.

3.1.3 Controlador del motor

El controlador del motor debe tener la capacidad para gestionar el giro de un motor paso a paso mediante salidas digitales. Ha de ser capaz de entregar una corriente de hasta 1 A por bobina en cada canal, y deberá incluir regulación de corriente y tensión. El tamaño ha de ser reducido, de no más de 30 x 20 mm.

3.1.4 Microcontrolador

Debe de tener al menos 13 salidas digitales, tensión de alimentación de 5 V, y conexión vía USB al PC. La placa debe hacer uso de software libre y su tamaño también será valorado.

3.1.5 Interfaz

El control del sistema ha de ser sencillo e intuitivo. Tiene que permitir la configuración de todos los parámetros de la bomba, y tiene que mostrar en tiempo real el estado de la bomba.

3.2 Controles de calidad

3.2.1 Alimentación del sistema

Para comprobar que la fuente de alimentación funciona correctamente, se ha de conectar un multímetro y comprobar que los valores de intensidad y voltaje proporcionados son corrector y repetitivos.

3.2.2 Motor

El motor se ha de poner en funcionamiento para comprobar su correcto giro, y mediante un multímetro asegurar que la intensidad demandada nunca supera a la nominal. También será necesario comprobar que no existe sobrecalentamiento.

3.2.3 Controlador del motor

Se debe de alimentar con 12 V, asegurarse que no hay sobrecalentamiento, y comprobar las salidas de los pines digitales.

3.2.4 Microcontrolador

Se debe alimentar mediante un puerto USB, comprobar que se enciende el led de la placa, y que el ordenador la reconoce.

3.2.5 Interfaz

Una vez montado siguiendo el esquema que aparece en los planos, se debe comprobar que se muestra por pantalla toda la información necesaria, y que los elementos de control responden correctamente.

4. CONDICIONES DE LA EJECUCIÓN

4.1 Descripción

Para el montaje se deben conectar todos los componentes electrónicos siguiendo el esquema de los planos, asegurando una buena conexión. Además, se ha de montar el motor y todas las partes mecánicas tal y como se ve en el apartado de montaje. El montaje se debe realizar en un lugar libre de polvo y suciedad. Para la manipulación de las distintas partes, se deben usar guantes de látex y, en caso de ensuciar alguno de estos componentes, se debe limpiar cuidadosamente con alcohol.

4.2 Controles de calidad

Se debe medir con un multímetro que los componentes hacen buen contacto, y que no existe ningún cortocircuito. Antes de montar el dispositivo completo, se debe comprobar que tanto el microcontrolador como el controlador del motor se alimentan correctamente una vez se conecten a la fuente de alimentación.

5. PRUEBAS Y AJUSTES FINALES O DE SERVICIO

Se debe comprobar que los valores de velocidad de flujo obtenidos son los expuestos en la memoria. Se debe comprobar también que el dispositivo cumple lo establecido en el Real Decreto 186/2016, de 6 de mayo, por el que se regula la compatibilidad electromagnética de los equipos eléctricos y electrónicos.

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

DISEÑO Y REALIZACIÓN DE UNA MICRO BOMBA PERISTÁLTICA PARA APLICACIONES DE SENSADO EN TIEMPO REAL

DOCUMENTO Nº4: PRESUPUESTO

Trabajo final de grado

Grado en Ingeniería Electrónica Industrial y Automática

Autor: Sesma Jarauta, Jorge

Tutor: Ponce Alcántara, Salvador

Cotutor: García Rupérez, Jaime

Curso académico: 2022/2023

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

Índice de tablas

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

Descripción	Precio (€/h)	Horas	Total (€)
Planificación del proyecto	35,00	20	700,00
Diseño de sistema y selección de componentes	35,00	40	1400,00
Desarrollo de software	35,00	42	1470,00
Montaje del sistema	20,00	8	160,00
Validación	35,00	25	875,00
Redacción del proyecto	35,00	30	1050,00
		Subtotal	**5655,00**
21% IVA			1187,55
		Total	**6842,55**

Tabla 1. Presupuesto de planificación, diseño y ejecución del proyecto

Descripción	Vida útil (años)	Horas de uso	Factor de amortización	Precio (€)	Total (€)
Fuente de alimentación	6	30	0,0173	312,00	5,4
Ordenador	8	165	0,0376	615,00	23,12
Multímetro	10	2	0,0023	83,00	0,19
				Subtotal	**28,71**
			21% IVA		6,03
				TOTAL	**34,74**

Tabla 2. Herramientas

Descripción	Precio (ud.)	Unidades	Total (€)
Microcontrolador Arduino NANO	2,29	1	2,29
Motor 14HS13-0804S-PG5	25,34	1	25,34
Codificador óptico 1000 CPR	38,76	1	38,76
Otros componentes (Cableado, botones, pantalla LCD, encoder)	10,78	1	10,78
Material impresión 3D (PLA)	0,03	47	1,41
		Subtotal	**78,58**
21% IVA			16,50
		Total	**95,08**

Tabla 3. Presupuesto de materiales

Descripción	Precio (€)
Diseño y ejecución del proyecto	5655,00
Herramientas	28,71
Materiales	78,58
Subtotal	**5762,29**
21% IVA	1210,08
Total	**6972,37**

Tabla 4. Resumen del presupuesto

El presupuesto total asciende a SEIS MIL NOVECIENTOS SETENTA Y DOS EUROS CON TREINTA Y SIETE CÉNTIMOS, IVA incluido.

ANEXO I

```
//Bibliotecas incluídas
#include <LiquidCrystal_I2C.h>
#include <Encoder.h>


// Inicializacion display LCD
LiquidCrystal_I2C lcd(0x27,16,2);

// establecemos el botón start/stop
const int buttonStart=7;
// establecemos el botón vaciado
const int buttonVacio=8;

// Pines del encoder rotativo y botón
const int encoderPinA = 3;
const int encoderPinB =4;
const int buttonPin = 2;

//Pines encoder optico
 int codificadorA=11;
 int codificadorB=12;

//variables cuentas del encoder
int counter;
int aLast;
int aState;
int bState;


unsigned long startTime;
unsigned long endTime;
unsigned long elapsedTime;
float speed;
float rpm;
unsigned long previousTime;


//pines stepper driver
int dirPin=5;
int stepPin=6;

Encoder encoder(encoderPinA, encoderPinB);
long encoderPosition = 0;
int lastMenuIndex = 0;
bool buttonPressed = false;
int lastButtonState = HIGH;
bool submenuActive = false;
long submenuPosition = 0;
int lastSubMenuIndex = 0;
const char *rotationDirection = "";
bool returningToMenu = false; // Inicializamos la variable returningToMenu en false

int velFlujo = 10; // Inicializamos el flujo en 10 uL/min
float dtubo = 0.00;
float velrot=0.00;

// Menú

const char *menuItems[] = {
    "Sentido de giro",
    "Vel. de flujo",
    "Diametro tubo",
    "Ver ajustes"
};
const int menuSize = sizeof(menuItems) / sizeof(menuItems[0]);

// Submenú
const char *submenuItems[] = {
  "Horario",
  "Antihorario"
};
const int submenuSize = sizeof(submenuItems) / sizeof(submenuItems[0]);

void setup() {

  Serial.begin(9600);
```

```
  //Inicializar pines
    pinMode(buttonPin, INPUT_PULLUP);
    pinMode(buttonStart, INPUT_PULLUP);
    pinMode(codificadorA, INPUT);
    pinMode(codificadorB, INPUT);
    pinMode(dirPin, OUTPUT);
    pinMode(stepPin, OUTPUT);

    aLast=digitalRead(codificadorA);

    lcd.init();
    lcd.backlight();
    lcd.setCursor(1,0);
    displayMenuItems(0);
    updateMenu(0);

    // Establecer los valores predeterminados de rotationDirection y dtubo
    rotationDirection = "horario";
    dtubo = 0.51;

  }

  unsigned long lastButtonPressTime = 0;
  const unsigned long debounceDelay = 50;

  // Muestra las opciones del menú en la pantalla LCD
void displayMenuItems(int startIndex) {
  int baseIndex = startIndex > 0 ? startIndex - 1 : startIndex;

  for (int i = 0; i < 2; i++) {
    int currentItemIndex = baseIndex + i;
    if (currentItemIndex < 0 || currentItemIndex >= menuSize) {
      continue;
    }
    lcd.setCursor(1, i);
    lcd.print(menuItems[currentItemIndex]);
    lcd.print("                ");
  }
}

// Actualiza la posición del signo '>' en el menú
void updateMenu(int menuIndex) {
  int cursorPos = (menuIndex > 0) ? 1 : 0;

  if (menuIndex != lastMenuIndex && menuIndex < menuSize) {
    lcd.setCursor(0, lastMenuIndex % 2);
    lcd.print(" ");
    lcd.setCursor(0, cursorPos);
    lcd.print(">");
    displayMenuItems(menuIndex);
    lastMenuIndex = menuIndex;
  }
}


void option1Function() {
  submenuPosition = 0;
  lastSubMenuIndex = -1;
  submenuActive = true;

  displaySubMenuItems(submenuPosition, submenuItems, submenuSize, lastSubMenuIndex);
  updateSubMenu(submenuPosition, submenuItems, submenuSize, lastSubMenuIndex);

  while (submenuActive) {
    long newSubmenuPosition = encoder.read() / 4;

    if (newSubmenuPosition < 0) {
      newSubmenuPosition = 0;
      encoder.write(newSubmenuPosition * 4);
    } else if (newSubmenuPosition > submenuSize - 1) {
      newSubmenuPosition = submenuSize - 1;
      encoder.write(newSubmenuPosition * 4);
    }

    if (newSubmenuPosition != submenuPosition) {
      submenuPosition = newSubmenuPosition;

     displaySubMenuItems(submenuPosition, submenuItems, submenuSize, lastSubMenuIndex);
     updateSubMenu(submenuPosition, submenuItems, submenuSize, lastSubMenuIndex);
}

buttonPressed = digitalRead(buttonPin) == LOW;
```

```
157
158 if (buttonPressed && lastButtonState == HIGH) {
159   lastButtonPressTime = millis();
160   lastButtonState = LOW;
161 }
162
163 if (!buttonPressed && lastButtonState == LOW) {
164   lastButtonState = HIGH;
165   unsigned long currentMillis = millis();
166
167   if (currentMillis - lastButtonPressTime > debounceDelay) {
168     if (submenuPosition == 0) {
169       rotationDirection = "horario";
170     } else if (submenuPosition == 1) {
171       rotationDirection = "antihorario";
172     }
173
174     submenuActive = false;
175
176     displayMenuItems(lastMenuIndex);
177     updateMenu(lastMenuIndex);
178   }
179 }
180 }
181 }
182
183 void displaySubMenuItems(int startIndex, const char **submenuItems, int submenuSize, int lastSubmenuIndex) {
184   int baseIndex = startIndex > 0 ? startIndex - 1 : startIndex;
185
186   for (int i = 0; i < 2; i++) {
187     int currentItemIndex = baseIndex + i;
188     if (currentItemIndex < 0 || currentItemIndex >= submenuSize) {
189       lcd.setCursor(1, i);
190       lcd.print("                ");
191       continue;
192     }
193
194     lcd.setCursor(1, i);
195     lcd.print(submenuItems[currentItemIndex]);
196     lcd.print("                ");
197   }
198 }
199
200
201 void updateSubMenu(int menuIndex, const char **submenuItems, int submenuSize, int &lastSubmenuIndex) {
202   int cursorPos = (menuIndex > 0) ? 1 : 0;
203
204   if (menuIndex != lastSubmenuIndex && menuIndex < submenuSize) {
205     lcd.setCursor(0, lastSubmenuIndex % 2);
206     lcd.print(" ");
207     lcd.setCursor(0, cursorPos);
208     lcd.print(">");
209
210     if (cursorPos == 0 && menuIndex == submenuSize -1) {
211       lcd.setCursor(0, 1);
212       lcd.print(" ");
213     }
214
215       displaySubMenuItems(menuIndex, submenuItems, submenuSize, lastSubmenuIndex);
216       lastSubmenuIndex = menuIndex;
217     }
218   }
219
220
221 void option2Function() {
222   lcd.clear();
223   lcd.setCursor(0, 0);
224   lcd.print("Flujo: ");
225
226   while (true) {
227     int newPosition = encoder.read() / 4;
228     if (newPosition < 10) {
229       newPosition = 10;
230       encoder.write(newPosition * 4);
231     } else if (newPosition > 120) {
232       newPosition = 120;
233       encoder.write(newPosition * 4);
234     }
235
236     if (newPosition != velFlujo) {
```

```
      velFlujo = newPosition;
      lcd.setCursor(7, 0);
      lcd.print(velFlujo);
      lcd.print(" uL/min    ");
    }

    buttonPressed = digitalRead(buttonPin) == LOW;
    if (buttonPressed && lastButtonState == HIGH) {
      lastButtonState = LOW;
      delay(50); // Pequeña pausa para evitar rebotes
      if (digitalRead(buttonPin) == LOW) {
        // El botón ha sido pulsado, salimos del bucle
        break;

      }
    }
  }

  // Al salir del bucle, hemos terminado de seleccionar la velocidad de flujo
  lcd.clear();
  displayMenuItems(lastMenuIndex);
  updateMenu(lastMenuIndex);

  returningToMenu = true; // Establecemos returningToMenu en true antes de volver al menú principal
}

void option3Function() {
  const char* submenuValues[] = {"////", "0.51", "0.75","1.00"};
  int submenuSize = sizeof(submenuValues) / sizeof(submenuValues[0]);;
  int lastSubMenuIndex = 0;
  bool submenuActive = true;

  displaySubMenuItems(0, submenuValues, submenuSize, lastSubMenuIndex);
  updateSubMenu(0, submenuValues, submenuSize, lastSubMenuIndex);

  while (submenuActive) {
    long newSubmenuPosition = encoder.read() / 4;

//movimiento por el menu
    if (newSubmenuPosition <0) {
      newSubmenuPosition = 0;
      encoder.write(newSubmenuPosition *4);
    } else if (newSubmenuPosition > submenuSize -1) {
      newSubmenuPosition = submenuSize -1;
      encoder.write(newSubmenuPosition * 4);
    }

    if (newSubmenuPosition != lastSubMenuIndex) {
      lastSubMenuIndex = newSubmenuPosition;
      displaySubMenuItems(lastSubMenuIndex, submenuValues, submenuSize, lastSubMenuIndex);
      updateSubMenu(lastSubMenuIndex, submenuValues, submenuSize, lastSubMenuIndex);
    }

    buttonPressed = digitalRead(buttonPin) == LOW;


    if (buttonPressed && lastButtonState == HIGH) {
      lastButtonPressTime = millis();
      lastButtonState = LOW;
    }

    if (!buttonPressed && lastButtonState == LOW) {
      lastButtonState = HIGH;
      unsigned long currentMillis = millis();

      if (currentMillis - lastButtonPressTime > debounceDelay) {
        switch (lastSubMenuIndex) {
          case 0:
            dtubo = 0.00;
            break;
          case 1:
            dtubo = 0.51;
            break;
          case 2:
            dtubo = 0.75;
            break;
            case 3:
            dtubo = 1.00;
            break;
          default:
            break;
        }
```

```
            submenuActive = false;

            displayMenuItems(lastMenuIndex);
            updateMenu(lastMenuIndex);
          }
        }
      }
    }


void option4Function() {
  char buffer[10]; char buffer2[10];
  dtostrf(velFlujo, 4, 2, buffer);  // convert velFlujo to a string with 2 decimal places
  dtostrf(dtubo, 4, 2, buffer2);

  strcat(buffer, " uL/m");  // concatenate " uL/min" to buffer
strcat(buffer2, " mm");     // concatenate " mm" to buffer2
  const char* submenuValues[] = {rotationDirection, buffer, buffer2, "Volver"};
  int submenuSize = sizeof(submenuValues) / sizeof(submenuValues[0]);
  int lastSubMenuIndex = 0;
  bool submenuActive = true;

  displaySubMenuItems(0, submenuValues, submenuSize, lastSubMenuIndex);
  updateSubMenu(0, submenuValues, submenuSize, lastSubMenuIndex);

  while (submenuActive) {
    long newSubmenuPosition = encoder.read() / 4;

    if (newSubmenuPosition < 0) {
      newSubmenuPosition = 0;
      encoder.write(newSubmenuPosition * 4);
    } else if (newSubmenuPosition > submenuSize - 1) {
      newSubmenuPosition = submenuSize - 1;
      encoder.write(newSubmenuPosition * 4);
    }

    if (newSubmenuPosition != lastSubMenuIndex) {
      lastSubMenuIndex = newSubmenuPosition;
      displaySubMenuItems(lastSubMenuIndex, submenuValues, submenuSize, lastSubMenuIndex);
      updateSubMenu(lastSubMenuIndex, submenuValues, submenuSize, lastSubMenuIndex);
    }

    buttonPressed = digitalRead(buttonPin) == LOW;

    if (buttonPressed && lastButtonState == HIGH) {
      lastButtonPressTime = millis();
      lastButtonState = LOW;
    }

    if (!buttonPressed && lastButtonState == LOW) {
      lastButtonState = HIGH;
      unsigned long currentMillis = millis();


      if (currentMillis - lastButtonPressTime > debounceDelay) {
        if (lastSubMenuIndex == submenuSize - 1) {
          // Go back to main menu if "Volver" option is selected
          submenuActive = false;
          displayMenuItems(lastMenuIndex);
          updateMenu(lastMenuIndex);
        }
      }
    }
  }
}



float retardo=0.00;

void loop() {


float stepsperrev=1036.00; //constante motor
```

```
393
394  bool lastButtonState1=LOW;
395  bool updatingFlow = false;
396  bool vacio=false;
397
398   if(digitalRead(buttonVacio)==HIGH&&lastButtonState1==LOW){ //si se pulsa el botón de vaciado
399
400    delay(1000); // esperamos un segundo para evitar detectar múltiples pulsaciones
401    vacio=true; //se activa el bucle de vaciado
402
403    while(vacio){
404
405     if(rotationDirection=="horario"){
406       digitalWrite(dirPin, LOW); //girar sentido antihorario
407     }else if(rotationDirection=="antihorario"){
408       digitalWrite(dirPin, HIGH);//girar sentido horario
409     }
410
411      digitalWrite(stepPin,HIGH);
412      delayMicroseconds(1044);// retardo para flujo de 120 uL/min
413      digitalWrite(stepPin,LOW);
414
415      lastButtonState1 = HIGH ;
416
417     if (digitalRead(buttonVacio) == HIGH && lastButtonState1 == HIGH) { // si se detecta una pulsación del botón de nuevo
418       vacio = false; // desactivamos el bucle de vaciado
419        }
420     }

421    delay(300); //evitar rebotes
422  }
423
424   if (digitalRead(buttonStart)== HIGH && lastButtonState1 == LOW) { // si el botón está pulsado
425
426      float pasosmin=0.00;
427      float pasosseg=0.00;
428
429      velrot=velFlujo/(578.08*pow((dtubo/2),2));
430      pasosmin=velrot*stepsperrev;
431      pasosseg=pasosmin/60.00;
432
433      retardo= 1000000.00/pasosseg;
434      delay(1000); // esperamos un segundo para evitar detectar múltiples pulsaciones
435
436
437      startTime=micros(); //iniciar temporizador para codific.
438      counter=0;
439      updatingFlow = true; // activamos el bucle de actualización de velFlujo
440      lcd.clear();
441
442     while(updatingFlow){
443
444       int newPosition = encoder.read() / 4;
445       if (newPosition < 10) {
446
447         lcd.setCursor(0, 0);

448         lcd.print("Flowing... ");
449         lcd.setCursor(0, 1);
450         lcd.print("Flujo: ");
451
452
453        newPosition = velFlujo;
454        lcd.setCursor(7, 1);
455        lcd.print(velFlujo);
456        lcd.print(" uL/min    ");
457        encoder.write(newPosition * 4);
458
459       } else if (newPosition > 120) {
460      newPosition = 120;
461      encoder.write(newPosition * 4);}
462
463      if (newPosition != velFlujo) {
464       velFlujo = newPosition;
465
466        lcd.setCursor(7, 1);
467        lcd.print(velFlujo);
468
469      float pasosmin=0.00;
470      float pasosseg=0.00;
471
```

```
    velrot=velFlujo/(578.08*pow((dtubo/2),2));
    pasosmin=velrot*stepsperrev;
    pasosseg=pasosmin/60.00;

    retardo= 1000000.00/pasosseg;
  }



   //establecer sentido de giro del motor
  if(rotationDirection=="horario"){
      digitalWrite(dirPin, HIGH); //girar sentido horario
   }else if(rotationDirection=="antihorario"){
      digitalWrite(dirPin, LOW);//girar sentido antihorario
   }

    //GIRO DEL MOTOR
    digitalWrite(stepPin,HIGH);
    delayMicroseconds(1000);
    digitalWrite(stepPin,LOW);


    //Leer el estado del encoder óptico
   aState=digitalRead(codificadorA);
    if(aState !=aLast){
     bState = digitalRead(codificadorB);
     if (bState != aState) {
```

```
      counter++;
     } else {
      counter--;
    }
     }

  aLast = aState;

 endTime = micros(); // Detener el temporizador
 elapsedTime = endTime - startTime; // Calcular el tiempo transcurrido en microsegundos
 speed = (float)counter / (float)elapsedTime * 1000000.0; // Calcular la velocidad en pasos por segundo
 rpm = speed * 60.0 / 997.0; // Convertir la velocidad a RPM
 velFlujo=rpm*37,59; //Calcular la velocidad de flujo leida


 unsigned long currentTime=millis(); //obtener el tiempo actual
 unsigned long timeDiff=currentTime-previousTime;

 //Si han pasado 3 segundos,  Mostrar la cantidad de pasos en una vuelta del motor y la velocidad de giro
if(timeDiff>=3000){
 Serial.print("\nPasos en una vuelta: ");
 Serial.println(counter);
 Serial.print("Velocidad de giro: ");
 Serial.print(speed);
 Serial.println(" pasos/segundo");
  Serial.print(rpm);
 Serial.println(" RPM");
```

```
 Serial.print( "Velocidad de flujo: ");
 Serial.print(velFlujo);
 Serial.print(" uL/min");



 int diferencia = velrot - rpm;

 if (abs(diferencia) > 15) {
   if (diferencia > 0) {
    retardo -= 15; // Corrección hacia una velocidad más rápida
   } else {
    retardo += 15; // Corrección hacia una velocidad más lenta
   }

 }


previousTime=currentTime; //actualizar el tiempo anterior
}


    lastButtonState1 = HIGH ;

  if (digitalRead(buttonStart) == HIGH && lastButtonState1 == HIGH) { // si se detecta una pulsación del botón de nuevo
```

```
        updatingFlow = false; // desactivamos el bucle de actualización de velFlujo
      }
    }
     // pequeña pausa para evitar detectar múltiples pulsaciones del botón
     lcd.clear();
     delay(300);


    }

    long newPosition = encoder.read() / 4;

    // Restringe el movimiento del encoder a un rango específico
    if (newPosition < 0) {
      newPosition = 0;
      encoder.write(newPosition * 4);
    } else if (newPosition > menuSize - 1) {
      newPosition = menuSize - 1;
      encoder.write(newPosition * 4);
    }

   // Actualiza el menú si la posición del encoder ha cambiado
    if (newPosition != encoderPosition) {
      encoderPosition = newPosition;
      displayMenuItems(encoderPosition);
      updateMenu(encoderPosition);
```

```
    }

    // Verifica si el botón está presionado
  buttonPressed = digitalRead(buttonPin) == LOW;

  if (buttonPressed && lastButtonState == HIGH && !returningToMenu) {
  lastButtonPressTime = millis();
  lastButtonState = LOW;
  }

  if (!buttonPressed && lastButtonState == LOW) {
  lastButtonState = HIGH;
  unsigned long currentMillis = millis();

if (currentMillis - lastButtonPressTime > debounceDelay && !returningToMenu) {
    switch (lastMenuIndex) {
      case 0:
        option1Function();
        break;
      case 1:
        option2Function();
        break;
      case 2:
        option3Function();
        break;
      case 3:
        option4Function();
        break;
```

```
      default:
        break;
    }
  }
}
  // Si se está volviendo al menú principal desde la opción 2, esperamos a que se libere el botón
    if (returningToMenu && !buttonPressed) {
      returningToMenu = false;
    }
  }
```

ANEXO II

STEPPER MOTOR CONTROLLER IC

Check for Samples: DRV8825

FEATURES

- PWM Microstepping Motor Driver
  - Built-In Microstepping Indexer
  - Five-Bit Winding Current Control Allows Up to 32 Current Levels
  - Low MOSFET On-Resistance
- 2.5-A Maximum Drive Current at 24 V, 25°C
- Built-In 3.3-V Reference Output
- 8.2-V to 45-V Operating Supply Voltage Range
- Thermally Enhanced Surface Mount Package

APPLICATIONS

- Automatic Teller Machines
- Money Handling Machines
- Video Security Cameras
- Printers
- Scanners
- Office Automation Machines
- Gaming Machines
- Factory Automation
- Robotics

DESCRIPTION

The DRV8825 provides an integrated motor driver solution for printers, scanners, and other automated equipment applications. The device has two H-bridge drivers, and can drive a bipolar stepper motor or two DC motors. The output driver block for each consists of N-channel power MOSFET's configured as full H-bridges to drive the motor windings. The DRV8825 can supply up to 2.5-A peak or 1.75-A RMS output current (with proper heatsinking at 24 V and 25°C).

A simple step/direction interface allows easy interfacing to controller circuits. Pins allow configuration of the motor in full-step up to 1/32-step modes. Decay mode is programmable.

Internal shutdown functions are provided for overcurrent protection, short circuit protection, undervoltage lockout and overtemperature.

The DRV8825 is available in a 28-pin HTSSOP package with PowerPAD™ (Eco-friendly: RoHS & no Sb/Br).

ORDERING INFORMATION(1)

T_A	PACKAGE(2)		ORDERABLE PART NUMBER	TOP-SIDE MARKING
–40°C to 85°C	PowerPAD™ (HTSSOP) - PWP	Reel of 2000	DRV8825PWPR	8825

(1) For the most current packaging and ordering information, see the Package Option Addendum at the end of this document, or see the TI web site at www.ti.com.
(2) Package drawings, thermal data, and symbolization are available at www.ti.com/packaging.

Please be aware that an important notice concerning availability, standard warranty, and use in critical applications of Texas Instruments semiconductor products and disclaimers thereto appears at the end of this data sheet.

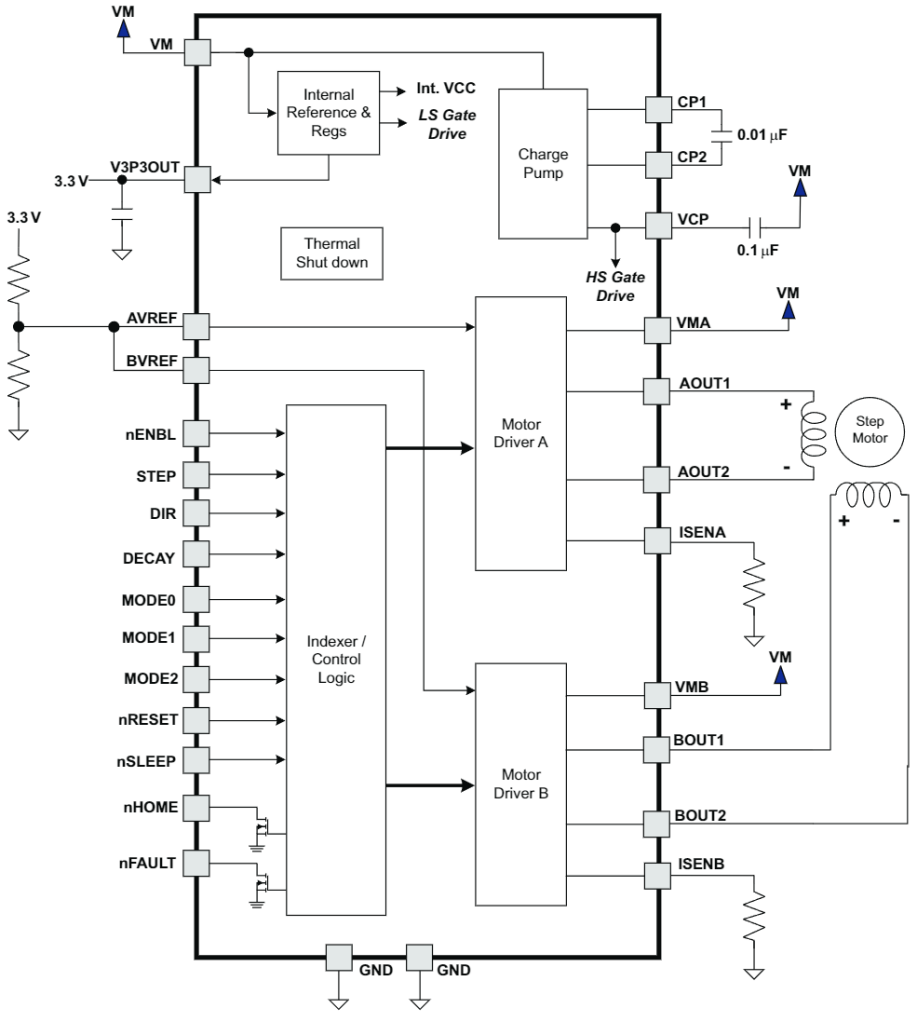
PowerPAD is a trademark of Texas Instruments.

PRODUCTION DATA information is current as of publication date. Products conform to specifications per the terms of the Texas Instruments standard warranty. Production processing does not necessarily include testing of all parameters.

Copyright © 2010–2011, Texas Instruments Incorporated

DEVICE INFORMATION

Functional Block Diagram



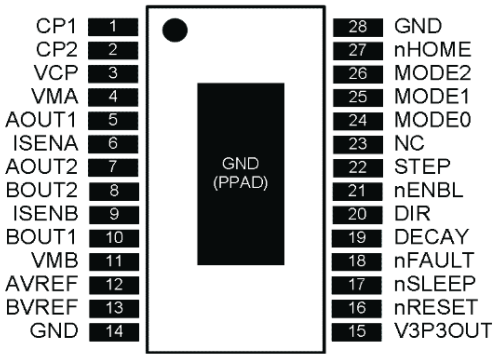
Copyright © 2010–2011, Texas Instruments Incorporated

Table 1. TERMINAL FUNCTIONS

NAME	PIN	I/O[1]	DESCRIPTION	EXTERNAL COMPONENTS OR CONNECTIONS
POWER AND GROUND				
GND	14, 28	-	Device ground	
VMA	4	-	Bridge A power supply	Connect to motor supply (8.2 - 45 V). Both pins must be connected to same supply.
VMB	11	-	Bridge B power supply	
V3P3OUT	15	O	3.3-V regulator output	Bypass to GND with a 0.47-µF 6.3-V ceramic capacitor. Can be used to supply VREF.
CP1	1	IO	Charge pump flying capacitor	Connect a 0.01-µF 50-V capacitor between CP1 and CP2.
CP2	2	IO	Charge pump flying capacitor	
VCP	3	IO	High-side gate drive voltage	Connect a 0.1-µF 16-V ceramic capacitor to VM.
CONTROL				
nENBL	21	I	Enable input	Logic high to disable device outputs and indexer operation, logic low to enable. Internal pulldown.
nSLEEP	17	I	Sleep mode input	Logic high to enable device, logic low to enter low-power sleep mode. Internal pulldown.
STEP	22	I	Step input	Rising edge causes the indexer to move one step
DIR	20	I	Direction input	Level sets the direction of stepping
MODE0	24	I	Microstep mode 0	MODE0 - MODE2 set the step mode - full, 1/2, 1/4, 1/8/ 1/16, or 1/32 step
MODE1	25	I	Microstep mode 1	
MODE2	26	I	Microstep mode 2	
DECAY	19	I	Decay mode	Low = slow decay, open = mixed decay, high = fast decay. Internal pulldown and pullup.
nRESET	16	I	Reset input	Active-low reset input initializes the indexer logic and disables the H-bridge outputs. Internal pulldown.
AVREF	12	I	Bridge A current set reference input	Reference voltage for winding current set. Normally AVREF and BVREF are connected to the same voltage. Can be connected to V3P3OUT.
BVREF	13	I	Bridge B current set reference input	
STATUS				
nHOME	27	OD	Home position	Logic low when at home state of step table
nFAULT	18	OD	Fault	Logic low when in fault condition (overtemp, overcurrent)
OUTPUT				
ISENA	6	IO	Bridge A ground / Isense	Connect to current sense resistor for bridge A.
ISENB	9	IO	Bridge B ground / Isense	Connect to current sense resistor for bridge B.
AOUT1	5	O	Bridge A output 1	Connect to bipolar stepper motor winding A. Positive current is AOUT1 → AOUT2
AOUT2	7	O	Bridge A output 2	
BOUT1	10	O	Bridge B output 1	Connect to bipolar stepper motor winding B. Positive current is BOUT1 → BOUT2
BOUT2	8	O	Bridge B output 2	

(1) Directions: I = input, O = output, OZ = tri-state output, OD = open-drain output, IO = input/output

Copyright © 2010–2011, Texas Instruments Incorporated



Copyright © 2010–2011, Texas Instruments Incorporated

ABSOLUTE MAXIMUM RATINGS

over operating free-air temperature range (unless otherwise noted) (1) (2)

			VALUE	UNIT
VMx	Power supply voltage range		–0.3 to 47	V
	Digital pin voltage range		–0.5 to 7	V
VREF	Input voltage		–0.3 to 4	V
	ISENSEx pin voltage		–0.3 to 0.8	V
	Peak motor drive output current, t < 1 µS		Internally limited	A
	Continuous motor drive output current(3)		2.5	A
	ESD rating	HBD (human body model)	2000	V
		CDM (charged device model)	500	
	Continuous total power dissipation		See Dissipation Ratings table	
T_J	Operating virtual junction temperature range		–40 to 150	°C
T_A	Operating ambient temperature range		–40 to 85	°C
T_{stg}	Storage temperature range		–60 to 150	°C

(1) Stresses beyond those listed under absolute maximum ratings may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated under recommended operating conditions is not implied. Exposure to absolute maximum rated conditions for extended periods may affect device reliability.
(2) All voltage values are with respect to network ground terminal.
(3) Power dissipation and thermal limits must be observed.

THERMAL INFORMATION

THERMAL METRIC(1)		DRV8825 PWP 28 PINS	UNITS
θ_{JA}	Junction-to-ambient thermal resistance(2)	31.6	°C/W
θ_{JCtop}	Junction-to-case (top) thermal resistance(3)	15.9	
θ_{JB}	Junction-to-board thermal resistance(4)	5.6	
ψ_{JT}	Junction-to-top characterization parameter(5)	0.2	
ψ_{JB}	Junction-to-board characterization parameter(6)	5.5	
θ_{JCbot}	Junction-to-case (bottom) thermal resistance(7)	1.4	

(1) For more information about traditional and new thermal metrics, see the *IC Package Thermal Metrics* application report, SPRA953.
(2) The junction-to-ambient thermal resistance under natural convection is obtained in a simulation on a JEDEC-standard, high-K board, as specified in JESD51-7, in an environment described in JESD51-2a.
(3) The junction-to-case (top) thermal resistance is obtained by simulating a cold plate test on the package top. No specific JEDEC-standard test exists, but a close description can be found in the ANSI SEMI standard G30-88.
(4) The junction-to-board thermal resistance is obtained by simulating in an environment with a ring cold plate fixture to control the PCB temperature, as described in JESD51-8.
(5) The junction-to-top characterization parameter, ψ_{JT}, estimates the junction temperature of a device in a real system and is extracted from the simulation data for obtaining θ_{JA}, using a procedure described in JESD51-2a (sections 6 and 7).
(6) The junction-to-board characterization parameter, ψ_{JB}, estimates the junction temperature of a device in a real system and is extracted from the simulation data for obtaining θ_{JA} , using a procedure described in JESD51-2a (sections 6 and 7).
(7) The junction-to-case (bottom) thermal resistance is obtained by simulating a cold plate test on the exposed (power) pad. No specific JEDEC standard test exists, but a close description can be found in the ANSI SEMI standard G30-88.

RECOMMENDED OPERATING CONDITIONS

over operating free-air temperature range (unless otherwise noted)

		MIN	NOM	MAX	UNIT
V_M	Motor power supply voltage range(1)	8.2		45	V
V_{REF}	VREF input voltage(2)	1		3.5	V
I_{V3P3}	V3P3OUT load current	0		1	mA
f_{PWM}	Externally applied PWM frequency	0		100	kHz

(1) All V_M pins must be connected to the same supply voltage.
(2) Operational at VREF between 0 V and 1 V, but accuracy is degraded.

Copyright © 2010–2011, Texas Instruments Incorporated

ELECTRICAL CHARACTERISTICS

over operating free-air temperature range (unless otherwise noted)

PARAMETER		TEST CONDITIONS	MIN	TYP	MAX	UNIT
POWER SUPPLIES						
I_{VM}	VM operating supply current	V_M = 24 V, f_{PWM} < 50 kHz		5	8	mA
I_{VMQ}	VM sleep mode supply current	V_M = 24 V		10	20	µA
V_{UVLO}	VM undervoltage lockout voltage	V_M rising		7.8	8.2	V
V3P3OUT REGULATOR						
V_{3P3}	V3P3OUT voltage	IOUT = 0 to 1 mA	3.2	3.3	3.4	V
LOGIC-LEVEL INPUTS						
V_{IL}	Input low voltage			0.6	0.7	V
V_{IH}	Input high voltage		2.2		5.25	V
V_{HYS}	Input hysteresis		0.3	0.45	0.6	V
I_{IL}	Input low current	VIN = 0	–20		20	µA
I_{IH}	Input high current	VIN = 3.3 V			100	µA
R_{PD}	Internal pulldown resistance			100		kΩ
nHOME, nFAULT OUTPUTS (OPEN-DRAIN OUTPUTS)						
V_{OL}	Output low voltage	I_O = 5 mA			0.5	V
I_{OH}	Output high leakage current	V_O = 3.3 V			1	µA
DECAY INPUT						
V_{IL}	Input low threshold voltage	For slow decay mode			0.8	V
V_{IH}	Input high threshold voltage	For fast decay mode	2			V
I_{IN}	Input current				±40	µA
R_{PU}	Internal pullup resistance (up to 3.3 V)			130		kΩ
R_{PD}	Internal pulldown resistance			80		kΩ
H-BRIDGE FETS						
$R_{DS(ON)}$	HS FET on resistance	V_M = 24 V, I_O = 1 A, T_J = 25°C		0.2		Ω
		V_M = 24 V, I_O = 1 A, T_J = 85°C		0.25	0.32	
	LS FET on resistance	V_M = 24 V, I_O = 1 A, T_J = 25°C		0.2		
		V_M = 24 V, I_O = 1 A, T_J = 85°C		0.25	0.32	
I_{OFF}	Off-state leakage current		–20		20	µA
MOTOR DRIVER						
f_{PWM}	Internal current control PWM frequency			30		kHz
t_{BLANK}	Current sense blanking time			4		µs
t_R	Rise time		30		200	ns
t_F	Fall time		30		200	ns
PROTECTION CIRCUITS						
I_{OCP}	Overcurrent protection trip level		3			A
t_{TSD}	Thermal shutdown temperature	Die temperature	150	160	180	°C
CURRENT CONTROL						
I_{REF}	xVREF input current	xVREF = 3.3 V	–3		3	µA
V_{TRIP}	xISENSE trip voltage	xVREF = 3.3 V, 100% current setting	635	660	685	mV
ΔI_{TRIP}	Current trip accuracy (relative to programmed value)	xVREF = 3.3 V, 5% current setting	–25		25	%
		xVREF = 3.3 V, 10% - 34% current setting	–15		15	
		xVREF = 3.3 V, 38% - 67% current setting	–10		10	
		xVREF = 3.3 V, 71% - 100% current setting	–5		5	
A_{ISENSE}	Current sense amplifier gain	Reference only		5		V/V

Copyright © 2010–2011, Texas Instruments Incorporated

TIMING REQUIREMENTS

			MIN	MAX	UNIT
1	f_{STEP}	Step frequency		250	kHz
2	$t_{WH(STEP)}$	Pulse duration, STEP high	1.9		µs
3	$t_{WL(STEP)}$	Pulse duration, STEP low	1.9		µs
4	$t_{SU(STEP)}$	Setup time, command to STEP rising	650		ns
5	$t_{H(STEP)}$	Hold time, command to STEP rising	650		ns
6	t_{ENBL}	Enable time, nENBL active to STEP	650		ns
7	t_{WAKE}	Wakeup time, nSLEEP inactive to STEP	1.7		ms

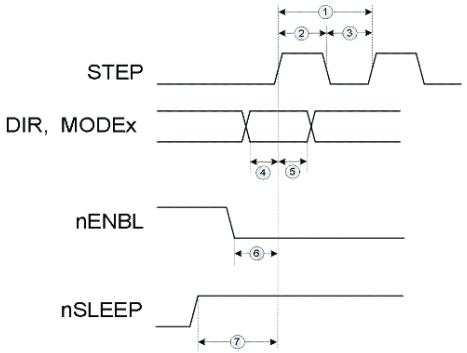


Figure 1. Timing Diagram

Copyright © 2010–2011, Texas Instruments Incorporated

FUNCTIONAL DESCRIPTION

PWM Motor Drivers

The DRV8825 contains two H-bridge motor drivers with current-control PWM circuitry. A block diagram of the motor control circuitry is shown in Figure 2.

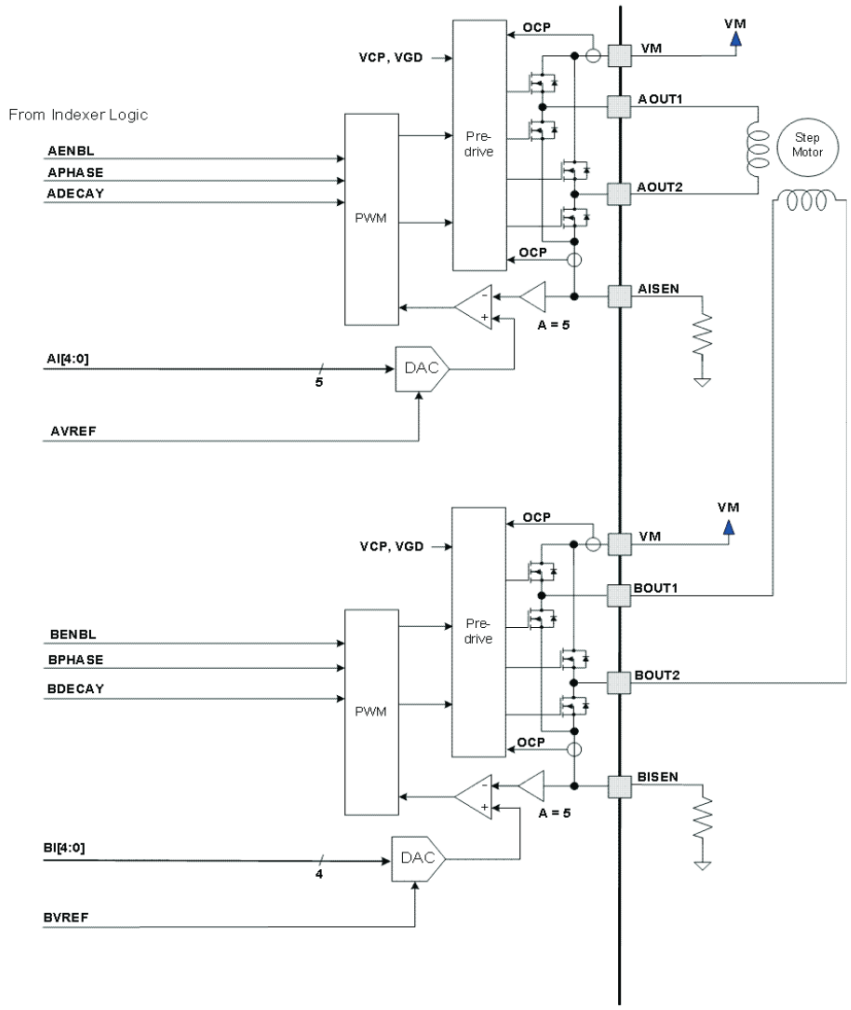


Figure 2. Motor Control Circuitry

Note that there are multiple VM motor power supply pins. All VM pins must be connected together to the motor supply voltage.

Copyright © 2010–2011, Texas Instruments Incorporated

Current Regulation

The current through the motor windings is regulated by a fixed-frequency PWM current regulation, or current chopping. When an H-bridge is enabled, current rises through the winding at a rate dependent on the DC voltage and inductance of the winding. Once the current hits the current chopping threshold, the bridge disables the current until the beginning of the next PWM cycle.

In stepping motors, current regulation is used to vary the current in the two windings in a semi-sinusoidal fashion to provide smooth motion.

The PWM chopping current is set by a comparator which compares the voltage across a current sense resistor connected to the xISEN pins, multiplied by a factor of 5, with a reference voltage. The reference voltage is input from the xVREF pins.

The full-scale (100%) chopping current is calculated in Equation 1.

$$I_{CHOP} = \frac{V_{REFX}}{5 \bullet R_{ISENSE}} \quad (1)$$

Example:

If a 0.25-Ω sense resistor is used and the VREFx pin is 2.5 V, the full-scale (100%) chopping current will be 2.5 V / (5 x 0.25 Ω) = 2 A.

The reference voltage is scaled by an internal DAC that allows fractional stepping of a bipolar stepper motor, as described in the microstepping indexer section below.

Decay Mode

During PWM current chopping, the H-bridge is enabled to drive current through the motor winding until the PWM current chopping threshold is reached. This is shown in Figure 3 as case 1. The current flow direction shown indicates positive current flow.

Once the chopping current threshold is reached, the H-bridge can operate in two different states, fast decay or slow decay.

In fast decay mode, once the PWM chopping current level has been reached, the H-bridge reverses state to allow winding current to flow in a reverse direction. As the winding current approaches zero, the bridge is disabled to prevent any reverse current flow. Fast decay mode is shown in Figure 3 as case 2.

In slow decay mode, winding current is re-circulated by enabling both of the low-side FETs in the bridge. This is shown in Figure 3 as case 3.

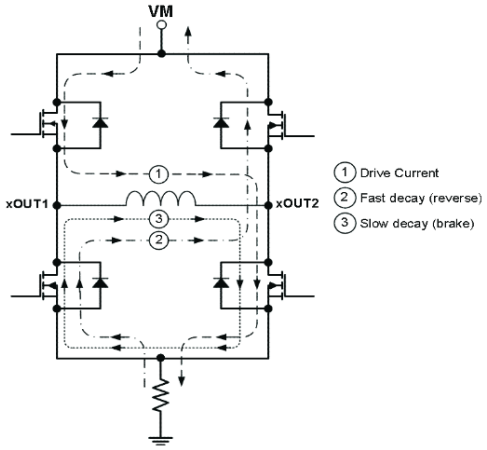


Figure 3. Decay Mode

Copyright © 2010–2011, Texas Instruments Incorporated

The DRV8825 supports fast decay, slow decay and a mixed decay mode. Slow, fast, or mixed decay mode is selected by the state of the DECAY pin - logic low selects slow decay, open selects mixed decay operation, and logic high sets fast decay mode. The DECAY pin has both an internal pullup resistor of approximately 130 kΩ and an internal pulldown resistor of approximately 80 kΩ. This sets the mixed decay mode if the pin is left open or undriven.

Mixed decay mode begins as fast decay, but at a fixed period of time (75% of the PWM cycle) switches to slow decay mode for the remainder of the fixed PWM period. This occurs only if the current through the winding is decreasing (per the indexer step table); if the current is increasing, then slow decay is used.

Blanking Time

After the current is enabled in an H-bridge, the voltage on the xISEN pin is ignored for a fixed period of time before enabling the current sense circuitry. This blanking time is fixed at 3.75 µs. Note that the blanking time also sets the minimum on time of the PWM.

Microstepping Indexer

Built-in indexer logic in the DRV8825 allows a number of different stepping configurations. The MODE0 - MODE2 pins are used to configure the stepping format as shown in Table 2.

Table 2. Stepping Format

MODE2	MODE1	MODE0	STEP MODE
0	0	0	Full step (2-phase excitation) with 71% current
0	0	1	1/2 step (1-2 phase excitation)
0	1	0	1/4 step (W1-2 phase excitation)
0	1	1	8 microsteps / step
1	0	0	16 microsteps / step
1	0	1	32 microsteps / step
1	1	0	32 microsteps / step
1	1	1	32 microsteps / step

Table 3 shows the relative current and step directions for different settings of MODEx. At each rising edge of the STEP input, the indexer travels to the next state in the table. The direction is shown with the DIR pin high; if the DIR pin is low the sequence is reversed. Positive current is defined as xOUT1 = positive with respect to xOUT2.

Note that if the step mode is changed while stepping, the indexer will advance to the next valid state for the new MODEx setting at the rising edge of STEP.

The home state is 45°. This state is entered at power-up or application of nRESET. This is shown in Table 3 by the shaded cells.

Table 3. Relative Current and Step Directions

1/32 STEP	1/16 STEP	1/8 STEP	1/4 STEP	1/2 STEP	FULL STEP 70%	WINDING CURRENT A	WINDING CURRENT B	ELECTRICAL ANGLE
1	1	1	1	1		100%	0%	0
2						100%	5%	3
3	2					100%	10%	6
4						99%	15%	8
5	3	2				98%	20%	11
6						97%	24%	14
7	4					96%	29%	17
8						94%	34%	20
9	5	3	2			92%	38%	23
10						90%	43%	25
11	6					88%	47%	28

Copyright © 2010–2011, Texas Instruments Incorporated

Table 3. Relative Current and Step Directions (continued)

1/32 STEP	1/16 STEP	1/8 STEP	1/4 STEP	1/2 STEP	FULL STEP 70%	WINDING CURRENT A	WINDING CURRENT B	ELECTRICAL ANGLE
12						86%	51%	31
13	7	4				83%	56%	34
14						80%	60%	37
15	8					77%	63%	39
16						74%	67%	42
17	9	5	3	2	1	71%	71%	45
18						67%	74%	48
19	10					63%	77%	51
20						60%	80%	53
21	11	6				56%	83%	56
22						51%	86%	59
23	12					47%	88%	62
24						43%	90%	65
25	13	7	4			38%	92%	68
26						34%	94%	70
27	14					29%	96%	73
28						24%	97%	76
29	15	8				20%	98%	79
30						15%	99%	82
31	16					10%	100%	84
32						5%	100%	87
33	17	9	5	3		0%	100%	90
34						–5%	100%	93
35	18					–10%	100%	96
36						–15%	99%	98
37	19	10				–20%	98%	101
38						–24%	97%	104
39	20					–29%	96%	107
40						–34%	94%	110
41	21	11	6			–38%	92%	113
42						–43%	90%	115
43	22					–47%	88%	118
44						–51%	86%	121
45	23	12				–56%	83%	124
46						–60%	80%	127
47	24					–63%	77%	129
48						–67%	74%	132
49	25	13	7	4	2	–71%	71%	135
50						–74%	67%	138
51	26					–77%	63%	141
52						–80%	60%	143
53	27	14				–83%	56%	146
54						–86%	51%	149
55	28					–88%	47%	152
56						–90%	43%	155
57	29	15	8			–92%	38%	158

Copyright © 2010–2011, Texas Instruments Incorporated

Table 3. Relative Current and Step Directions (continued)

1/32 STEP	1/16 STEP	1/8 STEP	1/4 STEP	1/2 STEP	FULL STEP 70%	WINDING CURRENT A	WINDING CURRENT B	ELECTRICAL ANGLE
58						–94%	34%	160
59	30					–96%	29%	163
60						–97%	24%	166
61	31	16				–98%	20%	169
62						–99%	15%	172
63	32					–100%	10%	174
64						–100%	5%	177
65	33	17	9	5		–100%	0%	180
66						–100%	–5%	183
67	34					–100%	–10%	186
68						–99%	–15%	188
69	35	18				–98%	–20%	191
70						–97%	–24%	194
71	36					–96%	–29%	197
72						–94%	–34%	200
73	37	19	10			–92%	–38%	203
74						–90%	–43%	205
75	38					–88%	–47%	208
76						–86%	–51%	211
77	39	20				–83%	–56%	214
78						–80%	–60%	217
79	40					–77%	–63%	219
80						–74%	–67%	222
81	41	21	11	6	3	–71%	–71%	225
82						–67%	–74%	228
83	42					–63%	–77%	231
84						–60%	–80%	233
85	43	22				–56%	–83%	236
86						–51%	–86%	239
87	44					–47%	–88%	242
88						–43%	–90%	245
89	45	23	12			–38%	–92%	248
90						–34%	–94%	250
91	46					–29%	–96%	253
92						–24%	–97%	256
93	47	24				–20%	–98%	259
94						–15%	–99%	262
95	48					–10%	–100%	264
96						–5%	–100%	267
97	49	25	13	7		0%	–100%	270
98						5%	–100%	273
99	50					10%	–100%	276
100						15%	–99%	278
101	51	26				20%	–98%	281
102						24%	–97%	284
103	52					29%	–96%	287

Copyright © 2010–2011, Texas Instruments Incorporated

Table 3. Relative Current and Step Directions (continued)

1/32 STEP	1/16 STEP	1/8 STEP	1/4 STEP	1/2 STEP	FULL STEP 70%	WINDING CURRENT A	WINDING CURRENT B	ELECTRICAL ANGLE
104						34%	–94%	290
105	53	27	14			38%	–92%	293
106						43%	–90%	295
107	54					47%	–88%	298
108						51%	–86%	301
109	55	28				56%	–83%	304
110						60%	–80%	307
111	56					63%	–77%	309
112						67%	–74%	312
113	57	29	15	8	4	71%	–71%	315
114						74%	–67%	318
115	58					77%	–63%	321
116						80%	–60%	323
117	59	30				83%	–56%	326
118						86%	–51%	329
119	60					88%	–47%	332
120						90%	–43%	335
121	61	31	16			92%	–38%	338
122						94%	–34%	340
123	62					96%	–29%	343
124						97%	–24%	346
125	63	32				98%	–20%	349
126						99%	–15%	352
127	64					100%	–10%	354
128						100%	–5%	357

nRESET, nENBLE and nSLEEP Operation

The nRESET pin, when driven active low, resets internal logic, and resets the step table to the home position. It also disables the H-bridge drivers. The STEP input is ignored while nRESET is active.

The nENBL pin is used to control the output drivers and enable/disable operation of the indexer. When nENBL is low, the output H-bridges are enabled, and rising edges on the STEP pin are recognized. When nENBL is high, the H-bridges are disabled, the outputs are in a high-impedance state, and the STEP input is ignored.

Driving nSLEEP low will put the device into a low power sleep state. In this state, the H-bridges are disabled, the gate drive charge pump is stopped, the V3P3OUT regulator is disabled, and all internal clocks are stopped. In this state all inputs are ignored until nSLEEP returns inactive high. When returning from sleep mode, some time (approximately 1 ms) needs to pass before applying a STEP input, to allow the internal circuitry to stabilize. Note that nRESET and nSLEEP have internal pulldown resistors of approximately 100 kΩ. These signals need to be driven to logic high for device operation.

Protection Circuits

The DRV8825 is fully protected against undervoltage, overcurrent and overtemperature events.

Overcurrent Protection (OCP)

An analog current limit circuit on each FET limits the current through the FET by removing the gate drive. If this analog current limit persists for longer than the OCP time, all FETs in the H-bridge will be disabled and the nFAULT pin will be driven low. The device will remain disabled until either nRESET pin is applied, or VM is removed and re-applied.

Copyright © 2010–2011, Texas Instruments Incorporated

Overcurrent conditions on both high and low side devices; i.e., a short to ground, supply, or across the motor winding will all result in an overcurrent shutdown. Note that overcurrent protection does not use the current sense circuitry used for PWM current control, and is independent of the I_{SENSE} resistor value or VREF voltage.

Thermal Shutdown (TSD)

If the die temperature exceeds safe limits, all FETs in the H-bridge will be disabled and the nFAULT pin will be driven low. Once the die temperature has fallen to a safe level operation will automatically resume.

Undervoltage Lockout (UVLO)

If at any time the voltage on the VM pins falls below the undervoltage lockout threshold voltage, all circuitry in the device will be disabled and internal logic will be reset. Operation will resume when V_M rises above the UVLO threshold.

Copyright © 2010–2011, Texas Instruments Incorporated

THERMAL INFORMATION

Thermal Protection

The DRV8825 has thermal shutdown (TSD) as described above. If the die temperature exceeds approximately 150°C, the device will be disabled until the temperature drops to a safe level.

Any tendency of the device to enter TSD is an indication of either excessive power dissipation, insufficient heatsinking, or too high an ambient temperature.

Power Dissipation

Power dissipation in the DRV8825 is dominated by the power dissipated in the output FET resistance, or $R_{DS(ON)}$. Average power dissipation when running a stepper motor can be roughly estimated by Equation 2.

$$P_{TOT} = 4 \bullet R_{DS(ON)} \bullet (I_{OUT(RMS)})^2 \quad (2)$$

where P_{TOT} is the total power dissipation, $R_{DS(ON)}$ is the resistance of each FET, and $I_{OUT(RMS)}$ is the RMS output current being applied to each winding. $I_{OUT(RMS)}$ is equal to the approximately 0.7x the full-scale output current setting. The factor of 4 comes from the fact that there are two motor windings, and at any instant two FETs are conducting winding current for each winding (one high-side and one low-side).

The maximum amount of power that can be dissipated in the device is dependent on ambient temperature and heatsinking.

Note that $R_{DS(ON)}$ increases with temperature, so as the device heats, the power dissipation increases. This must be taken into consideration when sizing the heatsink.

Heatsinking

The PowerPAD™ package uses an exposed pad to remove heat from the device. For proper operation, this pad must be thermally connected to copper on the PCB to dissipate heat. On a multi-layer PCB with a ground plane, this can be accomplished by adding a number of vias to connect the thermal pad to the ground plane. On PCBs without internal planes, copper area can be added on either side of the PCB to dissipate heat. If the copper area is on the opposite side of the PCB from the device, thermal vias are used to transfer the heat between top and bottom layers.

For details about how to design the PCB, refer to TI application report SLMA002, " PowerPAD™ Thermally Enhanced Package" and TI application brief SLMA004, " PowerPAD™ Made Easy", available at www.ti.com.

In general, the more copper area that can be provided, the more power can be dissipated. It can be seen that the heatsink effectiveness increases rapidly to about 20 cm^2, then levels off somewhat for larger areas.

Copyright © 2010–2011, Texas Instruments Incorporated

PACKAGING INFORMATION

Orderable Device	Status [1]	Package Type	Package Drawing	Pins	Package Qty	Eco Plan [2]	Lead/ Ball Finish	MSL Peak Temp [3]	Samples (Requires Login)
DRV8825PWP	ACTIVE	HTSSOP	PWP	28	50	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-3-260C-168 HR	
DRV8825PWPR	ACTIVE	HTSSOP	PWP	28	2000	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-3-260C-168 HR	

[1] The marketing status values are defined as follows:
ACTIVE: Product device recommended for new designs.
LIFEBUY: TI has announced that the device will be discontinued, and a lifetime-buy period is in effect.
NRND: Not recommended for new designs. Device is in production to support existing customers, but TI does not recommend using this part in a new design.
PREVIEW: Device has been announced but is not in production. Samples may or may not be available.
OBSOLETE: TI has discontinued the production of the device.

[2] Eco Plan - The planned eco-friendly classification: Pb-Free (RoHS), Pb-Free (RoHS Exempt), or Green (RoHS & no Sb/Br) - please check http://www.ti.com/productcontent for the latest availability information and additional product content details.
TBD: The Pb-Free/Green conversion plan has not been defined.
Pb-Free (RoHS): TI's terms "Lead-Free" or "Pb-Free" mean semiconductor products that are compatible with the current RoHS requirements for all 6 substances, including the requirement that lead not exceed 0.1% by weight in homogeneous materials. Where designed to be soldered at high temperatures, TI Pb-Free products are suitable for use in specified lead-free processes.
Pb-Free (RoHS Exempt): This component has a RoHS exemption for either 1) lead-based flip-chip solder bumps used between the die and package, or 2) lead-based die adhesive used between the die and leadframe. The component is otherwise considered Pb-Free (RoHS compatible) as defined above.
Green (RoHS & no Sb/Br): TI defines "Green" to mean Pb-Free (RoHS compatible), and free of Bromine (Br) and Antimony (Sb) based flame retardants (Br or Sb do not exceed 0.1% by weight in homogeneous material)

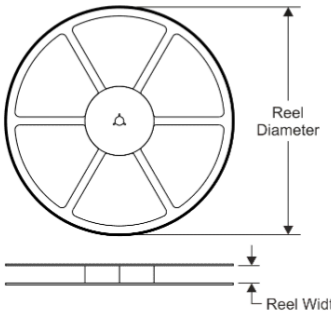
[3] MSL, Peak Temp. -- The Moisture Sensitivity Level rating according to the JEDEC industry standard classifications, and peak solder temperature.

Important Information and Disclaimer: The information provided on this page represents TI's knowledge and belief as of the date that it is provided. TI bases its knowledge and belief on information provided by third parties, and makes no representation or warranty as to the accuracy of such information. Efforts are underway to better integrate information from third parties. TI has taken and continues to take reasonable steps to provide representative and accurate information but may not have conducted destructive testing or chemical analysis on incoming materials and chemicals. TI and TI suppliers consider certain information to be proprietary, and thus CAS numbers and other limited information may not be available for release.

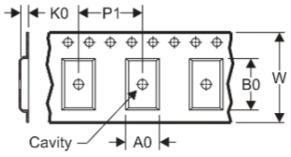
In no event shall TI's liability arising out of such information exceed the total purchase price of the TI part(s) at issue in this document sold by TI to Customer on an annual basis.

TAPE AND REEL INFORMATION

REEL DIMENSIONS

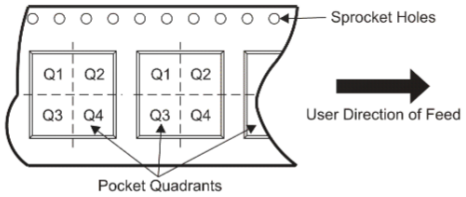


TAPE DIMENSIONS



A0	Dimension designed to accommodate the component width
B0	Dimension designed to accommodate the component length
K0	Dimension designed to accommodate the component thickness
W	Overall width of the carrier tape
P1	Pitch between successive cavity centers

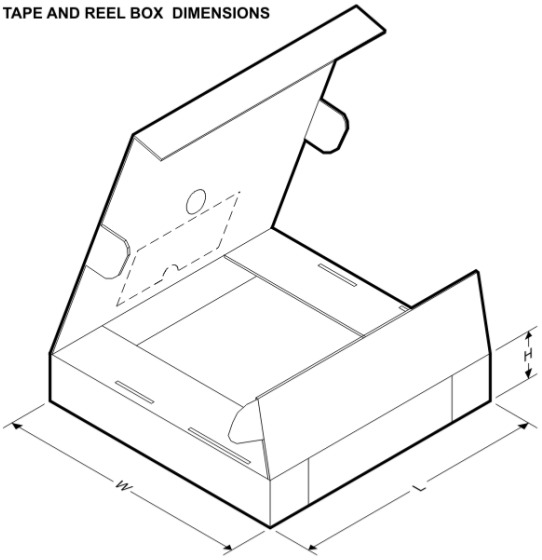
QUADRANT ASSIGNMENTS FOR PIN 1 ORIENTATION IN TAPE



*All dimensions are nominal

Device	Package Type	Package Drawing	Pins	SPQ	Reel Diameter (mm)	Reel Width W1 (mm)	A0 (mm)	B0 (mm)	K0 (mm)	P1 (mm)	W (mm)	Pin1 Quadrant
DRV8825PWPR	HTSSOP	PWP	28	2000	330.0	16.4	6.9	10.2	1.8	12.0	16.0	Q1

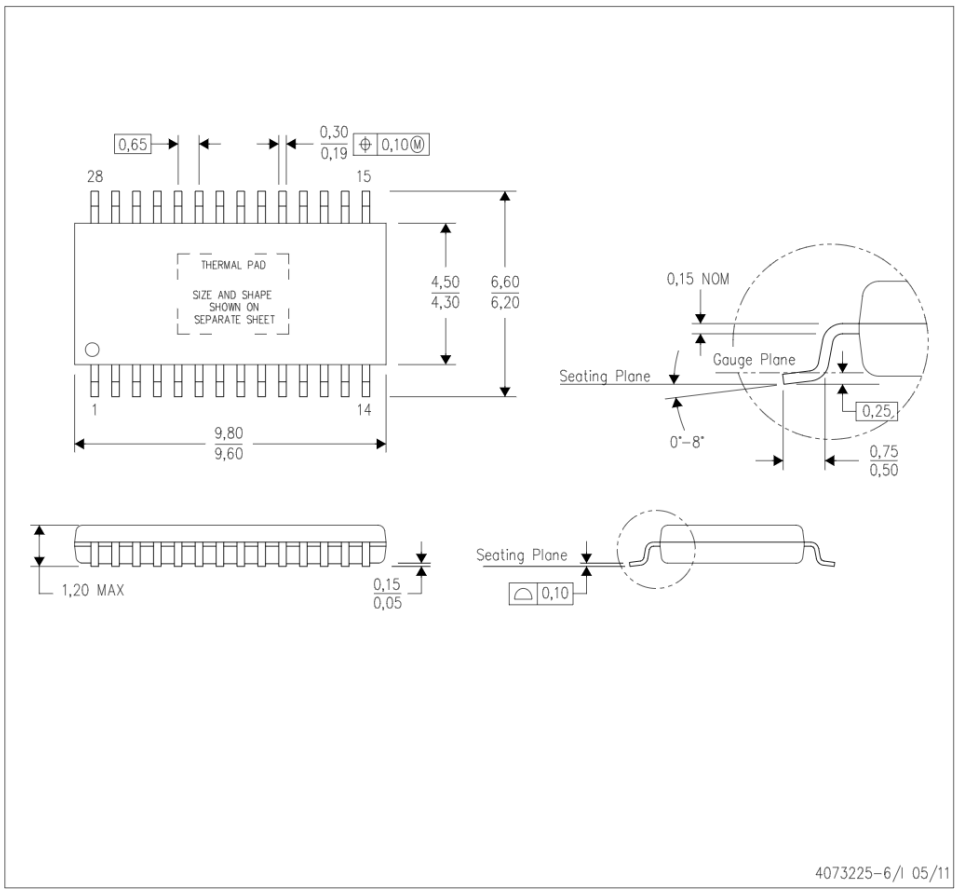
TAPE AND REEL BOX DIMENSIONS



*All dimensions are nominal

Device	Package Type	Package Drawing	Pins	SPQ	Length (mm)	Width (mm)	Height (mm)
DRV8825PWPR	HTSSOP	PWP	28	2000	346.0	346.0	33.0

PWP (R–PDSO–G28) PowerPAD™ PLASTIC SMALL OUTLINE



NOTES:
A. All linear dimensions are in millimeters.
B. This drawing is subject to change without notice.
C. Body dimensions do not include mold flash or protrusions. Mold flash and protrusion shall not exceed 0.15 per side.
D. This package is designed to be soldered to a thermal pad on the board. Refer to Technical Brief, PowerPad Thermally Enhanced Package, Texas Instruments Literature No. SLMA002 for information regarding recommended board layout. This document is available at www.ti.com <http://www.ti.com>.
E. See the additional figure in the Product Data Sheet for details regarding the exposed thermal pad features and dimensions.
E. Falls within JEDEC MO–153

PowerPAD is a trademark of Texas Instruments.

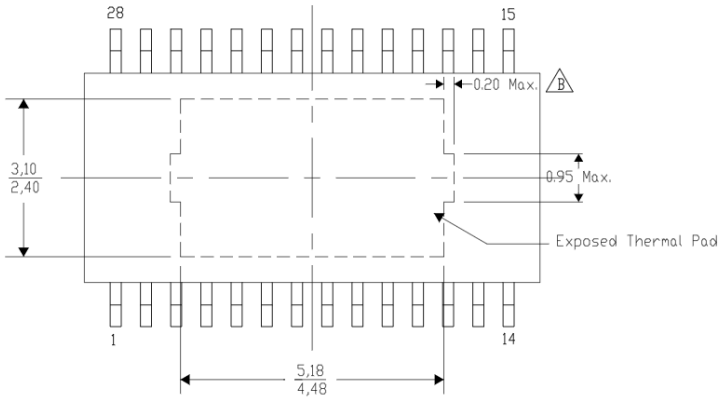
PWP (R–PDSO–G28) PowerPAD™ SMALL PLASTIC OUTLINE

THERMAL INFORMATION

This PowerPAD™ package incorporates an exposed thermal pad that is designed to be attached to a printed circuit board (PCB). The thermal pad must be soldered directly to the PCB. After soldering, the PCB can be used as a heatsink. In addition, through the use of thermal vias, the thermal pad can be attached directly to the appropriate copper plane shown in the electrical schematic for the device, or alternatively, can be attached to a special heatsink structure designed into the PCB. This design optimizes the heat transfer from the integrated circuit (IC).

For additional information on the PowerPAD package and how to take advantage of its heat dissipating abilities, refer to Technical Brief, PowerPAD Thermally Enhanced Package, Texas Instruments Literature No. SLMA002 and Application Brief, PowerPAD Made Easy, Texas Instruments Literature No. SLMA004. Both documents are available at www.ti.com.

The exposed thermal pad dimensions for this package are shown in the following illustration.



Top View

Exposed Thermal Pad Dimensions

4206332–28/W 05/11

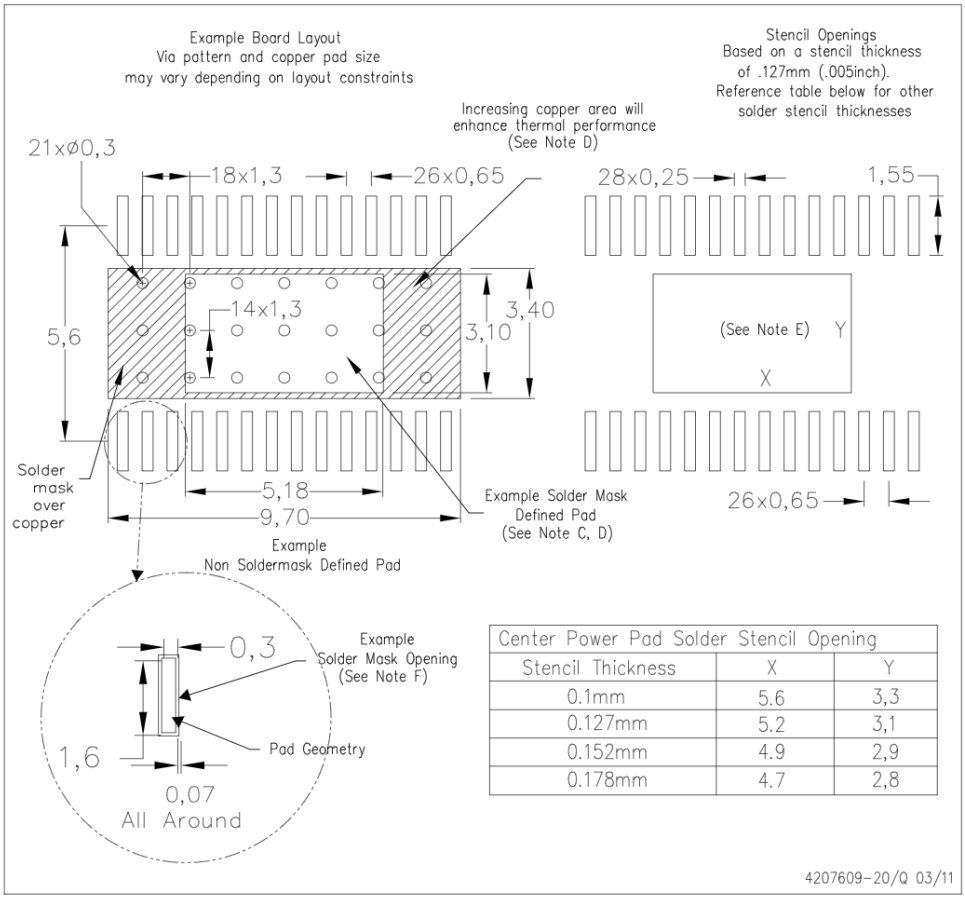
NOTE: A. All linear dimensions are in millimeters

B. Exposed tie strap features may not be present.

PowerPAD is a trademark of Texas Instruments

LAND PATTERN DATA

PWP (R–PDSO–G28) PowerPAD™ PLASTIC SMALL OUTLINE



Center Power Pad Solder Stencil Opening		
Stencil Thickness	X	Y
0.1mm	5.6	3,3
0.127mm	5.2	3,1
0.152mm	4.9	2,9
0.178mm	4.7	2,8

4207609–20/Q 03/11

NOTES:
A. All linear dimensions are in millimeters.
B. This drawing is subject to change without notice.
C. Customers should place a note on the circuit board fabrication drawing not to alter the center solder mask defined pad.
D. This package is designed to be soldered to a thermal pad on the board. Refer to Technical Brief, PowerPad Thermally Enhanced Package, Texas Instruments Literature No. SLMA002, SLMA004, and also the Product Data Sheets.
E. For specific thermal information, via requirements, and recommended board layout. These documents are available at www.ti.com <http://www.ti.com>. Publication IPC–7351 is recommended for alternate designs. Laser cutting apertures with trapezoidal walls and also rounding corners will offer better paste release. Customers should contact their board assembly site for stencil design recommendations. Example stencil design based on a 50% volumetric metal load solder paste. Refer to IPC–7525 for other stencil
F. Customers should contact their board fabrication site for solder mask tolerances between and around signal pads.

PowerPAD is a trademark of Texas Instruments.

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional restrictions.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

TI products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably be expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing such use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and acknowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products and any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be provided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in such safety-critical applications.

TI products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are specifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military specifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at the Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

TI products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are designated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated products in automotive applications, TI will not be responsible for any failure to meet such requirements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Audio	www.ti.com/audio	Communications and Telecom	www.ti.com/communications
Amplifiers	amplifier.ti.com	Computers and Peripherals	www.ti.com/computers
Data Converters	dataconverter.ti.com	Consumer Electronics	www.ti.com/consumer-apps
DLP® Products	www.dlp.com	Energy and Lighting	www.ti.com/energy
DSP	dsp.ti.com	Industrial	www.ti.com/industrial
Clocks and Timers	www.ti.com/clocks	Medical	www.ti.com/medical
Interface	interface.ti.com	Security	www.ti.com/security
Logic	logic.ti.com	Space, Avionics and Defense	www.ti.com/space-avionics-defense
Power Mgmt	power.ti.com	Transportation and Automotive	www.ti.com/automotive
Microcontrollers	microcontroller.ti.com	Video and Imaging	www.ti.com/video
RFID	www.ti-rfid.com	Wireless	www.ti.com/wireless-apps
RF/IF and ZigBee® Solutions	www.ti.com/lprf		

TI E2E Community Home Page e2e.ti.com

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2011, Texas Instruments Incorporated

AEDB-9140 Series

Three Channel Optical Incremental Encoder Modules with Codewheel, 100 CPR to 500 CPR

Data Sheet

Description

The AEDB-9140 series are three channel optical incremental encoder modules offered with a codewheel. When used with a codewheel, these low cost modules detect rotary position. Each module consists of a lensed LED source and a detector IC enclosed in a small plastic package. Due to a highly collimated light source and a unique photodetector array, these modules are extremely tolerant to mounting misalignment.

The AEDB-9140 has two channel quadrature outputs plus a third channel index output. This index output is a 90 electrical degree high true index pulse which is generated once for each full rotation of the codewheel.

The AEDB-9140 is designed for use with a codewheel which has an optical radius of 11.00 mm (0.433 inch).

The quadrature signals and the index pulse are accessed through five 0.46 mm square pins located on 1.27 mm(pitch) centers.

Features

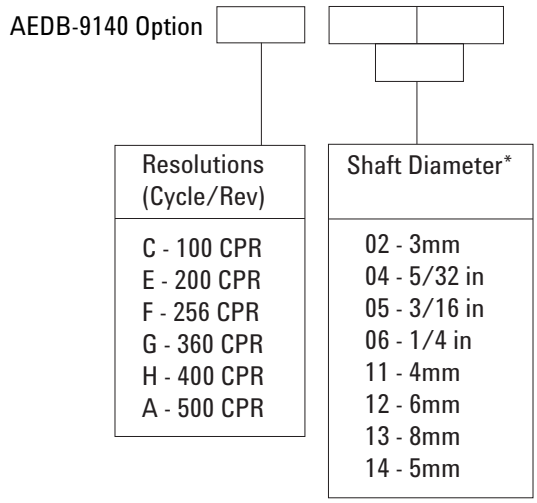
- Two Channel Quadrature Output with Index Pulse
- Resolution from 100 CPR to 500 CPR (Counts Per Revolution)
- Low Cost
- Easy to Mount
- No Signal Adjustment required
- Small Size
- Operating Temperature: -10°C to 85°C
- TTL Compatible
- Single 5V Supply

Applications

The AEDB-9140 provide sophisticated motion control detection at a low cost, making them ideal for high volume applications. Typical applications include printers, plotters, tape drives, and industrial and factory automation equipment.

Note: Avago Technologies encoders are not recommended for use in safety critical applications. Eg. ABS braking systems, power steering, life support systems and critical care medical equipment. Please contact sales representative if more clarification is needed.

Ordering Information



Three Channel Encoder Modules with Codewheel, 11 mm Optical Radius

* Please contact factory for other shaft diameters

Available Options

		Shaft Diameter Options							
Part No	**CPR**	**02**	**04**	**05**	**06**	**11**	**12**	**13**	**14**
AEDB-9140	C		•		•		•	•	
	E				•	•	•		•
	F		•				•		•
	G				•		•		•
	H				•				•
	A	•	•	•	•	•	•	•	•

Theory of Operation

The AEDB-9140 are emitter/detector modules. Coupled with a codewheel, these modules translate the rotary motion of a shaft into a three-channel digital output.

As seen in Figure 1, the modules contain a single Light Emitting Diode (LED) as its light source. The light is collimated into a parallel beam by means of a single polycarbonate lens located directly over the LED. Opposite the emitter is the integrated detector circuit. This IC consists of multiple sets of photodetectors and the signal processing circuitry necessary to produce the digital waveforms.

The codewheel rotates between the emitter and detector, causing the light beam to be interrupted by the pattern of spaces and bars on the codewheel.

The photodiodes which detect these interruptions are arranged in a pattern that corresponds to the radius and design of the code-wheel. These detectors are also spaced such that a light period on one pair of detectors corresponds to a dark period on the adjacent pair of detectors.

The photodiode outputs are then fed through the signal processing circuitry resulting in A, Abar, B, Bbar, I and Ibar. Comparators receive these signals and produce the final outputs for channels A and B. Due to this integrated phasing technique, the digital output of channel A is in quadrature with that of channel B (90 degrees out of phase).

Block Diagram

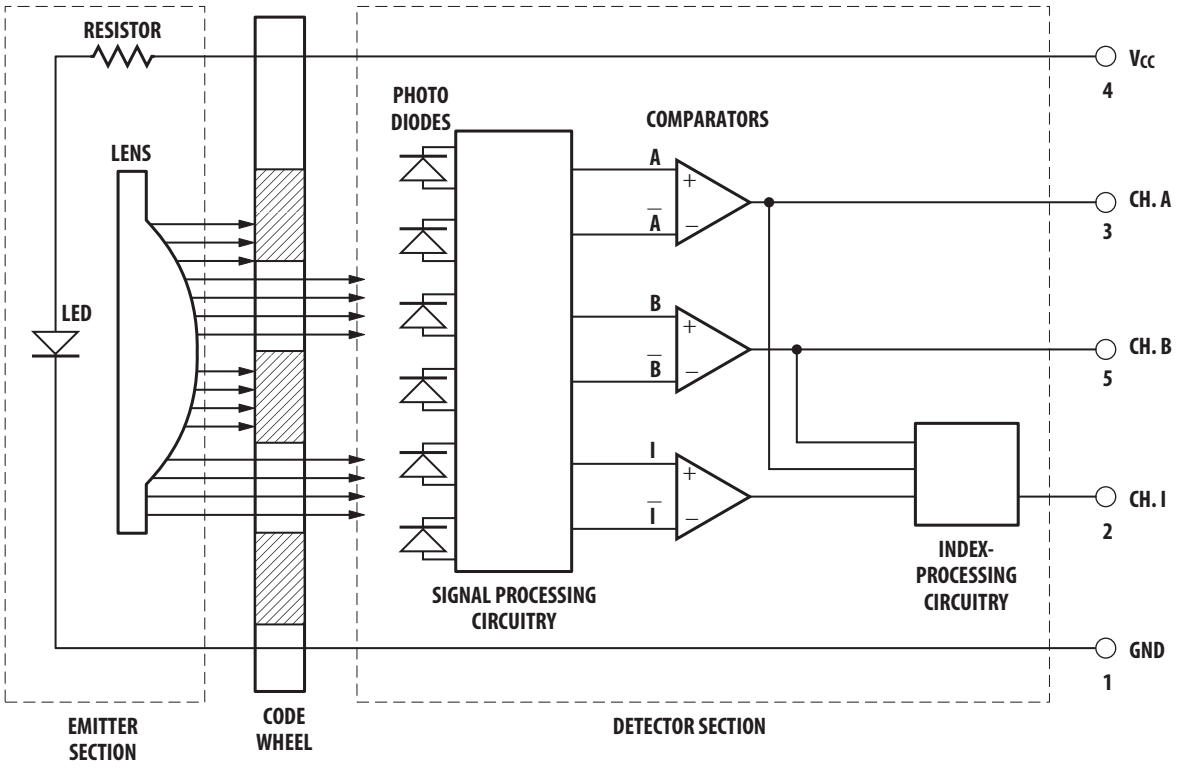


Figure 1.

Output Waveforms

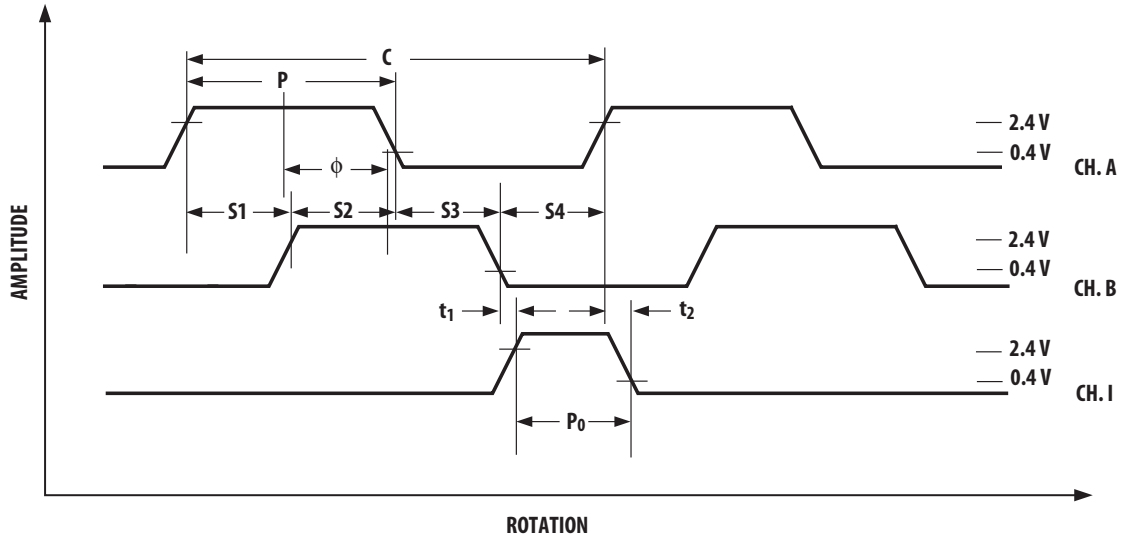


Figure 2.

Definitions

Note: Refer to Figure 2

Count (N): The number of bar and window pairs or counts per revolution (CPR) of the codewheel.

One Cycle (C): 360 electrical degrees (°e), 1 bar and window pair.

One Shaft Rotation: 360 mechanical degrees, N cycles.

Position Error ($\Delta\Theta$): The normalized angular difference between the actual shaft position and the position indicated by the encoder cycle count.

Cycle Error (ΔC): An indication of cycle uniformity. The difference between an observed shaft angle which gives rise to one electrical cycle, and the nominal angular increment of 1/N of a revolution.

Pulse Width (P): The number of electrical degrees that an output is high during 1 cycle. This value is nominally 180°e or 1/2 cycle.

Pulse Width Error (ΔP): The deviation, in electrical degrees, of the pulse width from its ideal value of 180°e.

State Width (S): The number of electrical degrees between a transition in the output of channel A and the neighboring transition in the output of channel B. There are 4 states per cycle, each nominally 90°e.

State Width Error (ΔS): The deviation, in electrical degrees, of each state width from its ideal value of 90°e.

Phase (f): The number of electrical degrees between the center of the high state of channel A and the center of the high state of channel B.

This value is nominally 90°e for quadrature output.

Phase Error ($\Delta\phi$): The deviation of the phase from its ideal value of 90°e.

Direction of Rotation: When the codewheel rotates in the clockwise direction viewing from top of the module (direction from V to G), channel A will lead channel B. If the codewheel rotates in the opposite direction, channel B will lead channel A.

Optical Radius (Rop): The distance from the codewheel's center of rotation to the optical center (O.C) of the encoder module.

Index Pulse Width (Po): The number of electrical degrees that an index is high during one full shaft rotation. This value is nominally 90°e or 1/4 cycle.

Absolute Maximum Ratings

Parameter	Symbol	Minimum	Typical	Maximum	Units	Notes
Storage Temperature	T_S	-10		85	°C	
Operating Temperature	T_A	-10		85	°C	
Supply Voltage	V_{CC}	-0.5		7	Volts	
Output Voltage	V_O	-0.5		V_{CC}	Volts	
Output Current per Channel, Iout	I_{OUT}	-1.0		18	mA	

Recommended Operating Conditions

Parameter	Symbol	Min.	Typ.	Max.	Units	Notes
Temperature	T_A	-10		85	°C	
Supply Voltage	V_{CC}	4.5	5.0	5.5	Volts	Ripple < 100mVp-p
Load Capacitance	C_L			100	pF	2.7 kΩ pull-up
Frequency	f			100	kHz	Velocity (rpm) x N/60
Shaft Perpendicularity Plus Axial Play				± 0.20(± 0.008)	mm(in.)	Refer to Mounting Consideration
Shaft Eccentricity Plus Radial Play				± 0.04(± 0.0015)	mm(in.)	

Electrical Characteristics

Electrical Characteristics Over the Recommended Operating Range. Typical Values at 25°C.

Parameter	Symbol	Minimum	Typical	Maximum	Units	Notes
Supply Current	I_{CC}	30	57	85	mA	
High Level Output Voltage	V_{OH}	2.4			V	Typ. I_{OH} = -0.5 mA
Low Level Output Voltage	V_{OL}			0.4	V	Typ. I_{OL} = 10 mA
Rise Time	t_r		180		ns	C_L = 25 pFR_L = 2.7 kΩ pull-up
Fall Time	t_f		50		ns	

Note: Typical values specified at Vcc = 5.0 V and 25 °C

Encoding Characteristics

Encoding Characteristics Over the Recommended Operating Conditions and Recommended Mounting Tolerances unless otherwise specified.

Parameter		Symbol	Minimum	Typical	Maximum	Units	Notes
Cycle Error		ΔC		3	10	°e	
Pulse Width Error		ΔP		7	30	°e	
Logic State Width Error		ΔS		5	30	°e	
Phase Error		Δϕ		2	15	°e	
Position Error		ΔΘ		10	40	min. of arc	
Index Pulse Width		Po	60	90	120	°e	
CH I rise afterCH B or CH A fall	-10°C to + 85°C	t_1	10	100	1000	ns	
CH I fall afterCH A or CH B rise	-10°C to + 85°C	t_2	10	300	1000	ns	

Electrical Interface

To insure reliable encoding performance, the AEDB-9140 three channel encoder modules require 2.7 kΩ (± 10%) pull-up resistors on output pins 2, 3, and 5 (Channels A, I and B) as shown in Figure 3. These pull-up resistors should be located as close to the encoder module as possible (within 4 feet). Each of the three encoder module outputs can drive a single TTL load in this configuration.

Customized Solutions

Customization of codewheel CPR is possible. It has to be based on the encoder LPI table given below:

Part Number	LPI
AEDB-9140 # C	36.7
AEDB-9140 # E	73.5
AEDB-9140 # F	94
AEDB-9140 # G	132.3
AEDB-9140 # H	147
AEDB-9140 # A	183

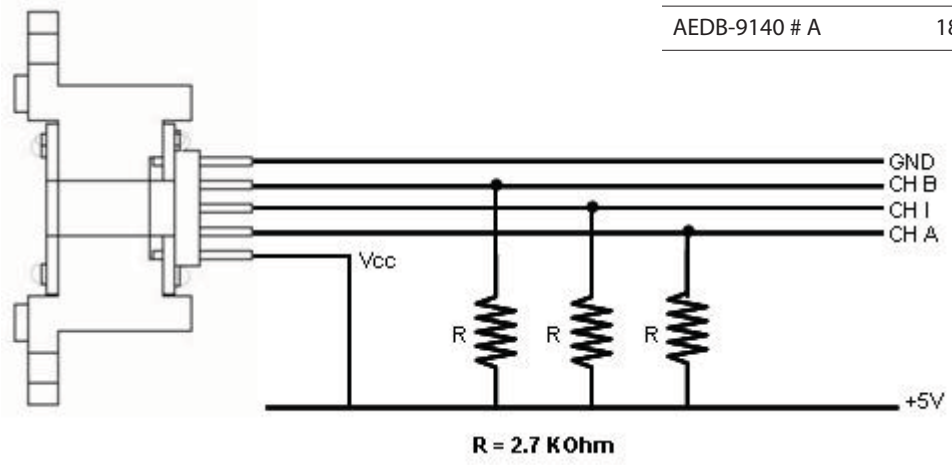


Figure 3.

CPR calculation formula:

CPR = (LPI x 25.4) x 2 x π x ROP

Where:

CPR = Counts Per Revolutions

LPI = Encoder LPI provided in the table

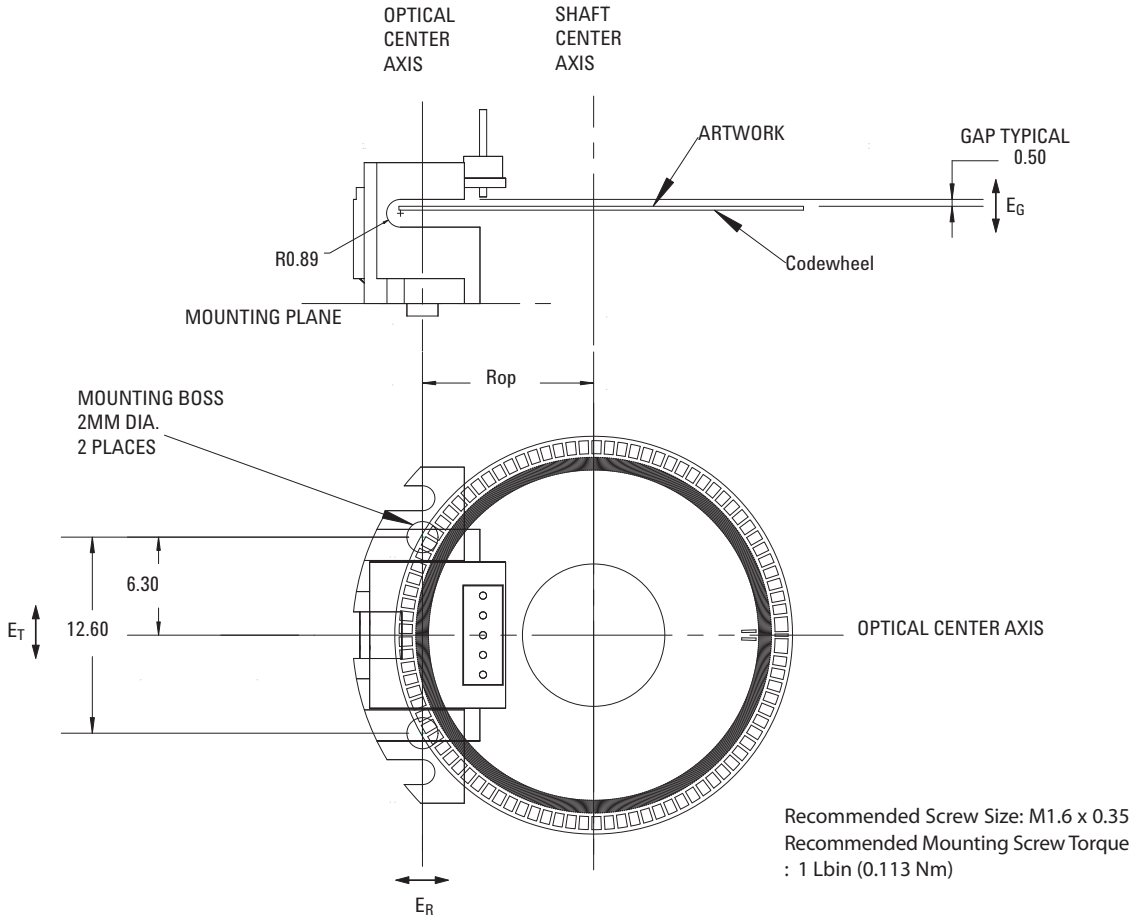
ROP = Encoder Optical Radius in mm

* Recommended maximum Codewheel diameter should not exceed 30mm

Note: The customization of codewheel method is valid from theoretical standpoint. However Avago Technologies strongly recommends a full characterization to be done to determine the actual performance of the encoder with customized codewheel.

Characterization means validating the encoding performance (consist of cycle error, pulse width error, logic state width error, phase error, position error & index pulse width , index channel rise and fall time) over the recommended operating conditions and recommended mounting tolerances.

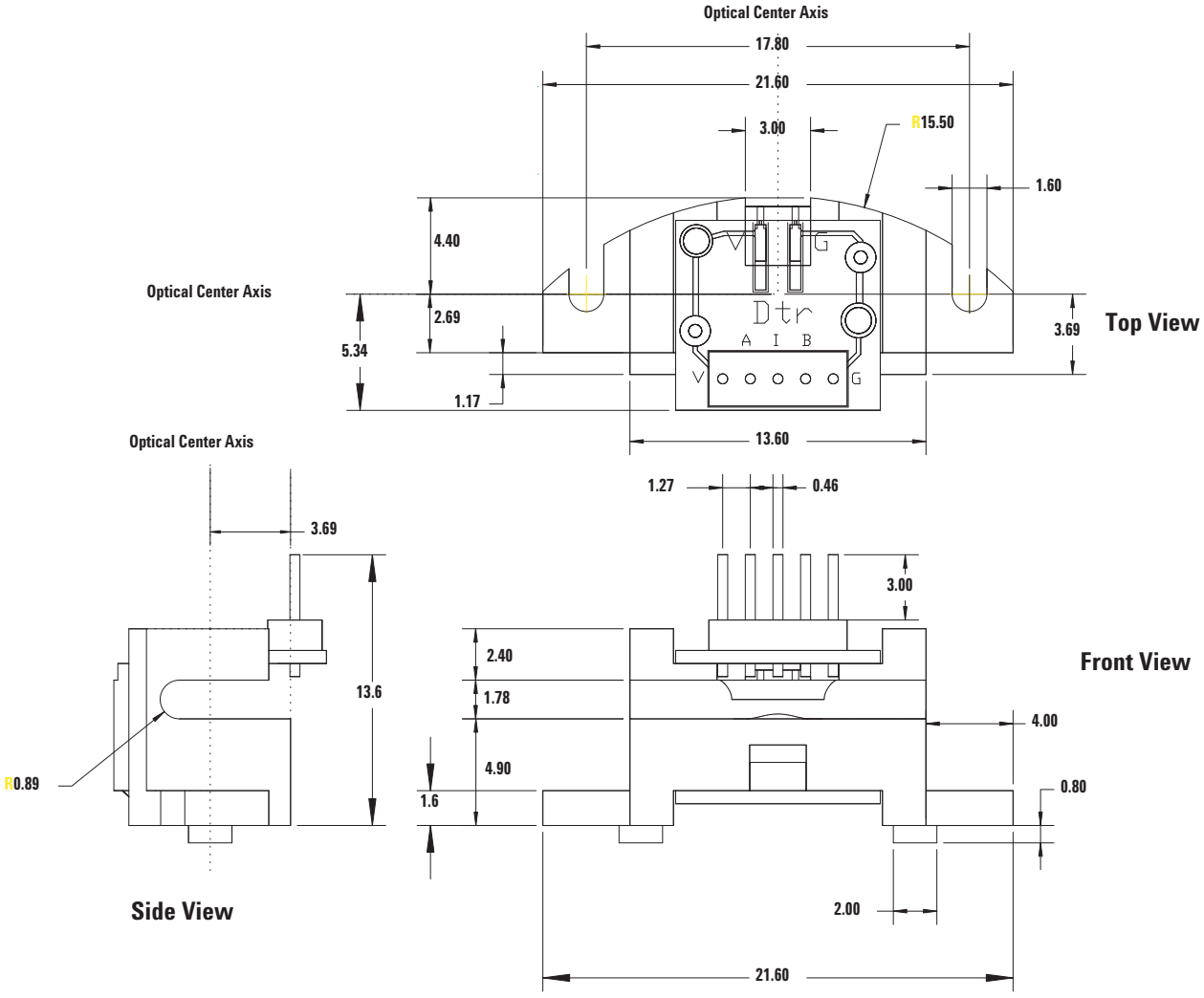
Mounting Considerations



Note:
These dimensions include shaft endplay and codewheel warp. All dimension for mounting the module and codewheel should be measured with respect to two mounting boss, as shown above.

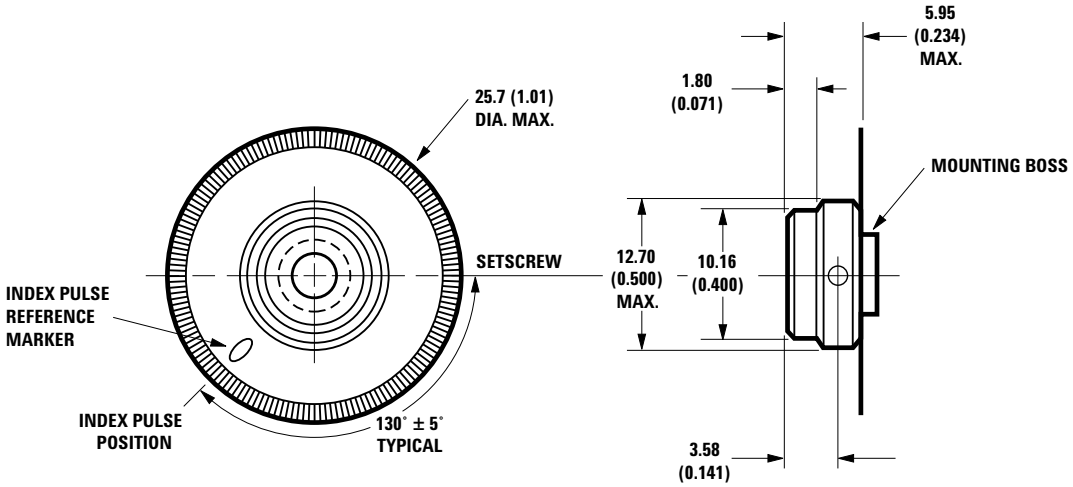
Error		Rop = 11mm	Unit	Notes
E_G	Gap	± 0.20	mm	Recommend to mount the codewheel closer to the detector side (upper side) for optimum encoder performance.
E_R	Radial	± 0.13	mm	
E_T	Tangential	± 0.13	mm	

Package Dimension



TYPICAL DIMENSIONS IN MILLIMETERS

Codewheel Mechanical Drawing



R_{OP} = 11.00 mm (0.433 in.)

DIMENSIONS IN MM (INCHES)

For product information and a complete list of distributors, please go to our web site: **www.avagotech.com**

Avago, Avago Technologies, and the A logo are trademarks of Avago Technologies in the United States and other countries.
Data subject to change. Copyright © 2005-2009 Avago Technologies. All rights reserved. Obsoletes 5989-3823EN
AV02-1584EN - Janaury 4, 2010

AVAGO
TECHNOLOGIES

ANEXO III

Anexo al Trabajo de Fin de Grado y Trabajo de Fin de Máster: Relación del trabajo con los Objetivos de Desarrollo Sostenible de la agenda 2030

Grado de relación del trabajo con los Objetivos de Desarrollo Sostenible (ODS).

Objetivos de Desarrollo Sostenibles	Alto	Medio	Bajo	No Procede
ODS 1. **Fin de la pobreza.**			X	
ODS 2. **Hambre cero.**			X	
ODS 3. **Salud y bienestar.**	X			
ODS 4. **Educación de calidad.**	X			
ODS 5. **Igualdad de género.**			X	
ODS 6. **Agua limpia y saneamiento.**		X		
ODS 7. **Energía asequible y no contaminante.**			X	
ODS 8. **Trabajo decente y crecimiento económico.**		X		
ODS 9. **Industria, innovación e infraestructuras.**	X			
ODS 10. **Reducción de las desigualdades.**			X	
ODS 11. **Ciudades y comunidades sostenibles.**			X	
ODS 12. **Producción y consumo responsables.**		X		
ODS 13. **Acción por el clima.**			X	
ODS 14. **Vida submarina.**				X
ODS 15. **Vida de ecosistemas terrestres.**				X
ODS 16. **Paz, justicia e instituciones sólidas.**				X
ODS 17. **Alianzas para lograr objetivos.**				X

Descripción de la alineación del TFG/TFM con los ODS con un grado de relación más alto.