AvaCient ISSN 2992-8567
Año 2, Núm. 2, Vol. IV, Julio Diciembre 2024 pp. 11-21

CONTROL INDUSTRIAL DE UN MOTOR A PASOS USANDO TECNOLOGÍA FRUGAL

Omar Trejoluna-Hernández [1], Felipe de Jesús Torres del Carmen[2*],
Ma. Concepción Alvarado[3], Miroslava Cano-Lara[4], Israel Martínez Ramírez[5].

ARTÍCULO DE INVESTIGACIÓN CIENTÍFICA
Recibido: 30/06/2024 Aceptado: 07/07/2024
https://doi.org/10.69823/avacient.v4n2a3

Resumen.- En este artículo se presenta el diseño, construcción y resultados experimentales de una plataforma para el control industrial de un motor a pasos Nema 34, el cual es controlado a través del *driver DQ860HA* por medio de la modulación del ancho de pulso que es generado por una minicomputadora Raspberry Pi, por lo que se requiere un código de programación que ejecute cíclicamente la cantidad de pulsos que deben ser enviados al controlador atendiendo el paso nominal de 400 pulsos por revolución. Además, se ha desarrollado una interfaz de usuario en donde se habilita la posibilidad de seleccionar algunas modalidades de operación del motor a pasos como la velocidad y sentido de giro, cantidad de pasos a desarrollar por el motor o bien, el número de revoluciones que girará el motor. Por tanto, el código de programación realizará los cálculos necesarios para generar la señal de pulsos que atiendan a los valores definidos por el usuario. Se ha utilizado lenguaje de programación Python para manejar el puerto de entrada/salida de la Raspberry Pi con resultados experimentales que muestran el correcto desempeño del trabajo desarrollado usando tecnología frugal aplicada en un equipo industrial.

Palabras Clave: minicomputadora, control, frugal, motor, interfaz.

INDUSTRIAL CONTROL OF A STEPPER MOTOR USING FRUGAL TECHNOLOGY

Abstract.- This paper presents the design, construction and experimental results of a platform for the industrial control of a Nema 34 stepper motor, which is controlled through the DQ860HA driver by means of the pulse width modulation that is generated by a Raspberry Pi minicomputer, so a programming code is required to cyclically execute the number of pulses that must be sent to the controller attending the nominal step of 400 pulses per revolution. In addition, a user interface has been developed to enable the possibility of selecting some operation modes of the stepper motor such as speed and direction of rotation, number of steps to be developed by the motor or the number of revolutions that the motor will rotate. Therefore, the programming code will perform the necessary calculus to generate the pulse signal that meets the values defined by the user. Python programming language has been used to manage the input/output port of the Raspberry Pi with experimental results that show the correct performance of the developed work using frugal technology applied to an industrial device.

Keywords: minicomputer, control, frugal, motor, interface.

Introducción

Actualmente, un concepto que está teniendo gran relevancia es la llamada Sociedad 5.0, término que apareció en Japón en 2016 como parte del 5to plan básico de ciencia y tecnología elaborado por el Despacho de Ciencia y Tecnología del gobierno japonés y, se refiere a un nuevo tipo de sociedad, donde la innovación en ciencia y tecnología ocupe un lugar prominente, cuyo objetivo consiste en alcanzar un balance social mientras se asegure un desarrollo económico (Salgues, 2018). Particularmente, en la Sociedad 5.0 se hace énfasis en restaurar la centralidad en el ser humano con

[1] Universidad de Guanajuato, División de Ingenierías Campus Irapuato-Salamanca, Carretera Federal Salamanca-Valle de Santiago km 3.5 + 1.8, Salamanca, Guanajuato, México, 36880. ORCID: 0009-0001-9614-5874
[2] Universidad de Guanajuato, División de Ingenierías Campus Irapuato-Salamanca, Carretera Federal Salamanca-Valle de Santiago km 3.5 + 1.8, Salamanca, Guanajuato, México, 36880. ORCID: 0000-0001-5792-2098, correo electrónico: fd.torres@ugto.mx, * **Autor corresponsal**.
[3] Tecnológico Nacional de México/Instituto Tecnológico Superior de San Martín Texmelucan, Camino a Barranca de Pesos S/N, San Martín Texmelucan, Puebla, México, 74120. ORCID: 0000-0002-4193-7452
[4] Tecnológico Nacional de México/Instituto Tecnológico Superior de Irapuato, Carretera Silao-Irapuato km 12.5, El Copal, Irapuato, Guanajuato, México, 36821. ORCID: 0000-0002-3335-2710
[5] Universidad de Guanajuato, División de Ingenierías Campus Irapuato-Salamanca, Carretera Federal Salamanca-Valle de Santiago km 3.5 + 1.8, Salamanca, Guanajuato, México, 36880. ORCID: 0000-0002-8186-4390

la utilización adecuada de la tecnología para el aprovechamiento de las oportunidades de innovación (Troisi et al., 2023).

Bajo este contexto, los desafíos que presentan actualmente distintos procesos industriales relacionados con el control y automatización se relacionan con la necesidad de utilizar tecnología de bajo costo, que, además, se posicionen como una opción eficiente para ser considerado por una gran mayoría, en particular, aquellos que no poseen los recursos económicos para adquirir equipo que se oferta en el mercado. Desde esta perspectiva, en los últimos años ha tenido un creciente auge entre la comunidad de ingeniería la aplicación de un concepto conocido como tecnología frugal, el cual refiere a medios y fines para hacer más con menos para muchos, es decir, hacer uso de la tecnología para obtener más de pocos recursos y así, lograr procesos o productos que sean accesibles a amplias poblaciones (Hossain et al., 2022).

Así, se pueden encontrar aplicaciones frugales en ingeniería en Sowinski et al., 2023, Kwon y Park, 2021, y Sanchez et al., 2023. Dentro de estas aplicaciones frugales, uno de los problemas atacados se ha focalizado en realizar simulaciones por computadora al menor costo posible, pero con un nivel de calidad aceptable (Jayabalan, 2021), por lo que se ha abordado desde dos perspectivas diferentes, una relacionada con el software de código abierto (OSS, por sus siglas en inglés) y otra con el uso de hardware de código abierto (OSH, por sus siglas en inglés). Esto debido a que la investigación y la enseñanza pueden vincularse más estrechamente a las arquitecturas de hardware del mundo real cuando se basan en sistemas OSH/OSS (Hannig y Teich, 2021).

Desde la perspectiva del OSS, Wajid et al., 2018 diseña un sistema operativo libre, fácilmente distribuible y personalizado basado en Ubuntu para ingeniería eléctrica/electrónica e informática; Tapaskar et al., 2018 destaca el empleo de herramientas OSS en los planes de estudio de ingeniería de posgrado; Lotfi et al., 2021 se centra en promover el uso de OSS como Python, GNU-Octave, Modelica, Java y Gazebo en la educación de ingeniería mecatrónica y robótica, donde se considera un motor de CD para mostrar la aplicación de OSS para el análisis y la simulación de modelos; Saluja et al., 2020 propone un sistema bajo un nuevo enfoque para el aprendizaje y la enseñanza de procesos de ingeniería de software a estudiantes de grado o postgrado; y Park, 2022 desarrolla un código automático GNU Octave con fines educativos para calcular la respuesta de un sistema de cadena cerrada multicuerpo; así mismo, en Choi et al., 2021, se hace especial hincapié en el uso de simulaciones en robótica donde se abordan algunas oportunidades en el desarrollo y validación de plataformas de simulación de código abierto.

En cuanto a la perspectiva de hardware de código abierto OSH, varios trabajos se han centrado en la sustitución de la computadora personal por una minicomputadora de placa única. Por ejemplo, Raikar et al., 2018 sugiere que la Raspberry Pi (RPi) se puede utilizar para la realización de prácticas de laboratorio dentro del plan de estudios de un programa de ingeniería informática; Alex David et al. 2018 diseña un laboratorio en línea acerca de un banco de pruebas de motores CD a través de una RPi para posibles aplicaciones en la enseñanza de la ingeniería eléctrica; Fernández et al., 2019 describe el desarrollo de un laboratorio remoto Arduino para soportar entornos de experimentación de aprendizaje IoT online conectado a RPi; Mbanisi et al. 2020 analiza las limitaciones y el potencial del hardware de código abierto como Arduino, RPi y BeagleBone para su uso en la enseñanza y la investigación en ingeniería; Fuentes et al., 2022 utiliza dispositivos RPi para animar a los estudiantes a trabajar de forma autónoma en las sesiones prácticas de los cursos de organización y diseño de computadoras; y Vaca et al. 2022 describe el diseño y desarrollo de un laboratorio remoto basado en RPi con aplicaciones en ingeniería electrónica y de control.
En aplicaciones frugales de ingeniería respecto al control de equipos o dispositivos, se requiere del uso tanto de software de código abierto como de hardware de código abierto para que sea considerada como una aplicación frugal. Por ejemplo, en Ramchandra et al., 2019 controlan un robot a través de una Raspberry Pi (RPi) y giroscopio ITG3200/ADXL345; Škraba et al., 2020 controla un motor de CD por medio de la señal de ancho de pulso PWM, en el cual utilizan tarjeta Arduino UNO y RPi 3B, y código en JavaScript; Karim y Thamrin, 2022 controla un servodriver PCA9685 con código Python sobre una Raspberry Pi para manipular los servomotores de un robot de 6 grados de libertad. Sin embargo, en todas estas aplicaciones frugales no se ha reportado que se realice en equipo o dispositivo de tipo industrial, lo cual es una recomendación de investigación relacionada con la tecnología OSH (Bonvoisin et al., 2021).

De esta manera, uno de los procesos industriales que son implementados de manera cotidiana, es el control de un motor a pasos, el cual es una máquina electromecánica que desarrolla un par mecánico y, por tanto, un desplazamiento angular, en función de la cantidad de pulsos eléctricos que se le envíen. Así, bajo el enfoque de tecnología frugal, se han reportado algunas aplicaciones frugales donde se ha utilizado una minicomputadora Raspberry Pi (RPi) para esta tarea de control; por ejemplo, en Khairudin et al., 2020 se describe el uso del driver TB6600 para controlar un motor

AvaCient ISSN 2992-8567
Año 2, Núm. 2, Vol. IV, Julio Diciembre 2024 pp. 11-21

a pasos Nema 23 en una máquina CNC a través de una RPi. Así mismo, en Ghani et al., 2021 se emplea una RPi 4 para manipular un motor a pasos Nema 34 para un sistema de asistencia de conducción; en Kriswanto, 2021 se desarrolla una máquina de empaquetado a través de una RPi y motor a pasos con el driver TB6600; en Emad et al, 2021 se usa el esquema RPi-Arduino-motor a pasos Nema 34 para hacer funcionar un robot delta; o bien, en Cámara, 2020 se realiza el control de un robot autoequilibrado basado en una RPi-Arduino-motor Nema 17. En estos trabajos se controla un motor a pasos de tipo industrial pero no se desarrolla una interfaz de usuario para seleccionar la modalidad de operación del motor, lo cual limita el funcionamiento deseado del sistema en su conjunto.

Por su parte, Fukumoto et al., 2020 propone un sistema de apoyo al aprendizaje basado en dispositivos móviles y de bajo costo para un motor a pasos que no es de tipo industrial, donde el sistema completo es concebido como un laboratorio remoto basado en la web utilizando una interfaz de usuario amigable; en la que se hace uso de dispositivos de bajo costo como RPi, dos plataformas de Arduino, controlador de medio puente SN754410 y lenguaje de programación Node.js. En este trabajo se ha diseñado una interfaz gráfica de usuario para interactuar con el motor a pasos, pero el motor no es de tipo industrial y en la interfaz, el usuario solo puede introducir el número de pulsos a desarrollar por el motor y el ancho del pulso. Aún más, en Yılmazlar et al., 2018 se presenta una interfaz de usuario programada con Microsoft Visual Studio que puede controlar en lazo abierto a motores a pasos 28 BYJ-48 que no son de tipo industrial, donde el usuario puede indicar el número de pasos, la dirección y la velocidad; se ha utilizado una RPi y circuitos integrados ULN2003A como drivers de los motores a pasos, que se resalta, no son de tipo industrial y con software que no es de código abierto.

Por todo lo anterior, la problemática consiste en que las aplicaciones de tecnología frugal en ingeniería se han desarrollado de manera separada bajo el enfoque de software de código abierto o del uso de hardware de código abierto; sin embargo, para un proceso de control basado en una aplicación de tecnología frugal, se debe incorporar tanto software como hardware de código abierto, lo cual es un desafío aún en la frontera del conocimiento delimitado por el control de un equipo de tipo industrial.

Por tanto, el objetivo de este artículo consiste en mostrar resultados experimentales de la aplicación de tecnología frugal tanto de software como de hardware de código abierto para el control de un motor a pasos de tipo industrial con la interacción a través de una interfaz gráfica de usuario que permita seleccionar la modalidad de funcionamiento del motor, sea a través del número de pulsos, velocidad del motor, sentido de giro, número de revoluciones, entre otras. Todo ello usando una minicomputadora Raspberry Pi y código de programación en Python para la implementación de los algoritmos de la puesta en marcha del motor a pasos de tipo industrial Nema 34 conducido por medio del driver DQ860HA. Esto bajo la hipótesis de que la tecnología frugal, basada en el uso tanto de software como de hardware de código abierto, sí puede ser implementada en procesos industriales que requieren de la interacción con el usuario como parte del desarrollo de una sociedad 5.0.

Materiales y métodos

- Motor a pasos Nema 34.

El motor a pasos trabaja bajo el principio de operación de una máquina eléctrica de CD, sin embargo, se le llama “a pasos” porque desarrolla un pequeño giro, conocido como paso, cada vez que recibe un pulso eléctrico. En particular, en este trabajo se utiliza un motor a pasos industrial, como se muestra en la Figura 1, de doble eje, Nema 34 (dimensión de la placa frontal de 3.4″ x 3.4″), con las características técnicas dadas en la Tabla 1.

Figura 1. Motor a pasos Nema 34 de tipo industrial.

Nota: imagen tomada de lionchip, 2024.

AvaCient ISSN 2992-8567
Año 2, Núm. 2, Vol. IV, Julio Diciembre 2024 pp. 11-21

Tabla 1. Características técnicas del motor a pasos Nema 34.

Parámetros	Valor
Ángulo de paso	1.8 °
Corriente por fase	3.5 A
Resistencia por fase	1.9 Ω
Inductancia por fase	22 mH
Par de retención	11.2 Nm
Par motor	115 kg/cm
Peso	5 kg

Nota: elaboración propia de los autores.

- Driver DQ860HA.

El DQ860HA de la Figura 2 es un controlador de motor a pasos híbrido de dos fases, diseñado para obtener el máximo rendimiento del motor. Este controlador se puede utilizar con motores que requieren una corriente de hasta 7.2 A para funcionar y cuenta con micro pasos de hasta $1/256$ pasos. Este controlador se utiliza normalmente en máquinas CNC, impresoras 3D, máquinas de corte por láser y otras máquinas de automatización. Las especificaciones técnicas del DQ860HA se detallan en la Tabla 2.

Figura 2. Driver DQ860HA para motor a pasos híbrido de dos fases.

Nota: imagen tomada de lionchip, 2024.

Tabla 2. Especificaciones técnicas del driver DQ860HA.

Parámetros	Valor
Voltaje de entrada	24 − 110 $V\ CD$ o 180 − 80 VCA
Corriente de entrada	< 6 A
Micro pasos	400 – 51200
Corriente de salida	2.1 – 7.2 $A\ pico$
Frecuencia del pulso	0 – 100 kHz
Temperatura ambiente en funcionamiento	0 – 70 °C

Nota: elaboración propia de los autores.

- Raspberry Pi 3B+.

La Raspberry Pi modelo 3B+ de la Figura 3, es una placa de desarrollo definida como una computadora en placa única, la cual es concebida como una minicomputadora por su tamaño similar al de una tarjeta de crédito, que soporta sistema operativo basado en Linux, particularmente Raspbian que es el recomendado por la fundación Raspberry Pi. De tal manera que en este trabajo se ha utilizado esta minicomputadora que permite manipular sus 40 pines de propósito general de entrada/salida (GPIO) para interactuar con el exterior a través de sus características técnicas como se detalla en Gamess y Hernandez, 2022; las cuales incluyen un procesador Broadcom BCM2837B0, Cortex-A53 (ARMv8) de

AvaCient ISSN 2992-8567
Año 2, Núm. 2, Vol. IV, Julio Diciembre 2024 pp. 11-21

64-bit SoC a una frecuencia de reloj de 1.4 GHz y 1 GB de memoria RAM, para implementar acciones de control industrial bajo con interacción con el usuario.

Figura 3. Raspberry Pi 3B+.

Nota: imagen tomada de raspberrypi.com, 2024.

- Python.

El lenguaje de programación Python es considerado como un lenguaje orientado a objetos que está teniendo gran auge en todo el mundo por ser pragmático en la estructura de la sintaxis utilizada. El sistema operativo Raspbian tiene precargadas plataformas de programación para código en Python, por lo que es un lenguaje que puede manipular los recursos de hardware de la Raspberry Pi.

- Metodología.

Actualmente, en la programación de la RPi a través de Python, se usa habitualmente librerías que han sido desarrolladas y compartidas en distintos foros dentro del internet. Sin embargo, en este trabajo se siguió la metodología clásica de la programación de equipos físicos a través del análisis y comprensión del funcionamiento del equipo a controlar de acuerdo con la hoja de datos del fabricante. Es importante notar que el motor a pasos Nema 34 es conducido por el driver DQ860HA, esto quiere decir que el algoritmo de programación debe estar basado en la hoja de datos que especifica cómo funciona el driver DQ860HA.

Una vez comprendido el funcionamiento del driver, se realizaron pruebas de la puesta en marcha del motor a pasos con acciones simples como el envío de una cantidad de pasos que asegurara un desplazamiento angular deseado. Es decir, de acuerdo con las especificaciones del driver y del motor, se analizó y calculó que el código de programación debería enviar por la RPi la cantidad de 400 pulsos hacia el driver DQ860HA para que este a su vez, permitiera que el motor a pasos gire 1 vuelta, sin embargo, para lograr este propósito se debe seguir una secuencia de programación que habilite al driver para que opere, así como el sentido de giro que se desee. Importante notar que la velocidad de rotación del motor está en función de la frecuencia con la cual se envía la señal de pulsos.

Después de haber puesto en marcha el motor, se inició con el diseño básico de una interfaz de usuario, a fin de que se pudiera interactuar con el motor a pasos y poder seleccionar alguna modalidad de operación. En este sentido se optó por solicitar al usuario el número de pulsos a enviar o el número de revoluciones y el sentido de giro a través de valores numéricos que el usuario introducía por medio del teclado. Se destaca que en esta primera etapa de la interacción con el usuario se mantenía una velocidad de rotación fija.

Habiendo logrado esta interacción básica con el usuario, se inició con el proceso de encontrar el algoritmo que permitiera modificar la frecuencia con la cual se envía la señal de pulsos al driver, esto daría como resultado el variar la velocidad de rotación del motor a pasos. Para esto, se tenía como variable de entrada un valor numérico de la velocidad de rotación deseada en revoluciones por minuto (rpm). Este valor debía ser convertido a ciclos por segundo y, posteriormente, dividirlo entre 2 que sería el número de pulsos (valor en alto) que la RPi debía enviar al driver en 1 segundo.

AvaCient ISSN 2992-8567
Año 2, Núm. 2, Vol. IV, Julio Diciembre 2024 pp. 11-21

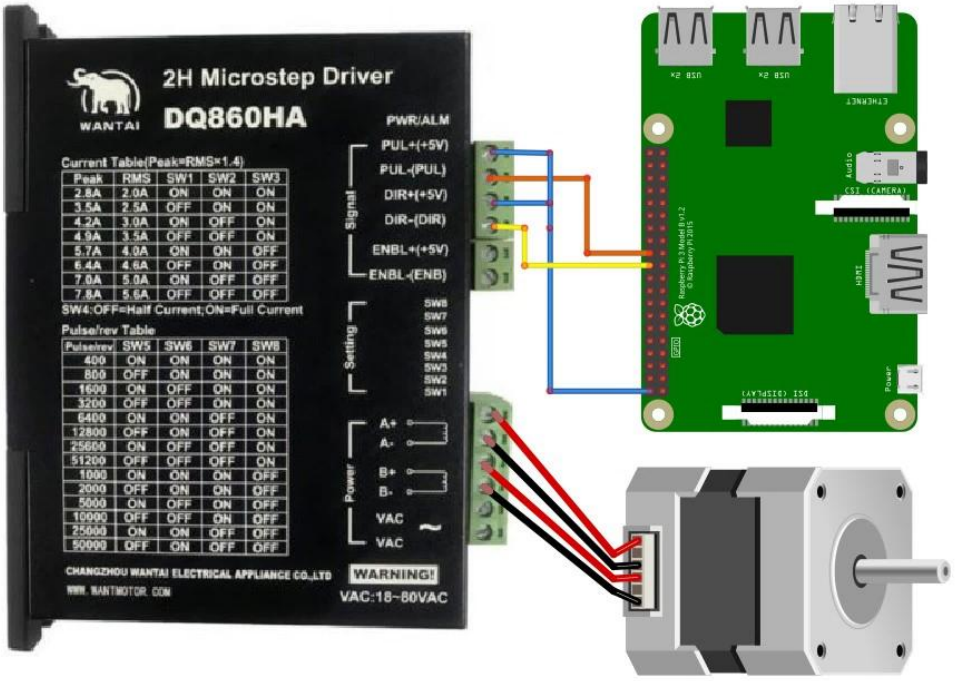
En este punto de la programación del código en Python, ya existía una interacción más interesante con el usuario, sin embargo, aún se tenía que conjuntar las versiones previas del código. Es decir, hacer funcionar al motor a pasos pero que el usuario seleccionara el sentido de giro, el número de pulsos o de revoluciones y la velocidad de rotación. De esta manera se implementaron contadores para el número de pulsos con el propósito de que el algoritmo no se quedara en un bucle infinito y al terminar, se enviara un mensaje al usuario de que había terminado con la operación solicitada y, además, el sistema estuviera habilitado para volver a enviar algún comando de operación indicado por el usuario.

Resultados

- Circuito de conexión.

Los dispositivos físicos que se han descrito anteriormente deben ser conectados entre sí de tal manera que la Raspberry Pi realice las funciones del control del motor a pasos a través del driver DQ860HA. Por lo que el circuito de conexión es el presentado en la Figura 4.

Figura 4. *Conexión entre los componentes.*



Nota: elaboración propia de los autores.

- Algoritmo de programación

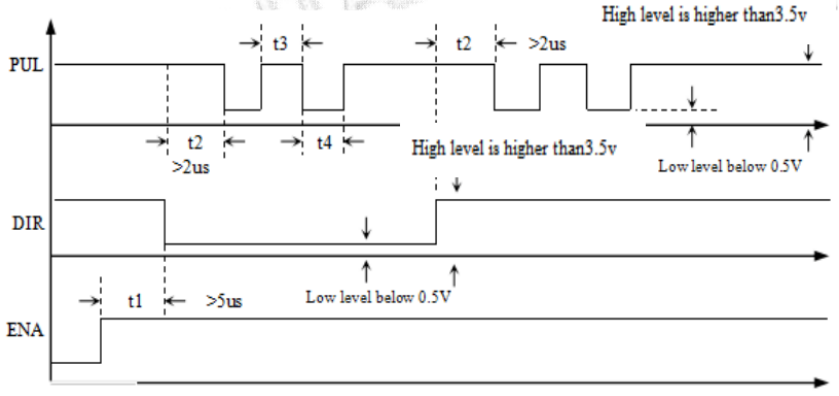
El proceso industrial que se ha implementado requiere de código de programación que se ha realizado usando el lenguaje Pyhton sobre una Raspberry Pi. Este código, principalmente consiste en el control de la puesta en marcha del motor a pasos, el cual es configurado por el usuario a través de una interfaz que permite la interacción con el dispositivo. Por lo cual, se identificaron las características técnicas del motor a pasos y del driver DQ860HA que se relacionen con modalidades de operación que el usuario pueda seleccionar o configurar y así, tener la interacción deseada.

En la Figura 5 se muestra la secuencia de señal de control por medio de pulsos que la Raspberry Pi debe enviar al driver DQ860HA para que este a su vez, pueda hacer girar al motor a pasos. Como puede observarse, la Raspberry Pi se conecta con el driver DQ860HA por medio de pines configurados como salida de la RPi para enviar las señales DIR y PUL. Posteriormente debe enviarse la señal DIR para indicar el sentido de giro (horario o antihorario), después se inicia el envío de la señal de pulsos PUL, la cual es generada por la Raspberry Pi ejecutando un ciclo que envía un valor alto durante cierto tiempo y luego envía un valor bajo (0 VCD) durante otro tiempo. Cada vez que el driver DQ860HA recibe un pulso, hace girar al motor a pasos y se requiere de 400 pulsos para dar una vuelta completa. De acuerdo con la frecuencia máxima del pulso generado por el driver de 100 kHz y, además, se requiere un mínimo de 400 pulsos para que el motor a pasos gire 1 revolución completa, se puede calcular que la velocidad máxima que

AvaCient ISSN 2992-8567
Año 2, Núm. 2, Vol. IV, Julio Diciembre 2024 pp. 11-21

alcanzaría el motor para cualquiera de los micropasos que sean especificados en el driver, es de 150 revoluciones por minuto (rpm).

Figura 5. Diagrama de secuencia de señal de control que recibe el driver desde la Raspberry Pi.



Nota: imagen tomada de la hoja de datos del driver DQ860HA.

Así, el pseudocódigo implementado en la Raspberry Pi es el siguiente:

1. Importar la librería *RPi.GPIO* para controlar los pines *GPIO* de la Raspberry Pi.
2. Importar la librería *time* para gestionar la modulación por ancho de pulso.
3. Desactivar las advertencias de *GPIO* para evitar mensajes no deseados.
4. Configurar el modo de los pines *GPIO* como *BCM*, donde el *GPIO 15* es usado para la señal *DIR* y el *GPIO 14* es usado para la señal *PUL*.
5. Iniciar un bucle:
   - 5.1. Declarar las variables *spin* (sentido de giro), *rev* (RPM del motor), *pul* (cantidad de pulsos a mandar por el GPIO 14), *sel* (selección entre cantidad de revoluciones o de pulsos). El usuario introduce los valores que desee a estas variables a través de la interfaz de usuario de la siguiente manera.
     - 5.1.1. Solicitar al usuario la velocidad angular deseada ω para el motor (en un rango permitido de 0-150 rpm).
     - 5.1.2. Verificar si la velocidad ingresada está dentro del rango permitido, de no ser así se mostrará un mensaje de error.
     - 5.1.3. Solicitar al usuario el sentido de giro del motor verificando que elija únicamente una de las dos opciones (horario o antihorario).
     - 5.1.4. Configurar el sentido de giro del motor según la selección del usuario a través del envío de la señal *DIR*.
     - 5.1.5. Solicitar al usuario si desea especificar el número de revoluciones o de pulsos verificando que elija únicamente una de las dos opciones.
     - 5.1.6. Calcular el período *T* de la señal de pulsos en función de la velocidad ingresada.

$$T = \frac{3}{20 \cdot \omega}$$

     - 5.1.7. Si se elige especificar el número de revoluciones:
       - 5.1.7.1. Solicitar al usuario el número de revoluciones deseadas validando que sea un valor mayor a cero.
       - 5.1.7.2. Calcular el número de pulsos requeridos para las revoluciones ingresadas (400 pulsos por revolución).
       - 5.1.7.3. Generar los pulsos necesarios para completar las revoluciones:
         - 5.1.7.3.1. Configurar el pin *GPIO 15* como alto para manejar la señal de pulsos *PUL* e iniciar el pulso.
         - 5.1.7.3.2. Esperar la mitad del periodo ($T/2$) para mantener el estado alto.
         - 5.1.7.3.3. Configurar el pin *GPIO 15* como bajo para manejar la señal de pulsos *PUL* y finalizar el pulso.
         - 5.1.7.3.4. Esperar la mitad del periodo ($T/2$) para mantener el estado bajo.
     - 5.1.8. Si se elige especificar el número de pulsos:
       - 5.1.8.1. Solicitar al usuario el número de pulsos deseados validando que sea un valor mayor a cero.
       - 5.1.8.2. Generar los pulsos determinados por el usuario.

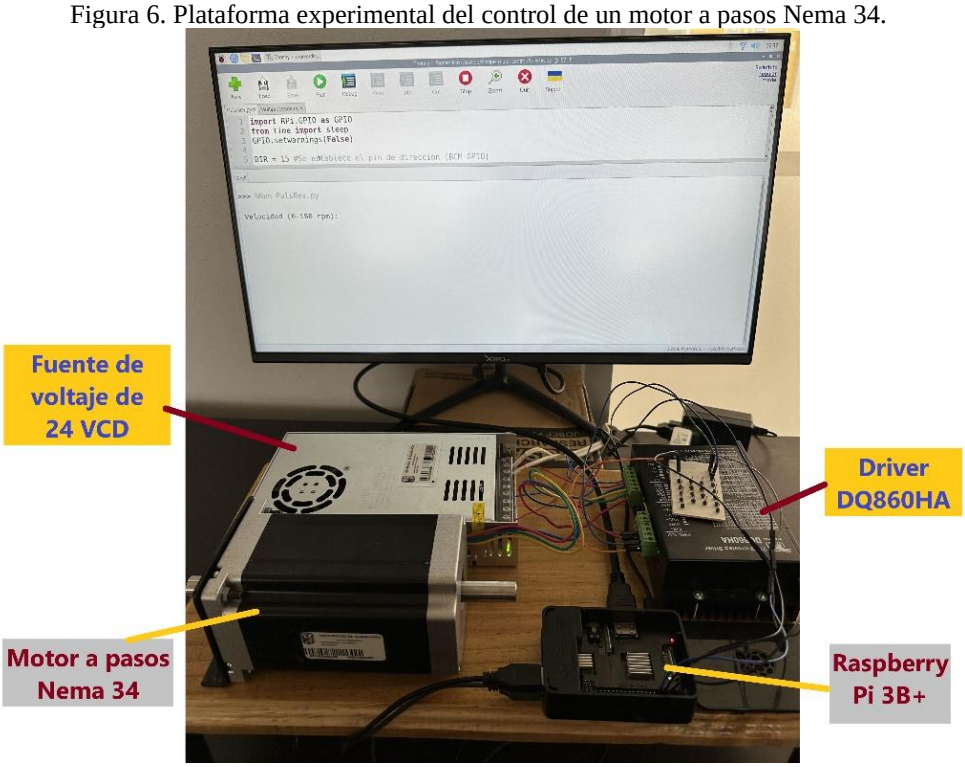
AvaCient ISSN 2992-8567
Año 2, Núm. 2, Vol. IV, Julio Diciembre 2024 pp. 11-21

5.2. Mostrar en pantalla un mensaje de haber concluido el proceso indicado por el usuario.
5.3. Regresar al paso 5.1.1 para reiniciar una nueva configuración de control del motor a pasos.

- Plataforma experimental

Para propósitos de dar evidencia con resultados experimentales del enfoque propuesto, se ha diseñado y construido una plataforma donde se han colocado los dispositivos descritos en la sección anterior siguiendo la conexión entre ellos como lo refiere la Figura 4. En la Figura 6 se muestra la apariencia física de la plataforma experimental, donde a través del teclado el usuario interactúa con el dispositivo físico.

Figura 6. Plataforma experimental del control de un motor a pasos Nema 34.



Nota: elaboración propia de los autores.

Discusión

En este trabajo se ha logrado el control de un motor a pasos Nema 34 de tipo industrial a través de la aplicación de tecnología frugal tanto de software de código abierto como de hardware de código abierto, obteniendo un sistema de control de bajo costo y con interacción con el usuario que puede configurar las características de operación del motor a pasos, como se muestra en la Figura 7a.

En comparación con resultados reportados en la literatura, relacionados directamente con el trabajo realizado, es importante destacar que en Yılmazlar et al., 2018 se ha reportado el diseño de una interfaz de usuario en un software bajo licencia, y el motor a pasos que se controla no es de tipo industrial. Respecto a los resultados reportados en Khairudin et al., 2020, Ghani et al., 2021, Kriswanto, 2021, Emad et al, 2021 y Cámara, 2020; en estos trabajos se desarrolla el control de un motor a pasos de tipo industrial, pero no se ha diseñado una interfaz de usuario como en el trabajo que se ha descrito en este artículo. Por último, en un trabajo más similar al realizado en este documento, Fukumoto et al., 2020 controla un motor a pasos que no es de tipo industrial y la interfaz de usuario solo permite interactuar con el número de pulsos y el ancho del pulso.

Por lo que en este trabajo que se ha desarrollado se han conjuntado ambas características de la aplicación de tecnología frugal en ingeniería, uso de software y hardware de código abierto, haciendo énfasis en la interacción que se logra con el usuario, como se destaca en la Figura 7b. Además, el usuario tiene la posibilidad de realizar una nueva configuración del control del motor a pasos cada vez que finaliza una ejecución. En las Figuras 7a y 7b se resalta que el usuario

AvaCient ISSN 2992-8567
Año 2, Núm. 2, Vol. IV, Julio Diciembre 2024 pp. 11-21

cambia el sentido de giro, además de que el control que se ha codificado permite que el usuario seleccione la cantidad de revoluciones que deberá girar el motor. El video de la ejecución de este control del motor a pasos puede ser consultado en el enlace:
https://drive.google.com/file/d/1dAR19Xr6PmPjwwAagcdrjSw8MNO4Ewmf/view?usp=drive_link

El correcto funcionamiento de la plataforma experimental que se ha logrado permitirá llevar a cabo prácticas de laboratorio en cursos de licenciatura relacionados con el control industrial, más aún, al ser una plataforma abierta, es posible servir como base de investigación y desarrollo tecnológico donde se pueda implementar herramientas de las tecnologías de la industria 4.0, entre otras.

Figura 7a. Usuario interactuando con el control del motor a pasos.

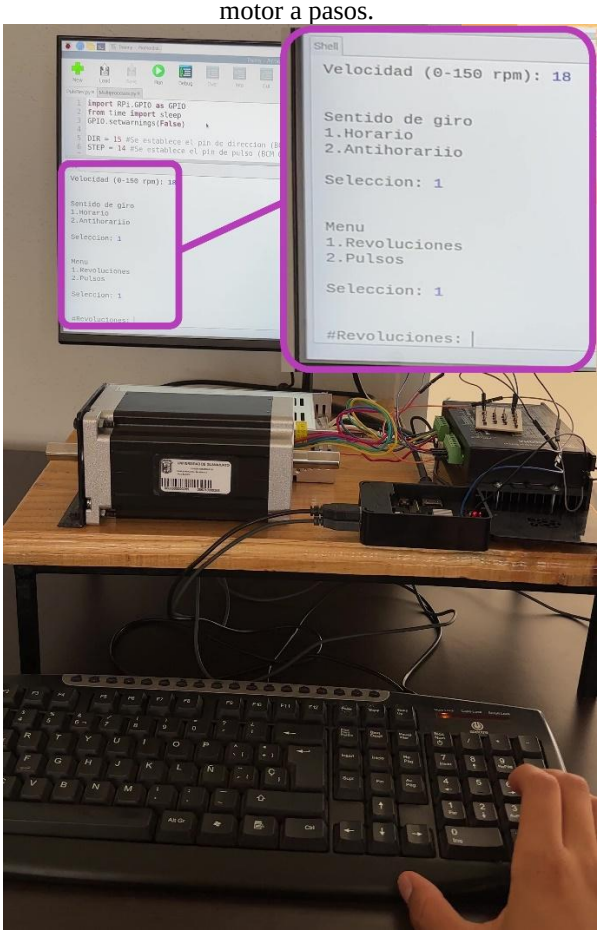
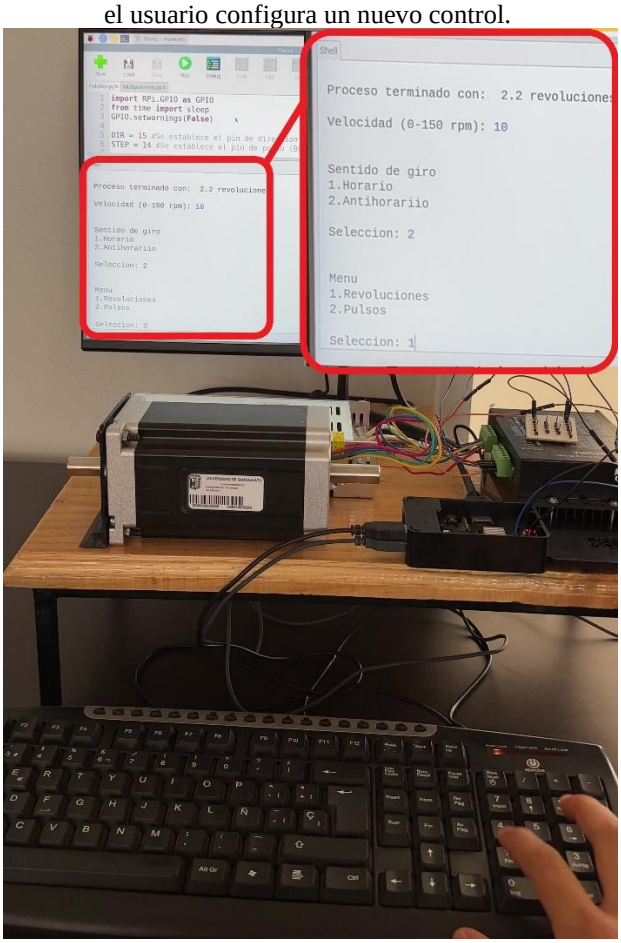


Figura 7b. El primer proceso de control ha concluido y el usuario configura un nuevo control.



Nota: elaboración propia de los autores.

Conclusiones

El control de un motor a pasos de tipo industrial Nema 34, donde se habilita al usuario la posibilidad de configurar la modalidad del control y realizarlo de manera cíclica, es posible implementarlo por medio de la aplicación de tecnología frugal de bajo costo, es decir, a través de software y hardware de código abierto como lo es el lenguaje de programación Python en una minicomputadora Raspberry Pi. Se ha construido una plataforma experimental que permite el control del motor a pasos de tipo industrial con interacción con el usuario que puede repetir la modalidad de control que desee cada vez que finaliza la ejecución del control dado. Esta plataforma experimental habilita la posibilidad de ser usada como equipo de laboratorio en la educación de la ingeniería en unidades de aprendizaje relacionadas con el control industrial. En trabajos futuros se implementarán acciones de monitoreo y registro de datos con la finalidad de que se complemente un sistema de control, adquisición y almacenamiento de datos, comúnmente conocido como sistema SCADA, por sus siglas en inglés.

Agradecimientos
Los autores desean agradecer a la Universidad de Guanajuato por el apoyo brindado para la realización de este trabajo a través de la Convocatoria Institucional de Investigación Científica 2024.

Referencias bibliográficas

Alex David, S.; Ravikumar, S.; Rizwana Parveen, A. Raspberry Pi in computer science and engineering education. In Intelligent Embedded Systems; *Springer*, 2018; pp. 11–16.

Bonvoisin, J., Mies, R., y Boujut, J. F. (2021). Seven observations and research questions about Open Design and Open Source Hardware. *Design Science*, 7, pp. e22.

Cámara, J. I. (2020). Diseño, realización y control de un robot autoequilibrado de bajo coste basado en una Raspberry Pi. Escuela Técnica Superior de Ingeniería, *Universidad de Sevilla*.

Choi, H.; Crump, C.; Duriez, C.; Elmquist, A.; Hager, G.; Han, D.; Hearl, F.; Hodgins, J.; Jain, A.; Leve, F.; et al. On the use of simulation in robotics: Opportunities, challenges, and suggestions for moving forward. *Proceedings of the National Academy of Sciences*, 2021(118).

DQ860HA Controlador Drive Nema 17, 23, 34 7.2A. (s.f.). lionchip. Recuperado el 15 de mayo de 2024, de https://www.lionchipmexico.com/product-page/controlador-drive-motor-a-pasos-nema-17-23-34-d-860-driver

Emad, E., Alaa, O., Hossam, M., Ashraf, M., y Shamseldin, M. A. (2021). Design and implementation of a low-cost microcontroller-based an industrial delta robot. *WSEAS Transactions on Computers*, 20, pp. 289-300.

Fernández-Pacheco, A.; Martin, S.; Castro, M. Implementation of an Arduino remote laboratory with raspberry Pi. In *Proceedings of the 2019 IEEE Global Engineering Education Conference (EDUCON)*. IEEE, 2019, pp. 1415–1418.

Fuentes, P.; Camarero, C.; Herreros, D.; Mateev, V.; Vallejo, F.; Martinez, C. Addressing Student Fatigue in Computer Architecture Courses. *IEEE Transactions on Learning Technologies*, 2022.

Fukumoto, H., Yamaguchi, T., Ishibashi, M., y Furukawa, T. (2020). Developing a remote laboratory system of stepper motor for learning support. *IEEE Transactions on Education*, 64(3), 292-298.

Gamess, E., y Hernandez, S. (2022). Performance evaluation of different Raspberry Pi models for a broad spectrum of interests. *International Journal of Advanced computer science and applications*, 13(2).

Ghani, S. A. C., Kettner, M., Aslan, F., Dzharuddin, F., y Sazali, N. (2021). Smart driving assistance system using Raspberry Pi and actuator networks. *In IOP Conference Series: Materials Science and Engineering*, 1068(1), pp. 012022.

Hannig, F., y Teich, J. (2021). Open source hardware. *Computer*, 54(10), pp. 111-115.

Hossain, M., Agarwal, N., Bhatti, Y., y Levänen, J. (2022). Frugal innovation: Antecedents, mediators, and consequences. *Creativity and Innovation Management*, 31(3), 521-540.

Jayabalan, J.; Dorasamy, M.; Raman, M. Reshaping higher educational institutions through frugal open innovation. *Journal of Open Innovation: Technology, Market, and Complexity*, 2021(7), pp. 145.

Karim, M. Z. B. A., y Thamrin, N. M. (2022). Servo Motor Controller using PID and Graphical User Interface on Raspberry Pi for Robotic Arm. *In Journal of Physics: Conference Series*, 2319(1), p. 012015. IOP Publishing

Khairudin, M., Asnawi, R., y Shah, A. (2020). The characteristics of TB6600 motor driver in producing optimal movement for the Nema23 stepper motor on CNC machine. *Telkomnika*, 18(1), 343-350.

Kriswanto, K., Karsan, K., Al-Janan, D. H., Roziqin, A., Hangga, A., Fathoni, K., y Wijayanto, B. (2021). Design and performance of the raspberry pi control system on packaging machine capacity 2400 Pcs/h. Rekayasa: *Jurnal Penerapan Teknologi Dan Pembelajaran*, 19(1), pp. 1-10.

Kwon, J.; Park, D. Hardware/software co-design for tinyml voice-recognition application on resource frugal Edge Devices. *Applied Sciences*, 2021(11), pp. 11073.

Lotfi, N.; Auslander, D.; Rodriguez, L.A.; Mbanisi, K.C.; Berry, C.A. Use of Open-source Software in Mechatronics and Robotics Engineering Education–Part I: Model Simulation and Analysis. *Computers in Education Journal*, 2021(12).

Mbanisi, K.C.; Auslander, D.M.; Berry, C.A.; Rodriguez, L.A.; Molki, M.; Lotfi, N. Promoting Open-source Hardware and Software Platforms in Mechatronics and Robotics Engineering Education. *In Proceedings of the 2020 ASEE Virtual Annual Conference Content Access*, 2020.

Nema 34 Motor a Pasos 115Kg/cm Doble Eje. (s.f.). Lionchip. Recuperado el 15 de mayo de 2024, de https://www.lionchipmexico.com/product-page/motor-de-paso-stepper-nema-34-cnc-impresora-3d-doble-eje-lionchip

Park, Y. Development of an Educational Code of Deriving Equations of Motion and Analyzing Dynamic Characteristics of Multibody Closed Chain Systems using GNU Octave for a Beginner. *Journal of Applied and Computational Mechanics*, 2022(8), pp. 232–244.

Raikar, M.M.; Desai, P.; Vijayalakshmi, M.; Narayankar, P. Upsurge of IoT (Internet of Things) in engineering education: A case study. *In Proceedings of the 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI*). IEEE, 2018, pp. 191–197.

Ramchandra, Y., Salim, M., Ankush, T., y Vasantrao, P. (2021). Self-balancing robot using Raspberry Pi and PID controller. *International Journal of Innovative Science and Research Technology*, 6(4), 321-322.

Raspberry Pi 3B+. (s.f.). Raspberrypi.com. Recuperado el 15 de mayo de 2024, de https://www.raspberrypi.com/products/raspberry-pi-3-model-b-plus/

Salgues, B. (2018). Society 5.0: industry of the future, technologies, methods and tools. John Wiley & Sons.

Saluja, M.K.; Thakur, S. Open Source Software Based Education and Training Framework for Software Engineering Education. *Solid State Technology*, 2020(63), pp. 9633–9645.

Sanchez, R.; Groc, M.; Vuillemin, R.; Pujo-Pay, M.; Raimbault, V. Development of a Frugal, In Situ Sensor Implementing a Ratiometric Method for Continuous Monitoring of Turbidity in Natural Waters. *Sensors*, 2023(23), pp. 1897.

Škraba, A., Stanovov, V., y Semenkin, E. (2020). Development of control systems kit for study of PID controller in the framework of cyber-physical systems. *In IOP Conference Series: Materials Science and Engineering*, 734, (1), p. 012105. IOP Publishing.

Sowinski, P.; Rachwał, K.; Danilenka, A.; Bogacka, K.; Kobus, M.; D ˛abrowska, A.; Paszkiewicz, A.; Bolanowski, M.; Ganzha, M.; Paprzycki, M. Frugal Heart Rate Correction Method for Scalable Health and Safety Monitoring in Construction Sites. *Sensors*, 2023(23), pp. 6464.

Tapaskar, R.; Revankar, P.; Gorwar, M.; Hosmath, R. Pedagogical Interventions through Software Tools in Postgraduate Engineering Programme. *Journal of Engineering Education Transformations*, 2018(31).

Troisi, O., Visvizi, A., y Grimaldi, M. (2023). Rethinking innovation through industry and society 5.0 paradigms: a multileveled approach for management and policy-making. *European Journal of Innovation Management*, 27(9), pp. 22-51.

Vaca, N.; Garcia-Loro, F.; Martin, S.; Rodriguez-Artacho, M. Raspberry Pi Applications in Electronics and Control Laboratories. *In Proceedings of the 2022 IEEE Global Engineering Education Conference (EDUCON).* IEEE, 2022, pp. 1709–1713.

Wajid, B.; Ekti, A.R.; AlShawaqfeh, M.K. Ecebuntu-an innovative and multi-purpose educational operating system for electrical and computer engineering undergraduate courses. *Electrica*, 2018(18), pp. 210–217.

Yılmazlar, E., Kuşçu, H., Erdemir, V., y Güllü, A. (2018). Design Of Stepper Motor Control Interface With Embedded Systems. *International Journal of Engineering Research and Development*.

http://avacient.chetumal.tecnm.mx/index.php/revista https://www.facebook.com/avacient
https://doi.org/10.69823/avacient.v4n2a3