UNIVERSIDAD AUTÓNOMA DEL ESTADO DE MÉXICO

CENTRO UNIVERSITARIO UAEM ATLACOMULCO

"Desarrollo de sistema para el manejo de 3 ejes implementado a una Máquina de Control Numérico"

T E S I S

Que para obtener el Título de:

Ingeniero en Computación

Presenta:

Axel Correa Espinoza

Director de Tesis:

Dr. Carlos Eduardo Torres Reyes

Agosto del 2017

RESUMEN

El presente documento presenta el desarrollo e implementación de un sistema para el manejo de los movimientos de tres ejes (X, Y, Z) de un sistema mecánico con un motor a pasos en cada eje, mediante dos interfaces gráficas desarrolladas en Python 3.4 y en Raspberry Pi, que permita al usuario manipular el movimiento de los ejes de manera manual mediante botones para permitir un desplazamiento en segmentos y dentro de esta misma el manejo de los ejes de manera libre mediante un dispositivo externo (Control USB), también se permita mover tres ejes automáticamente indicando la secuencia como la distancia y el tiempo para desplazar entre cada uno de ellos.

ABSTRACT

This paper presents the development and implementation of a system for the management of three-axis movements (X, Y, Z) of a mechanical system with a stepper motor on each axis, through two graphical interfaces developed in Python 3.4 and in Raspberry Pi, which allows the user to manipulate the movement of the axes manually by means of buttons to allow a movement in segments and within this same the management of the axes freely by means of an external device (USB Control), also allow to move Three axes automatically indicating the sequence as the distance and the time to move between each one of them.

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

1 INTRODUCCIÓN

1.1 Antecedentes

El desarrollo de sistemas de Control Numérico por Computadora es de gran importancia para la industria moderna, ya que permiten maquinar piezas de manera precisa. Se emplea para procesos de grabado en materiales como madera, metal, plástico y cuero. Estos sistemas de control numérico iniciaron después de la segunda guerra mundial, utilizando tarjetas perforadas con las coordenadas necesarias para la fabricación de partes de aviones. [1]

Actualmente, en la industria manufacturera el uso de Máquinas de Control Numérico y sistemas robóticos permiten mejorar la calidad de producción. La automatización de máquinas y procesos han permitido mejorar la productividad en las industrias, disminuye los costos de producción, genera productos de alta calidad de manera eficiente, precisa y de calidad. [2]

EL Control Numérico se encuentra adaptado para fresadoras y tornos como principal uso. La fresadora realiza mecanizados por arranque de viruta mediante el movimiento de una herramienta rotativa de varios filos de corte denomina fresa y puede operar en 3, 4, 5 y 6 grados de libertad. El torno es una herramienta que permite precisión en el mecanizado de las piezas, su trayectoria de la herramienta de torneado es controlada por computadora. [1]

Se empelan dos tipos de motores para el movimiento de ejes, los motores a pasos y los servomotores. Los servomotores se utilizan en sistemas de movimiento para CNC de tipo industrial, los cuales pueden proporcionar una resolución de hasta 25,000 pasos por vuelta y lograr velocidades de giro, de hasta 6,000 RPM, los motores a pasos dependen de un manejador electrónico (driver) que proporcione el número de pasos y la dirección de giro para lograr una revolución completa, proporcionando una resolución de 10,000 pasos por vuelta y una velocidad de giro de 700 RPM. [1]

La técnica de control **step/dir** es una técnica utilizada en el mundo aplicada a la CNC, **step** corresponde al bit de entrada e indica los pasos por medio de pulsos y **dir** indica la

dirección de giro en sentido horario o anti horario. El sistema central de control, se basa principalmente aplicado aun microcontrolador permitiendo generar una conexión entre la máquina CNC y la PC. [2] En la actualidad existen dispositivos como Raspberry Pi que permiten una serie de ventajas sobre un microcontrolador. La facilidad de conexión con otros periféricos y dispositivos, la velocidad de procesamiento y la implementación de lenguajes más poderosos que C. [3]

En el capítulo 2 se habla sobre el tema central del trabajo, que consiste en la propuesta de un sistema para el posicionamiento de tres ejes implementado a una Máquina de Control Numérico, en donde se permita desplazar una antorcha de plasma con una precisión de 8 milímetros y en un tiempo de 1s aproximadamente la revolución, de manera manual y automática. Por medio de Raspberry Pi y Python implementar interfaces gráficas que permitan al usuario interactuar con el sistema para el desplazamiento.

En el capítulo 3 se presenta la propuesta para la solución del problema, donde se propone el desarrollo del sistema mediante Raspberry Pi e implementando Python como lenguaje de programación para establecer un desplazamiento manual y automático de tres ejes. También se propone que el desplazamiento de los ejes pueda rotar a una velocidad aproximada de 1s por revolución o en 8 milímetros.

En el capítulo 4 presenta un análisis de las literaturas relacionadas con el tema a presentar, en donde se identifican los elementos importantes como: los componentes o dispositivos empleados, su funcionalidad y las técnicas empleadas para la elaboración del trabajo. Todo esto es presentado mediante una tabla de referencia donde se identifican las ventajas y desventajas que se consideran para la implementación de sistemas para Máquinas de Control Numérico.

En el capítulo 5 describe el procedimiento para implementar el sistema deseado, mediante diagramas UML de: caso de uso, clases, distribución, entre otros. Así como los requisitos y especificaciones técnicas del sistema, el proceso de diseño e implementación del sistema mediante diagramas de flujo y por último se describe el procedimiento para realizar las pruebas del funcionamiento del sistema.

2 PLANTEAMIENTO DEL PROBLEMA

El tratamiento de materiales por diversas técnicas entre ellas el plasma; permite mejorar sus propiedades, el inconveniente principal es llevar a la antorcha de plasma a la posición adecuada de manera precisa. El uso de Máquinas de Control Numérico (MCN) es una opción que resulta conveniente, debido a la precisión en el posicionamiento de los motores que utiliza (motores a pasos), aunado a esto, generalmente se tienen circuitos manejadores (Driver) comérciales que pueden proporcionar las señales eléctricas necesarias para el funcionamiento de los motores a pasos. Aunque están limitados en corriente y por consiguiente la potencia de consumo de los motores. Lo anterior afecta cuando estos motores requieren de un mayor torque para trasladar un peso significativo, aumentando la corriente de consumo (la potencia de consumo aumenta también).

2.1 Definición del problema

El Centro Universitario UAEM Atlacomulco cuenta con 4 Máquinas de Control Numérico, las cuales no cuentan con un sistema que se encargue de llevar una antorcha de plasma para el tratamiento de superficies en materiales, mediante un sistema de desplazamiento y posicionamiento dentro de 3 ejes “X”, “Y” y” Z”, por medio de una Máquina de Control Numérico.

2.2 Objetivos de investigación

Diseñar y construir un sistema electrónico y de software que permita el manejo de tres ejes de una máquina de control numérico para el posicionamiento y desplazamiento de una antorcha de plasma.

1.1.1 Objetivos específicos

- Analizar el funcionamiento de los motores a pasos y construir un driver para el funcionamiento del motor, que sea capaz de soportar corrientes de hasta 6 A, para el tratamiento de materiales mediante plasma, se requerirá de motores con un mayor consumo de corriente y el manejador electrónico deberá de soportar este consumo.
- Establecer la resolución de pasos en los motores para establecer un movimiento en milímetros.
- Diseñar una interfaz gráfica que permita manipular 3 ejes de manera manual.
- Diseñar una interfaz que permita manipular 3 ejes de manera automática.

- Integrar los dos modos de trabajo en una sola interfaz.
- Realizar pruebas para comprobar el funcionamiento correcto del sistema.

Para la realización de estos objetivos se requiere que, antes de empezar con el diseño de la interfaz, es necesario analizar el funcionamiento de los motores a pasos, es decir la cantidad de pasos necesarios para poder establecer un movimiento en milímetros e implementar el driver para el funcionamiento de los motores.

2.3 Preguntas de investigación

- ¿La programación Orientada a Objetos y TKinter permitirá desarrollar una interfaz gráfica para manipular 3 ejes?
- ¿Cuál será el tiempo de respuesta entre la acción del usuario y el movimiento del eje?
- ¿Qué cantidad de pasos del motor se requiere para establecer el movimiento en unidades milimétricas?

2.4 Justificación

Existe software comercial dedicado para sistemas CNC el cuál se requiere de la compra de una licencia, también se tienen versiones libres, pero no está hecho a la medida para cumplir con requerimientos específicos como manejo en modo manual y automático. En el caso del Centro Universitario UAEM Atlacomulco no cuenta con estos tipos de software, se cuenta con 4 Máquinas de Control Numérico y se considera importante desarrollar sistemas de software libre que puedan adaptarse a las necesidades dentro de la carrera Ingeniería en Computación, como el desplazamiento de una antorcha de plasma, maquinado de piezas, entre algunas aplicaciones que requieran un sistema de posicionamiento y desplazamiento en 3 ejes.

Para la implementación del sistema se propone el uso de sistemas embebidos como Raspberry Pi, ya que puede operar bajo diferentes sistemas operativos de la distribución Linux, permitiendo desarrollar programas como Python, C, C++, entre otros. Entre algunas de las características de Raspberry Pi en la versión 2 son que cuenta con un procesador de 900 MHZ de cuatro núcleos, 1 GB de RAM, 4 puertos USB, un puerto de red Ethernet, un puerto HDMI entre otras características encontradas. Además Raspberry Pi es un sistema Open Hardware permitiendo una accesibilidad en costo, rondando entre los $50 USD. [4]

Python es un lenguaje interpretado, es decir, no requiere de alguna compilación para ejecutar, como una manera fácil de programar ya que la sintaxis no es tan estricta y puede manejar datos y archivos como cualquier otro lenguaje de programación de alto nivel como Java y C++, haciendo de Python una buena opción de programación en la tarjeta Raspberry Pi ya es un lenguaje que no requiere de muchos recursos al ser interpretado para su ejecución. [5]

La importancia de este trabajo radica, en la implementación de un sistema para el manejo de 3 ejes en una Máquina de Control Numérico, por medio de una interfaz gráfica desarrollada en Python e implementando una programación Orientada a Objetos, se pretende lograr un sistema que permita manipular 3 ejes de manera manual y automática para situar una antorcha de plasma o algún dispositivo que requiera un posicionamiento milimétrico. Dicho sistema permitirá desplazar los ejes a una velocidad máxima de 0.6s y una mínima de 1s por revolución, teniendo como resolución de desplazamiento en una revolución 400 pasos o una revolución en 8mm aproximados. Además la interfaz permitirá establecer una secuencia de desplazamiento en los ejes "X","Y" "Z" en pasos, revolución o en milímetros. También permitir un desplazamiento con libertad de manejo por medio de un dispositivo externo y por medio de botones dentro de la interfaz. Además, se desarrollará un manejador para motores a pasos que permita ser adaptado para el consumo de una corriente máxima de 6A y mayor de 12V aproximadamente. Por otro lado, se proporcionará una herramienta que servirá para que en otros trabajos se aborde el tratamiento de materiales por plasma.

2.5 Impactos

- Tecnológico: Creación de interfaz gráfica por medio de Python y Tkinter para la manipulación de tres ejes "X","Y","Z" y el diseño de un manejador electrónico para motores a pasos.
- Educativo: Permitir a los alumnos de Ingería en Computación y posgrado una mayor accesibilidad y uso de las máquinas de control numérico con las que cuenta el Centro Universitario UAEM Atlacomulco.

3 HIPÓTESIS

Si por medio de Raspberry Pi se implementa un sistema que permita controlar 3 motores a pasos entonces, se podrá establecer un sistema que permita posicionar y desplazar una antorcha de plasma en una Máquina de Control Numérico, con una precisión de 8 milímetros por revolución y con un tiempo de desplazamiento de hasta 1s por revolución.

4 ESTADO DEL ARTE

Se presenta la tabla de referencias, en donde se analiza todo el estado del arte relacionado al tema obtenido durante la investigación, en donde se identifican posibles ventajas y desventajas para un análisis posterior y poder sustentar el trabajo realizado de investigación.

Tabla 4.1 Tabla de referencia correspondiente al estado de arte.

Referencia	Título	Síntesis	Ventajas/Desventajas
F. Acuña, D. Rivas y C. Navarrete, *IEEE Latin America Transactions,* vol. 13, nº 6, pp. 1893-1898, 06 06 2015.	"Design and Construction of a 3D Printer Auto Controller Wirelessly Trough of Free Software"	El diseño de un sistema CNC enfocado a la impresión 3D requiere de un sistema electromecánico, de control y un sistema electrónico. • Sistema electromecánico: Se requiere estudiar los parámetros para el diseño como el accionamiento eléctrico, estructura mecánica que sea rígida y liviana de aluminio y material de extrusión. • Sistema de control: Sistema informático que atreves de un software transforma un objeto en formato STL (Stereolitografia) a código G, y el software envía la información al sistema electrónico por medio de software Cura y Pronterface, los cuales son de software libre, multiplataforma y desarrollados en Python. • Sistema electrónico: Se encuentra establecido por una tarjeta Arduino, manejador A4889, el cuál maneja los motores a pasos.	**Ventajas:** • El uso de software libre permite reducir los costos para el desarrollo de un sistema CNC como "Software Cura" y "Pronterface" los cuales permiten diseñar y controlar impresoras 3D y se encuentran desarrolladas en Python. • Implementación de una interface de comunicación inalámbrica por medio de las tarjetas XBee, permiten manejar el sistema CNC a distancia. **Desventajas:** • El uso de la tarjeta Arduino se encuentra limitado en su programación estructural requiere de una interface de comunicación hacia la PC a diferencia de otros sistemas como las Raspberry Pi. • EL uso del manejador para motor a pasos A4988 limita la corriente de consumo de los motores a pasos a 2ª.
A. Sasongko, A. Tulus Purnomo y	**"**Development of Interface	El diseño de un Sistema CNC para el grabado de PCB requiere de una	**Ventajas:**

F. Ishad Hariadi, «Ieee Xplorer,» *IEEE*, p. 6, 9 12 2015.	and Coordination for Control Module CNC PCB Milling Machine**”**	herramienta de corte, un sistema electrónico y un sistema de software. El sistema electrónico está conformado principalmente por la tarjeta FPGA, la cual controla la posición y desplazamiento de dos ejes. La comunicación entre la tarjeta FPGA y la PC es asíncrona por medio serial. Se requiere de un software que permita leer Gerber-code y transformar a G-code, la información se transfiere a la tarjeta FPGA para la operación de grabado.	• La tarjeta FPGA a diferencia de microcontroladores es que, permite ejecutar múltiples tareas. • La implementación de dos módulos de control para la posición y el desplazamiento, permite establecer una ejecución paralela y un monitoreo en tiempo real de la operación de grabado.
S. Pandian y S. R. Pandian, «http://www.irdindia.in/journal_ijmer/pdf/vol2_iss1/2.pdf,» *ISSN*, vol. 2, nº 1, pp. 6-11, 2014.	**“**A low cost build your own three axis CNC mill prototype**”**	En la industria, la implementación de sistemas CNC es de alto costo. La búsqueda de nuevas alternativas para la creación de sistemas CNC representa la oportunidad de que sea más accesible la adquisición de estos sistemas. Algunas de las alternativas para reducir los costos se encuentran en el software y en el hardware. Respecto al software encontramos sistemas Linux los cuales son Software Libre, como RTLInux que es un software orientado a CNC y presenta un prototipo de controlador para motor a pasos. En cuanto al Hardware se	**Ventajas:** • Implementar una herramienta de software libre como Universal-G-Code-Sender es una alternativa para poder automatizar el movimiento de tres ejes en una CNC, de acuerdo al código G que interpreta y envía al Arduino. **Desventajas:** • Una de las desventajas para implementar Universal-Code-Sender es que, requiere un archivo con el código G para poder mover tres ejes en un CNC, si se desea implementar en alguna aplicación donde no se utilicen códigos G, el sistema es inservible para otras aplicaciones.

		encuentra la tarjeta Arduino y Raspberry Pi los cuales son Hardware Libre y a costos accesibles. Para el prototipo que cuenta con 3 sistemas, el sistema mecánico, sistema electrónico de control y el sistema de control por software. • Para el sistema mecánico es implementado un CNC ya ensamblado de ZenToolsWorks, el cual ya cuenta con accesorios como ejes, tornillos de avance, barras guía etc. Cuenta con 3 motores a pasos para tres ejes. • Para el sistema electrónico implementan la tarjeta Arduino para el control del manejador electrónico para motores a pasos, la cual recibe las instrucciones por medio de un intérprete desde la PC. • En el sistema de software implementan el software Universal-G-Code-Sender, el cual es basado en Java y es multiplataforma. Este software es una interfaz para poder comunicar el Arduino con la PC, permite enviar el código G adquirido de una imagen o diseño en CAD por medio serial.	
O. J. Leslie, E. Ú. Sequeira y Yamil, «Máquina de Control Numérico Computarizado	**“**Máquina de control numérico computarizado (CNC) de cuatro ejes, con comunicación USB para la automatización de los procesos	EL desarrollo de un prototipo de CNC tipo Fresadora para el grabado en materiales requiere de establecer un sistema electrónico para el movimiento de los motores, un sistema de software y una herramienta	**Ventajas:** • Implementar un microcontrolador puede ser una ventaja ya que son dispositivos de bajo costo y de fácil adquisición. **Desventajas:** Se implementa un sistema de comunicación por medio de un microcontrolador.

(CNC) de cuatro ejes, con comunicación USB, para la automatización de los procesos de grabado en madera, plástico y materiales no convencionlaes, » de *Tesis de Licenciatura*, Managua, Universidad Nacional de Ingeneria, Facultad de Electrónica y Computación, 2011, p. 71.	de grabado en madera, plástico y materiales no convencionales ”	de corte para el grabado de los materiales. • En el sistema electrónico emplearon dos microcontroladores, un microcontrolador permite ser una interfaz de comunicación entre la PC y un segundo microcontrolador. El segundo microcontrolador permite controlar el manejador LMD18245T y este controla tres motores a pasos. • En el sistema de software emplearon CNC USB, ya que existe software como Mach3, KCAM, EMC, entre otros, que utilizan una comunicación por el puerto paralelo y este puerto ya no es muy común en los equipos de cómputo. • Las herramientas para el corte usadas son “Spindle, Aspiradora (Flood) y Lámpara (MIST). Estas herramientas pueden ser controladas del software CNC USB y se pueden configurar independientemente.	• A diferencias de otros sistemas como Raspberry Pi, el cuál no requiere de un sistema externo para la comunicación, ya que este sistema embebido que cuenta con su propio sistema operativo, y pines externos para la comunicación, activación de dispositivos.
N. Londoño Ospina, P. León Simanca, J. Álvarez Díaz y E. Marín Zapata, «Descripción del diseño y construcción de un torno de control	“Descripción del diseño y construcción de un torno de Control Numérico”	El diseño de un torno CNC requiere de un software de alto nivel, sistema de control-bajo nivel (Microcontrolador), motores y sistema mecánico. • El sistema mecánico consta de un carro transversal, un carro longitudinal, Torreta porta-herramientas y motores a pasos. • El sistema electrónico y también eléctrico consta de una fuente de alimentación, la cual debe de ser capaz para alimentar los motores y	**Ventajas:** • El diseño de un manejador para motores a pasos, permite adaptar un sistema electrónico con mejores características de acuerdo al motor de pasos implementado, también permite establecer un mayor rango de trabajo en cuanto al consumo de corriente de los motores, permitiendo implementar motores de mayor tamaño y mayor consumo.

numérico,» *Ingenier´ıa y Ciencia, ISSN*, vol. 1, nº 2, pp. 41-51, 21 06 2005.		todo el sistema electrónico. Se requiere de una interfaz entre un microcontrolador y el PC, establecen el integrado RS-232 para comunicar con el puerto serial. Un microcontrolador PIC-18F442 que permita ejecutar las instrucciones que llegan desde la PC y controle los motores a pasos. El manejador para los motores a pasos fue elaborado mediante optocopladores y Mosfet, el cuál recibe una secuencia de señales digitales estableciendo la rutina de trabajo para el motor a pasos.	**Desventajas:** • La programación en un microcontrolador es limitada, ya que emplean lenguajes como C, el cual es un lenguaje estructural y no es orientado a objetos como la implementación de multiprocesos.
M. Eric, M. Cesar y V. Carlos, «Google Scholar,» *EAFIT INGENERÍA Y CIENCIA*, vol. 1, nº 1, pp. 01-09, 26 05 2014.	**"**Diseño de controlador empleando sistema Arduino para posicionamiento de cabezal de impresión de tres dimensiones, ejemplo de aplicación utilizando prototipo de impresora 3D**"**	Actualmente la creación de modelos físicos a partir de datos digitales es posible por medio de software CAD. Esto permite diseñar la pieza por computadora y extraer los datos del diseño y convertirlos a otros como Códigos G y M. La implementación de un sistema fue implementada en un CNC de tres ejes, implementaron un sistema entre un Arduino Mega. Implementaron el software Pronterface como el sistema que manipula tres ejes por medio de la adquisición del Código G. El código G es enviado desde Pronterface y es adquirido por la tarjeta Arduino, la cual envía por medio de pulsos a la tarjeta RepRap Ramps. RepRap	**Ventajas:** • Como ya fue mencionado anterior mente en el artículo [6], la implementación de Pronterface presenta una gran ventaja para la construcción de prototipos CNC, ya que es de software y por lo tanto es gratis. **Desventajas:** • La implementación de una tarjeta como RempRamps que cuenta con el manejador A4988 ya mencionado en la presenta tabla del artículo [6], presenta una limitación para el consumo que presentan los motores a pasos y es de 2A como máximo.

		Ramps es una tarjeta desarrollada para funcionar con Arduino Mega, cuenta con el manejador A4988 para motores a pasos, permitiendo interactuar desde Pronterface, Arduino y RepRamps a los motores a pasos para su desplazamiento.	
G. Mondragón García , «Sistema de posicionamiento y barrido asistido por computadora para la aplicación de descargas superficiales,» Atlacomulco, 2013.	**“**Sistema de posicionamiento y barrido asistido por computadora para la aplicación de descargas en superficies**”**	La literatura contiene la implementación de un sistema mediante LabView y Arduino para poder mover tres ejes de una Máquina de Control Numérico. El sistema se encuentra dividido se encuentra en 2 módulos, el sistema de software y electrónico. • Sistema de software: El sistema de software fue empleado mediante LabView, en el cuál se desarrolla la interfaz gráfica para el desplazamiento de dos ejes de forma manual y automática. • El sistema electrónico: Se encuentra conformado por la tarjeta Arduino Uno y el manejador L297 y L298. La interfaz gráfica presentada la cual fue desarrollada en LabView, permite interactuar de dos maneras al usuario, manual y automático. Las instrucciones son enviadas a un Arduino por medio serial. El Arduino procesa la información otorgada de LabView y envía una señal PWM al manejador L297, este	**Ventajas:** • LabView es un software flexible para crear interfaces gráficas y además al ser un software que emplea lenguaje gráfico. **Desventaja:** • LabView es un software de paga y presenta una desventaja para la implementación de sistemas de menor costo y fácil acceso a ellos.

		genera una secuencia de pasos y esta secuencia es enviada al manejador L298 y permite el movimiento de los motores a pasos.	

En la Tabla 1 se muestran algunas de las literaturas relacionadas al tema de investigación. Se presentan las implementaciones para sistemas de Máquinas de Control Numérico, como también, la selección de controladores para motores a pasos y el software empleado para la implementación del sistema.

En las literaturas ya mencionadas se desglosan en tres secciones importantes:

- Sistema mecánico: Lo compone la estructura de la Máquina de Control numérico, como motores, rieles, carro transversal, carro longitudinal, entre otras piezas.
- Sistema electrónico: Es compuesto por controladores para motores a pasos, fuentes de alimentación, interfaces de comunicación. En las literaturas encontramos controladores para motor a pasos como el A4988, LMD18245T, L297, L298 y el desarrollo de un Manejador para motor a pasos por medio de MOSFET.
- Sistema de software: Se encuentran software como LabView, PronterFace, Universal-G-Code-Sender, Cura 3D.

Analizando las literaturas, se puede destacar que los programas de software libre presentan una ventaja para implementar sistemas CNC, ya que son gratis y permite reducir el costo de implementación de un sistema, también el desarrollo de nuevos programas basados en lenguajes de software libre para establecer nuevos sistemas CNC y se adapten a las necesidades requeridas. También se encuentran controladores para motores a pasos, los cuales presentan una desventaja, ya que, si se desea implementar motores de un consumo mayor a 2A (corriente eléctrica que puede suministrar un manejador comercial), el manejador se puede dañar. Una ventaja ante lo anterior, es el diseño de un propio controlador para motores a pasos que pueda tener un amplio rango de trabajo, en caso de que se considere emplear motores de consumo mayor a 24W.

El sistema permitirá desplazar una antorcha de plasma frío para el tratamiento de materiales y la deposición de estructuras como nano estructuras de carbono. También permitirá desplazar componentes o dispositivos para diversas aplicaciones de mayor peso,

los cuales no requieran un consumo del motor mayor a 6A y voltaje mayor a 12V. Por otra parte, el sistema permitirá una velocidad de desplazamiento en los ejes a una velocidad aproximadamente de 1 segundo/revolución a 2 segundos/revolución, siendo la resolución de una revolución igual a 400 pasos y un desplazamiento por revolución de 8 milímetros. Además, el sistema permitirá una manipulación de los tres ejes a libertad por medio de dispositivo externo (Control JoyStick USB) y una manipulación automática de los 3 ejes, en donde el usuario establecerá las coordenadas en milímetros y el sistema ejecutará la secuencia.

5 METODOLOGÍA

A continuación, se describe la metodología empleada para la elaboración del presente trabajo. Se describen los elementos necesarios para la realización del sistema y las pruebas necesarias para comprobar el funcionamiento del mismo.

5.1 Requerimientos o especificaciones

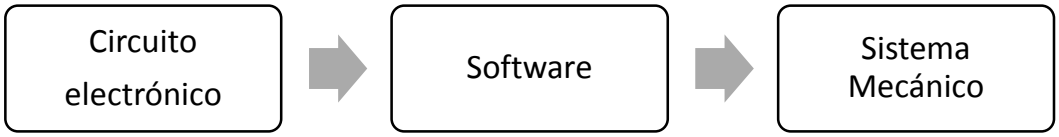


Figura 5-1 Elementos que componen el sistema total para el manejo de tres ejes.

En la figura 5.1 se presentan los 3 elementos que en conjunto permiten mover 3 ejes. Se cuenta con el sistema mecánico (Máquina de Control Numérico), que contiene: 3 motores a pasos (3 ejes), barras guía de acero, rodamientos lineales de precisión, tornillos M8x1.25 y construida con tableros de alta densidad de PVC [7]. Para mover los ejes del sistema mecánico se requiere de "drivers" que son circuitos electrónicos que permiten acondicionar señales cuadradas de cierta una duración determinada a señales eléctricas necesarias para mover los motores a pasos, y finalmente el software donde se desarrollará la interfaz gráfica para emitir las señales adecuadas para efectuar los movimientos deseados mediante Python. Las condiciones eléctricas necesarias para el sistema son:

- Compatible para motores a pasos (Unipolar y Bipolar).
- Tolerancia de Voltaje no mayor a 24V.
- Tolerancia al consumo de corriente no mayor a 6A.
- Pueda ser operado por algún microcontrolador o sistema embebido.

Debido a que comercialmente los drivers para motores a pasos se encuentran limitados el consumo de corriente y voltaje (potencia aproximada de 24W de consumo). Por lo tanto, se considera importante construir un driver que pueda soportar mayores corrientes y voltajes que los comerciales, debido a que si se requiere implementar en el sistema algún motor que consuma una mayor potencia no superior a 76W (de acuerdo al peso que puede tener la antorcha de plasma de acuerdo a su configuración).

Es necesario empelar algún dispositivo en el cual se pueda desarrollar interfaces gráficas y pueda comunicarse con el driver para el motor a pasos. En la Tabla 5.1 se presentan algunas tarjetas que cumplen con las características necesarias para el proyecto. Se considera la tarjeta Raspberry Pi 2 que es, una microcomputadora que opera bajo un sistema operativo (Linux, Windows, entre otros). Contiene: un microprocesador de dos núcleos, 40 pines que pueden ser programados para leer señales digitales (pulsos) o establecer salidas digitales, que permiten interactuar con componentes electrónicos, mediante: Python, C, C++, Java, entre otros.

Tabla 5.1 Tarjetas embebidas

Tipo	Raspberry Pi	Galileo	Udoo Neo Basic
Costo	$45.57 USD	$59.99 USD	$49.90 USD
Procesador	Quad Core @900Mhz	QuartX1000 @400Mhz	ARM Cortex-A9 core
Ram	1GB SDRAM @ 400Mhz	256MB	512MB
Gpio	40 Pin	12 Pin	22 Pin
Puerto Ethernet	Si	Si	Si
Sistema operativo	Linux, Windows	-	Linux

En el mercado existen microcomputadoras como Raspberry Pi, Udoo Neo y Galileo de Intel por mencionar algunas de ellas. De acuerdo a la tabla anterior Raspberry Pi es la tarjeta con mayor capacidad de memoria RAM, tiene un procesador de cuatro núcleos, es económica a comparación de las otras dos presentadas, y se puede considerar la tarjeta con mejores características de acuerdo a su precio.

El motor a pasos empleado es el SST59D3201, un motor unipolar de 6 bobinas o devanados dentro del motor. El ángulo de rotación por cada paso completo del motor es de 1.8°, para poder generar una vuelta completa o una rotación de 360° en una configuración de paso completo se requieren 200 pasos. Como se presenta en la Tabla 5.3 la configuración para una secuencia de paso normal.

Tabla 5.2 Secuencia de pasos tipo normal

Paso	Bobina 1	Bobina 2	Bobina 3	Bobina 4
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1

5.2 Diseño e implementación

El sistema se encuentra conformado por dos partes esenciales, la parte de software y hardware y por medio de diagramas UML (Lenguaje Unificado Modelado) y de flujo se presenta el análisis previo y el diseño de cada parte del sistema.

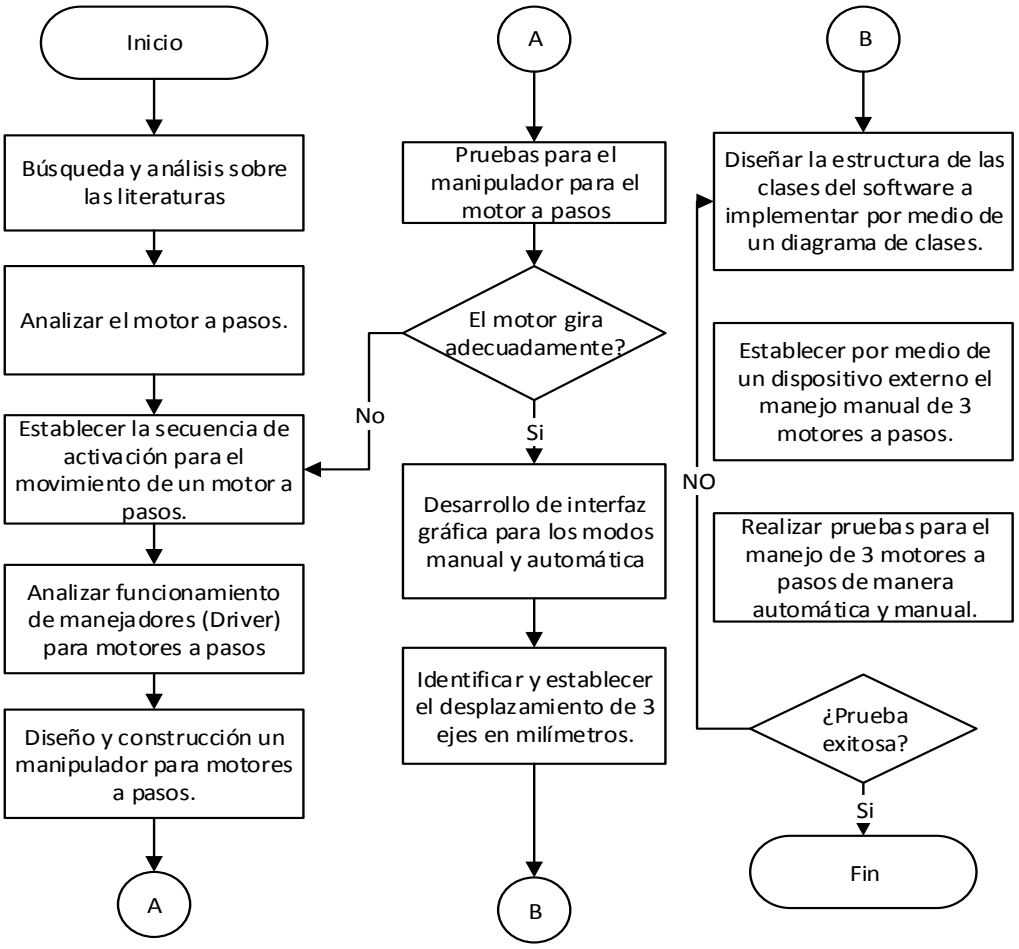


Figura 5.2 Procesos para la elaboración del sistema

El diagrama anterior muestra los procesos que se requieren realizar para la implementación del sistema. Se requiere analizar el funcionamiento de los motores a pasos, para poder establecer una secuencia que permita controlar la rotación y dirección del motor. De acuerdo al análisis anterior, se identifican las características o las variables que permiten rotar al motor como: corriente, voltaje, tiempo, resistencia, entre otras). Se debe realizar una prueba para determinar si el motor gira en la dirección establecida y rota la cantidad de pasos establecidos. Una vez que se tiene al sistema electrónico funcionando correctamente, se puede implementar el sistema de software por medio de interfaces gráficas.

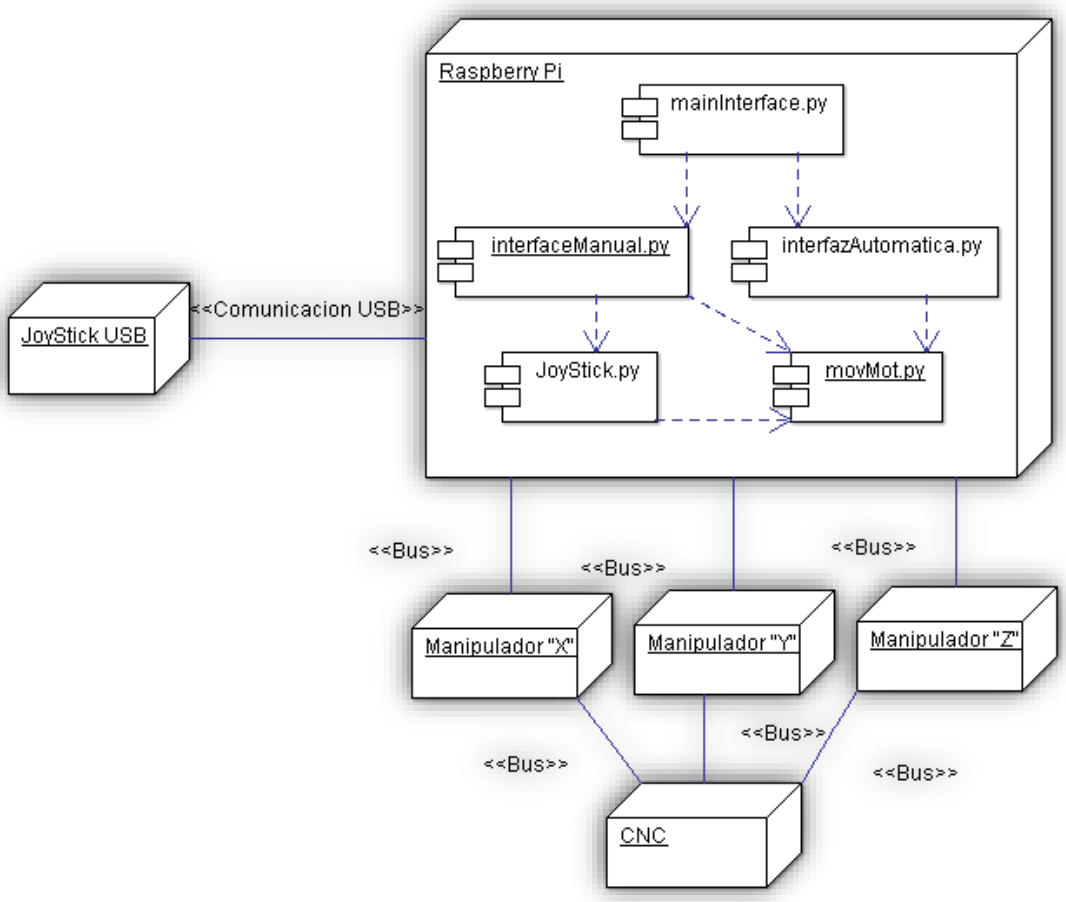


Figura 5.3 Distribución física de los componentes empleados.

La Figura 5.3, se muestra el diseño de la distribución de los elementos físicos empleados (nodos) que componen al sistema, así como la conexión que existe entre los nodos. La tarjeta Raspberry Pi se encuentra establecida como el nodo principal del sistema, dentro

de este se encuentra las clases desarrolladas para el funcionamiento de dicho sistema. Al nodo principal se encuentran conectados diversos nodos, uno de ellos es un control USB JoyStick y los drivers para motores a pasos por medio de buses de datos, y estos últimos nodos tienen como conexión común al sistema mecánico (Máquina de movimiento de tres ejes).

5.2.1 Sistema electrónico

Los motores a pasos son convertidores electromagnéticos que transforman una serie de impulsos eléctricos en rotaciones angulares del eje de salida. El sentido, la velocidad y el desplazamiento angular en la rotación, se encuentran determinados por la secuencia de pulsos recibidos al motor. El paso angular del motor a pasos se encuentra comúnmente determinado en 1.8° (0.9° y 90° menos frecuente). Los motores a pasos se han empelado actualmente en impresoras, scanners, fotocopiadoras, plotters, entre otros componentes. [8]

El sistema electrónico está compuesto principalmente por la tarjeta Raspberry Pi y el manejador para motores a pasos, Raspberry Pi envía la secuencia de activación mediante pulsos eléctricos de 3.3V al manipulador electrónico y este amplifica el pulso recibido a 12V, que este a su vez, alimenta las bobinas del motor a pasos y de acuerdo a la secuencia enviada desde la Raspberry Pi, el motor rotara a un sentido y a una velocidad.

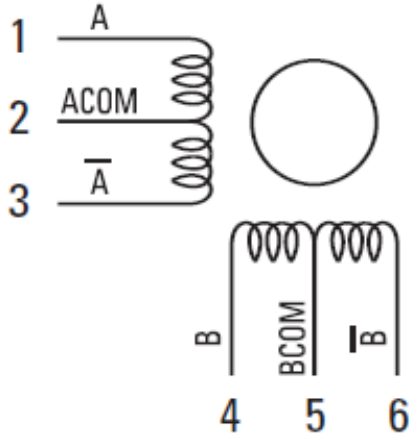


Figura 5.4 Cableado de las bobinas del motor a pasos empleado [9]

En la Figura 5.4 se muestran las terminales del cableado de las bobinas del motor a pasos empleado, el motor cuenta con 2 bobinas divididas en 4 devanados o fases, identificadas como A, $\bar{A}$, B y $\bar{B}$, y se encuentra la conexión común entre las bobinas identificadas como ACOM correspondiente a la bobina A y BCOM que corresponde a la bobina B. De acuerdo a lo anterior, se puede establecer la secuencia para la dirección de la rotación del motor alimentando o activando las fases de las bobinas del motor.

Se requiere de una secuencia de activación para determinar la dirección de giro del motor a pasos, la velocidad y la resolución de acuerdo a los pasos. En la Tabla 5.3 se presenta dicha secuencia diseñada para la rotación del motor en sentido horario a medio paso.

Tabla 5.3 Secuencia de activación para la rotación del motor con sentido horario.

Paso	Fase 1	Fase 2	Fase 3	Fase 4
1	1	0	0	0
2	1	1	0	0
3	0	1	0	0
4	0	1	1	0
5	0	0	1	0
6	0	0	1	1
7	0	0	0	1
8	1	0	0	1

En la Tabla 5.3 Secuencia de activación para la rotación del motor con sentido horario. Se presenta la secuencia de activación de las fases de las bobinas para la rotación del motor en sentido anti-horario con una configuración de medio paso.

Tabla 5.4 Secuencia de activación para la rotación del motor con sentido anti-horario.

Paso	Fase 1	Fase 2	Fase 3	Fase 4
1	1	0	0	1
2	0	0	0	1
3	0	0	1	1
4	0	0	1	0
5	0	1	1	0
6	0	1	0	0
7	1	1	0	1
8	1	0	0	0

Al establecer un avance de medio paso permite tener una resolución mayor, ya que, si se emplea por medio paso, el motor rota 0.9° y requiere de 400 pasos para una revolución. En el avance de paso completo (4 pasos) el motor rota 1.8° por paso y requiere de 200 pasos para completar una revolución, como se expresa en las ecuaciones de la Tabla 5.5. Es conveniente trabajar con medio paso ya que presenta el doble de pasos que en paso completo y permite tener mayor precisión por una diferencia de 0.02 milímetros.

Tabla 5.5 Comparación entre paso normal y paso medio

Paso Normal	Medio Paso
1 paso = 1.8° o 0.04 milímetros	1 paso = 0.9° o 0.02milímetros
1 revolución (200 pasos) = 8milímetros	1 revolución(400 pasos) = 8 milímetros
1.8° * 200 pasos = 360°	0.9° *400 = 360°

De acuerdo a la configuración ya establecida para el movimiento de los motores a pasos, se requiere diseñar un circuito electrónico que permita activar o alimentar las bobinas dentro del motor.

La secuencia para rotar el motor a pasos es programada en Raspberry Pi, que envía la secuencia por medio de pulsos eléctricos de 3.3V a través del puerto GPIO. Estas señales son recibidas por el driver y este amplifica hasta 12V CD los pulsos recibidos para

alimentar las fases del motor a pasos. Para proteger las etapas (señales de control de Raspberry Pi y señales de potencia hacia los motores), se implementa un circuito optocoplador. El optocoplador tiene como fin amplificar el voltaje para activar un MOSFET y este al ser activado por el optocoplador de acuerdo a la señal enviada de la Raspberry Pi, alimenta las fases de las bobinas dentro del motor. Este es el principio que se requiere para permitir la activación de las fases del motor.

Se requiere diseñar un circuito que permita conmutar señales de baja potencia a señales de mayor potencia para alimentar los devanados del motor, teniendo en cuenta que se desea establecer una corriente máxima de 6A, un voltaje mayor a 12V. Se establece el valor de 12V como el voltaje definido para aplicar al circuito. De acuerdo al voltaje ya establecido de 12V se obtiene a partir de la Figura 5.5, en donde el MOSFET (Q1) presenta una resistencia de 0.0078 Ω, la fase del motor presenta una resistencia de 2 Ω, la conexión común de la bobina del motor se conecta a una resistencia de 10 Ω y está a tierra para la protección del circuito.

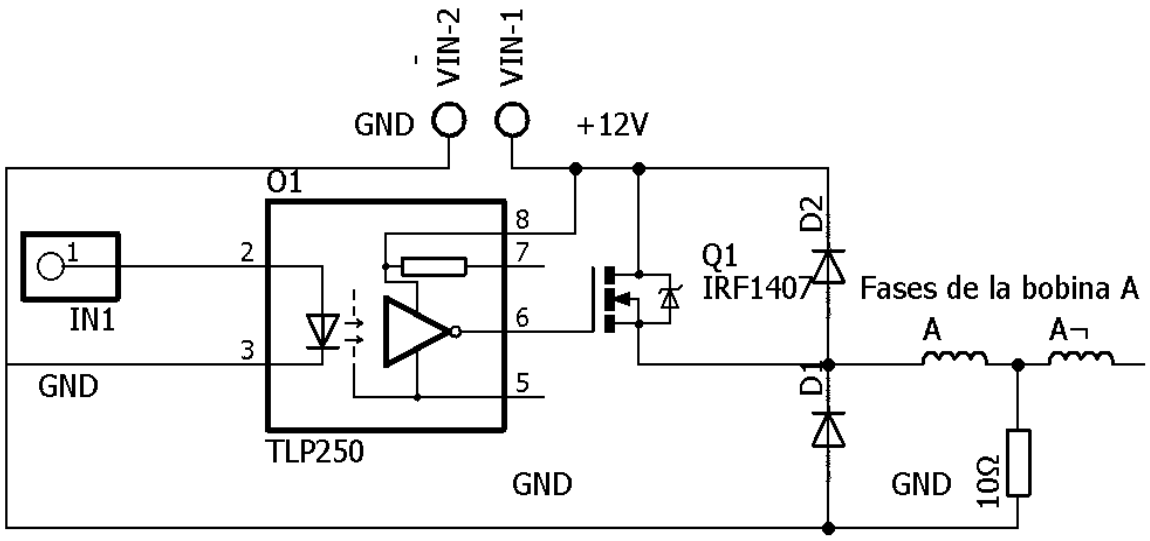


Figura 5.5 Configuración para activar una fase del motor

El driver para motores a pasos, permite enviar pulsos eléctricos provenientes de la tarjeta Raspberry Pi y permite rotar el motor en un sentido y dirección. El cuál se encuentra diseñado por optocopladores y MOSFET, el optocoplador con matrícula TLP250 y el MOFET IRF1407 (véase en Anexo A y Anexo B)

De acuerdo a los datos presentados de los componentes mencionados del MOSFET y el optocoplador, es posible realizar un diseño para la construcción de un driver para motores a pasos, ya que se conoce las características como voltaje, corriente, resistencia y frecuencia que manejan los dispositivos.

Se emplea la siguiente ecuación de acuerdo a la ley de Ohm para determinar el consumo de corriente que tendrá el circuito cuando una fase se encuentre alimentada:

$$Rmosfet + Rfase + Rcomun = Rtotal$$

La suma del total de la resistividad en el circuito es dada por la suma de la resistencia del MOSFET, fase del motor y de la resistencia conectada en el común del motor como se muestra en la Figura 5-9

$$0.0078\Omega + 2.0\Omega + 13\Omega = 15.0078\ \Omega$$

Se aplica ley de Ohm teniendo en cuenta que el voltaje aplicado en el circuito será de 12V y teniendo una resistencia total de 15.0078 Ω, obtenemos la corriente a consumir en el circuito dada por:

$$A = \frac{V}{Rtotal} = \frac{12V}{15.0078} = 0.7995A$$

Donde:

- *A*= corriente medida en Amperios
- *V*=voltaje medido en Volts
- *Rtotal*=Resistencia medida en Ohms.

La corriente que requerirá el circuito para alimentar una fase del motor será de 0.799 Amperios, mientras el voltaje aplicado sea 12V. La potencia que consume el manipulador electrónico es dada por:

$$P=A*V$$

Donde es el valor de la potencia medida en Watts (W), remplazando los valores de corriente y voltaje ya obtenidos, sustituimos la ecuación para obtener la potencia

$$P=0.799A*12V$$

P=9.58Wl la potencia otorgada en el circuito será de 9.58W.

Se requiere 4 activadores para las cuatro fases del motor a pasos, en la siguiente figura se muestra el diagrama completo del manipulador electrónico para el motor a pasos.

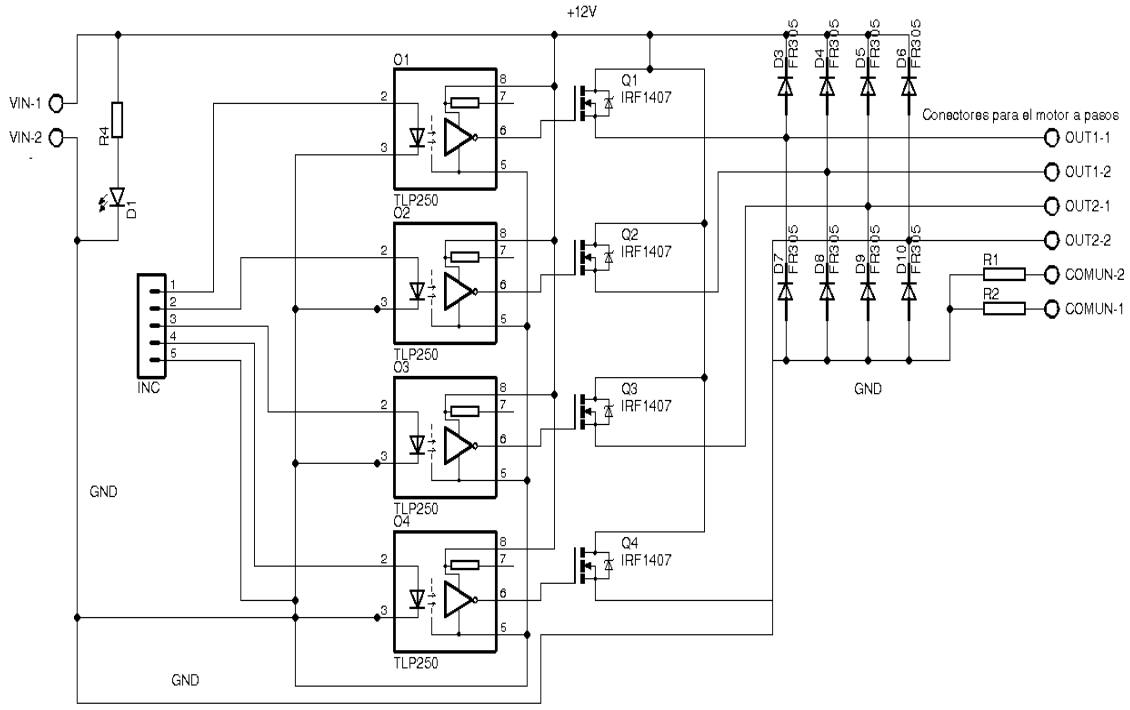


Figura 5.6 Diagrama electrónico del manipulador para motores a pasos

En la Figura 5.6 se muestra el diagrama electrónico del driver para motores a pasos diseñado, ya que el motor a pasos empleado contiene 4 fases, se requiere una conexión de entrada de pulsos en 4 pines, por consiguiente, se requiere 4 optocopladores y 4 MOSFET para activar las 4 fases. El optocoplador empleado es el TLP250 y el MOSFET es el IRF1407. En la Tabla 5.6 se presentan los valores de operación del driver.

Tabla 5.6 Parámetros de trabajo del manipulador electrónico para motores a pasos diseñado

Símbolo	Parámetro	Valor	Unidad
VIN	Entrada de la fuente de alimentación	12 a 24	V
OUT1-1 – OUT2-2	Pulsos de salida al motor	12 a 24	V
COMUN1-2	Común del motor	GND	V
INC	Entrada de datos	3.3 a 5	V

5.2.2 Sistema de Software

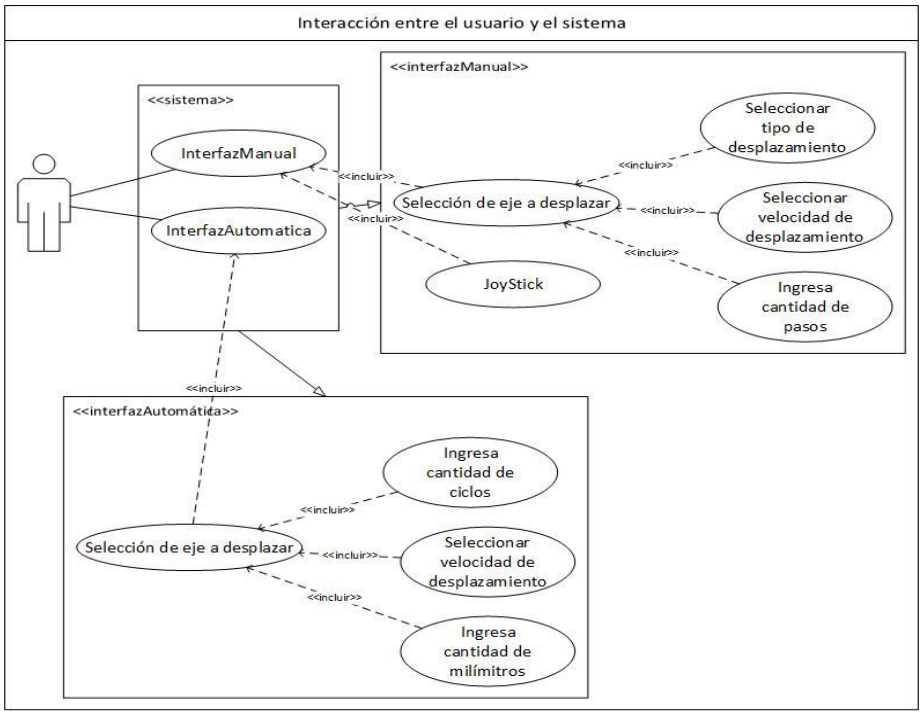


Figura 5.7 Interacción entre el usuario y el sistema gráfico

En el diagrama de caso de uso presentado en la Figura 5.7, se muestra la interacción que tiene el usuario con el sistema. Existe un nodo principal denominado "<<sistema>>", dos denominados "<<interfazManual>>" e "<<interfazAutomática>>". El sistema principal presenta dos opciones en la cual el usuario tendrá que seleccionar, si la selección es el caso de uso "interfazManual", se desplegara el nodo "<<interfazManual>>" el cuál requerirá que el usuario seleccione las opciones de, cantidad de pasos, velocidad de desplazamiento y tipo de desplazamiento; También el usuario dentro de este mismo nodo podrá seleccionar el caso "JoyStick" el cual permite la interacción mediante un control JoyStick USB. De lo contrario si el usuario selecciona el caso de uso "interfazAutomática", entonces se desplegará el nodo "<<interfazAutomática>>", el cual requerirá que el usuario seleccione las opciones de velocidad e ingrese la cantidad de milímetros y ciclos.

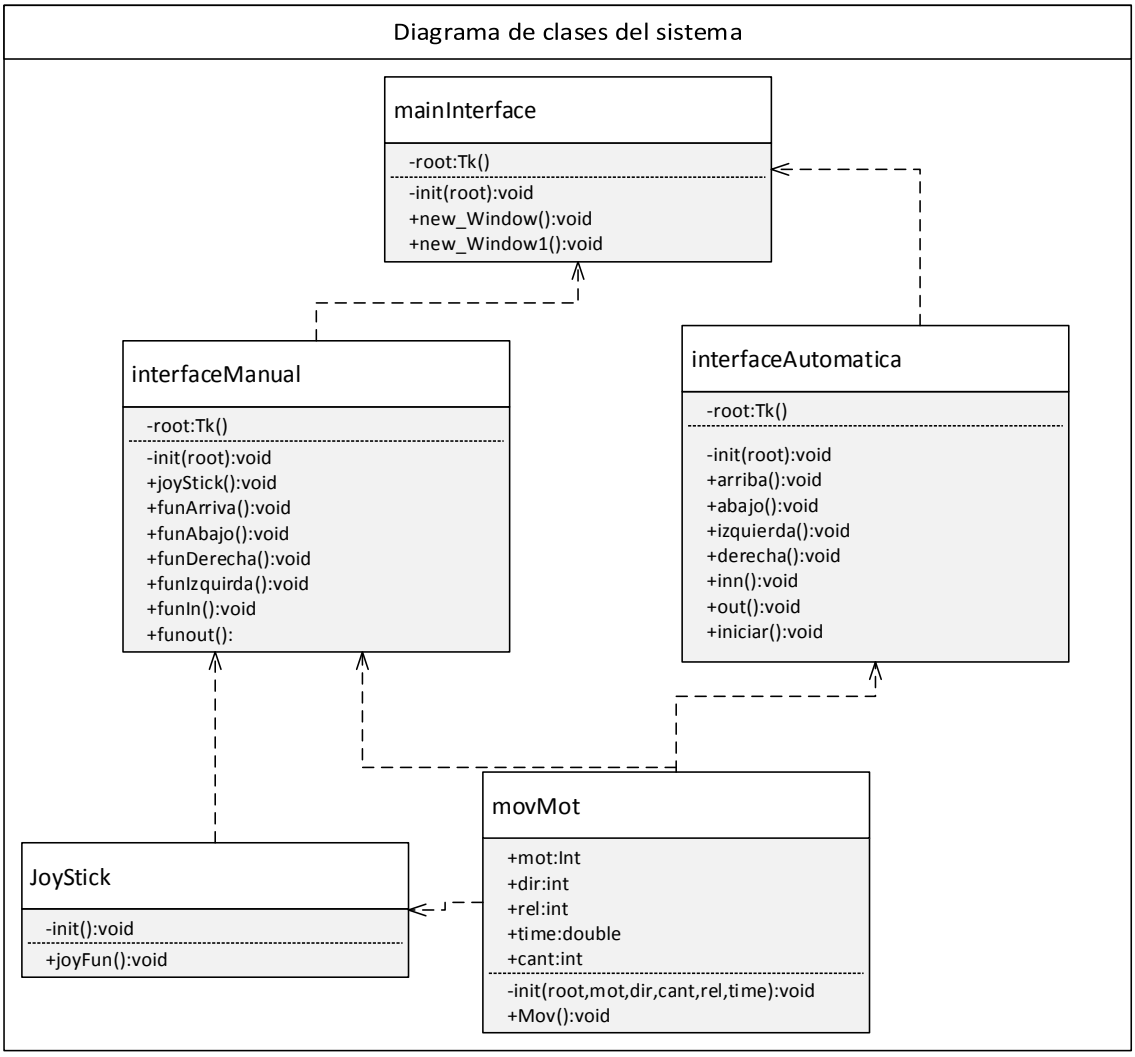


Figura 5.8 Diagrama de Clases

En la Figura 5.8, muestra las clases del sistema y la dependencia que existe entre cada una de ellas. La clase principal llamada “mainInterface” contiene los métodos e instrucciones para mostrar la interfaz principal y permite mandar llamar otra clase de acuerdo a la selección del usuario que visualiza otra interfaz (Manual o Automático). El sistema

Para poder manejar los ejes, se requiere del software, que permita al usuario decidir qué tipo de manejo emplear (Manual o Automático), la velocidad de desplazamiento y la unidad de desplazamiento (milímetros, revolución y pasos) y todo por medio de interfaces gráficas. Se implementa Tkinter en Python para la creación de interfaces gráficas, ya que Tkinter es una biblioteca de módulos para la creación de entornos gráficos como (Ventanas, botones, cajas de texto, entre otras más). Antes de la creación de las interfaces

se requiere desarrollar el programa que permita comunicar la Raspberry Pi con el manejador de los motores a pasos. Como se expresa en la figura 5-4 (Diagrama de clase) se emplea una clase llamada movMot.py.

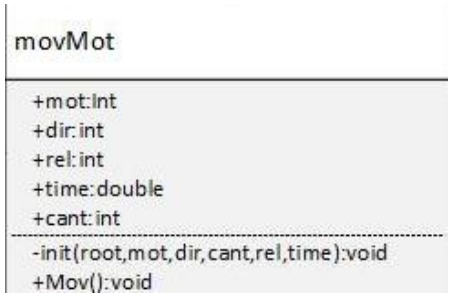


Figura 5.9 Clase movMot.py

En la Figura 5.9 se muestra la estructura y composición de la clase movMot.py empleada para a comunicación entre la Raspberry Pi y el driver electrónico para los motores a pasos, en donde se muestra las variables empleadas, funciones y métodos empleados, la clase requiere de 5 variables que indican cuál de los tres ejes (motores a pasos) se emplea para ejecutar esa secuencia, la dirección de giro, el relevador empleado para activar la alimentación del manipulador, el tiempo de retardo que tendrá entre cada paso y la cantidad de paso para rotar o desplazar y todas estas variables son asignadas por el método constructor identificado como “init”. Para determinar el funcionamiento de acuerdo a los datos asignados por el constructor se implementa una función dentro de la misma clase nombrada Mov().

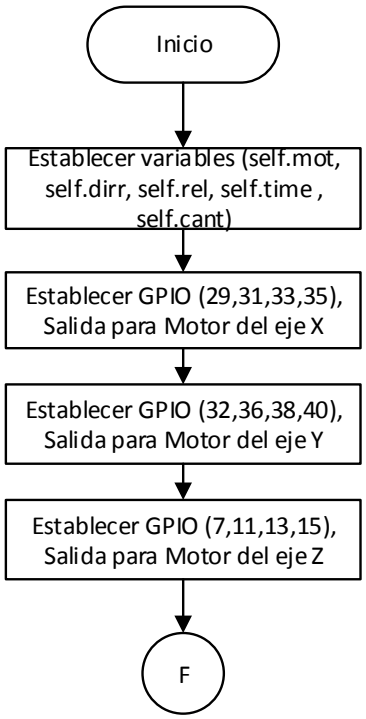


Figura 5.10 Procesos Iniciales

En la Figura 5.10 se aprecian los procesos que se requieren para iniciar la función Mov(), en los cuales se declaran las variables a emplear, se inicializan los pines GPIO de la Raspberry Pi como salidas(OUT) para comunicar a los manipuladores electrónicos de los motores a pasos. Las variables son asignadas mediante un constructor dentro de la clase mostrada en la Figura 5.11, el usuario mediante la interfaz ingresa los datos y estos son enviados a la clase y el método constructor establece los datos en las variables asignadas y el proceso continua en (F).

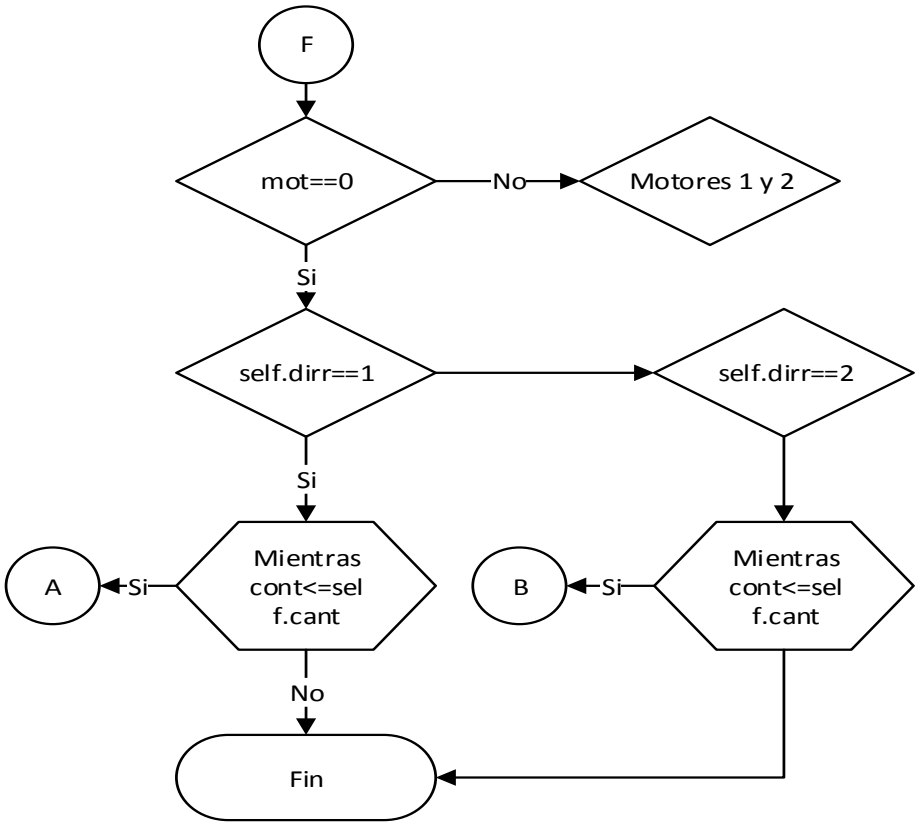


Figura 5.11 Proceso de selección y ejecución

De acuerdo a los datos ingresados como tipo de motor (eje "X", eje "Y" y eje "Z"), dirección de giro (de acuerdo al giro horario o anti-horario) se determina el motor que se activará y la dirección de rotación. Si el valor de la variable "mot = 0" se activa el motor que desplaza el eje "Y", de lo contrario se procede a verificar si "mot =1" corresponde al eje "X" y si "mot = 2" corresponde al eje "Z". Determinado motor como ejemplo "mot =0" que corresponde al eje "Y", se identifica cuál es la dirección de giro de acuerdo a la variable "dirr", si "dirr" es igual a 1 corresponde a un giro en dirección a las manecillas del reloj, este proceso de rotación se determina en la sección "A" y si "dirr" es igual a 2, la rotación procede de manera contraria determina en la sección "B" y este proceso se repite para cada motor.

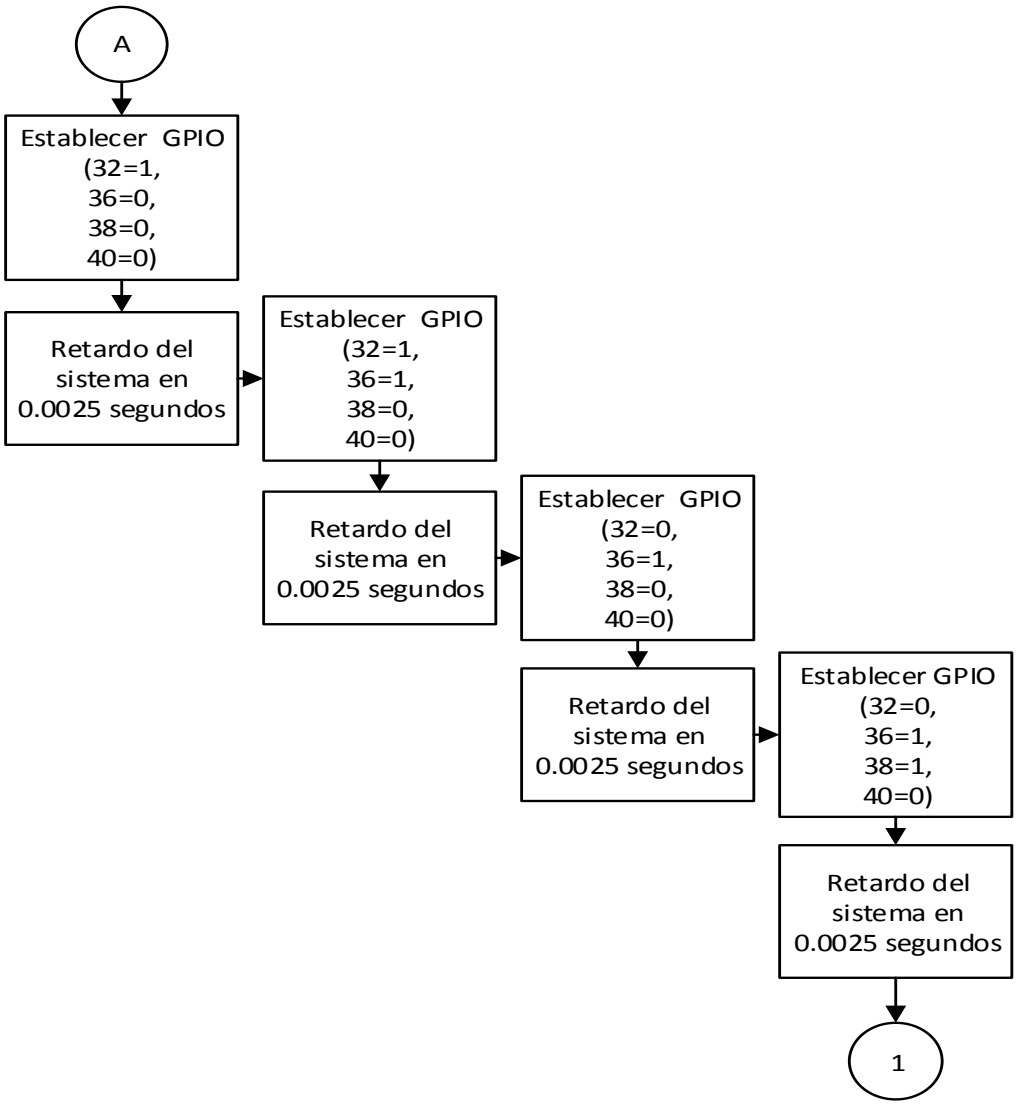


Figura 5.12 Proceso de rotación parte 1

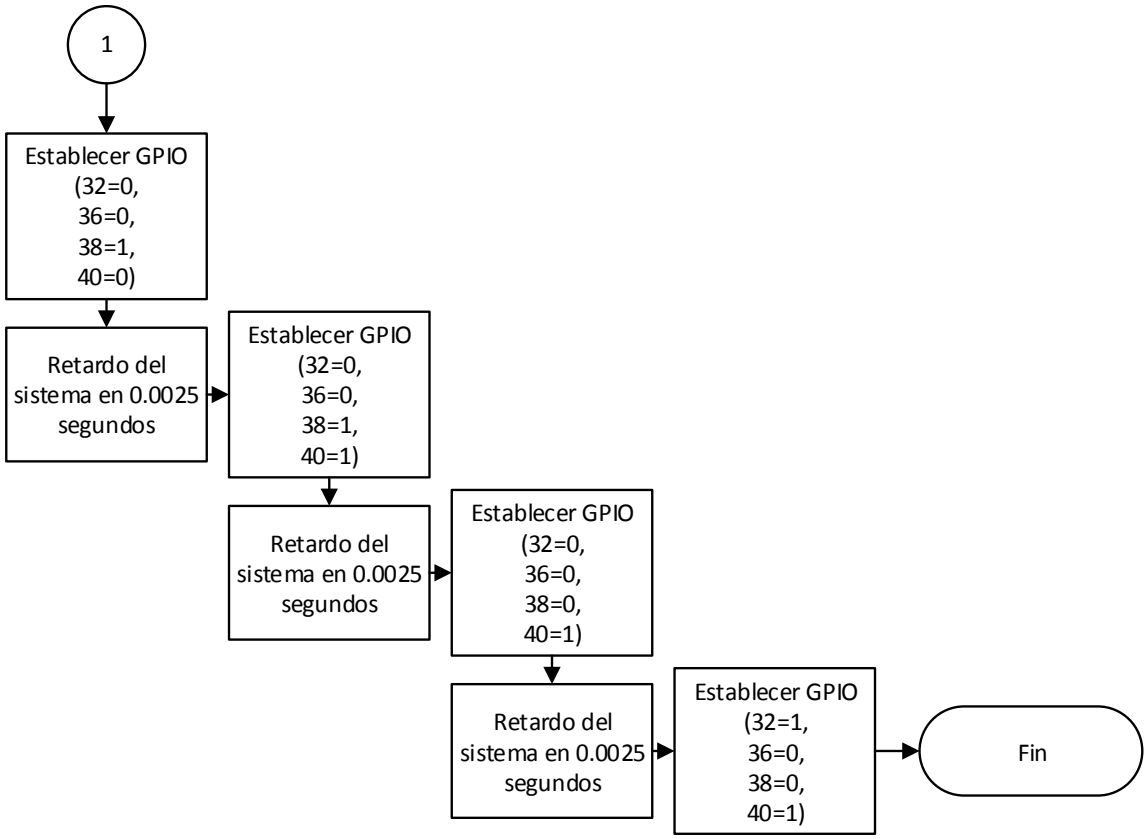


Figura 5.13 Proceso de rotación "A" parte 2

Las Figuras Figura 5.12 y Figura 5.13 presentan la secuencia para rotar el motor por medio de pasos en sentido horario presentando la activación de los pines GPIO en done "1" representa "3.3V" o "High" y "0" representa "0V" o "LOW". Se establece un retardo de 0.0025 segundos entre cada paso, ya que las bobinas requieren de ese tiempo para la carga y descarga, de acuerdo a la realización de pruebas con el tiempo de retardo entre cada paso, se determinó un tiempo mínimo (rotación rápida) de 0.0025 segundos y de 0.005 segundos, en donde se apreció una rotación correcta del motor. Para generar la secuencia para la rotación anti-horaria, es necesario ejecutar la secuencia de rotación "A" de manera contraria.

De acuerdo a lo anterior, la clase movMot() representa de gran importancia y esencial para el funcionamiento del sistema, ya que permite que la Raspberry Pi pueda interactuar con el manipulador para motores a pasos. La clase está diseñada para que la interfaz manual y la interfaz automática puedan interactuar con dicha clase sin que se requiera implementar una clase para cada una de las interfaces.

De acuerdo al diagrama de clase presentado en la Figura 5.8, se implementa la clase “interfaceManual.py” la cual presenta los elementos que se aprecian en la figura 5.14.

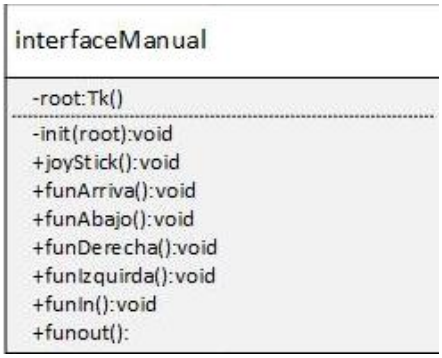


Figura 5.14 Clase interfaceManual.py

La clase “interfaceManual.py” se encuentra conformado por 7 métodos y un método constructor (Bloque de código que inicializa las variables y las configuraciones iniciales de la clase, funciones y características de Tkiner para visualizar la interfaz). En donde Tkinter es conocido como Tk y es un conjunto de herramientas para la creación de interfaces gráficas, Tk es multiplataforma, es decir, se puede ejecutar en cualquier otro sistema operativo (Linux, Windows, Mac O, entre otros). [10]

Figura 5.15 Interfaz para el manejo manual

La interfaz manual mostrada en la Figura 5.15, permite al usuario manipular tres ejes de la Máquina de Control Numérico, dentro de la interfaz el usuario debe de seleccionar que tipo de desplazamiento desea emplear (pasos, revolución o giro de 360° y en milímetros) y se requiere seleccionar la dirección a desplazar de acuerdo al eje por medio de botones,

“izquierda” y “derecha” corresponden el eje horizontal “X”, “arriba” y” abajo” corresponden al eje vertical “Y” y la dirección “In (Adentro)” y “Out (afuera)” al eje Z. También permite por medio de un botón habilitar el uso de un USB JoyStick para el manejo de los ejes a libertad como se presenta en la Figura 5.16.

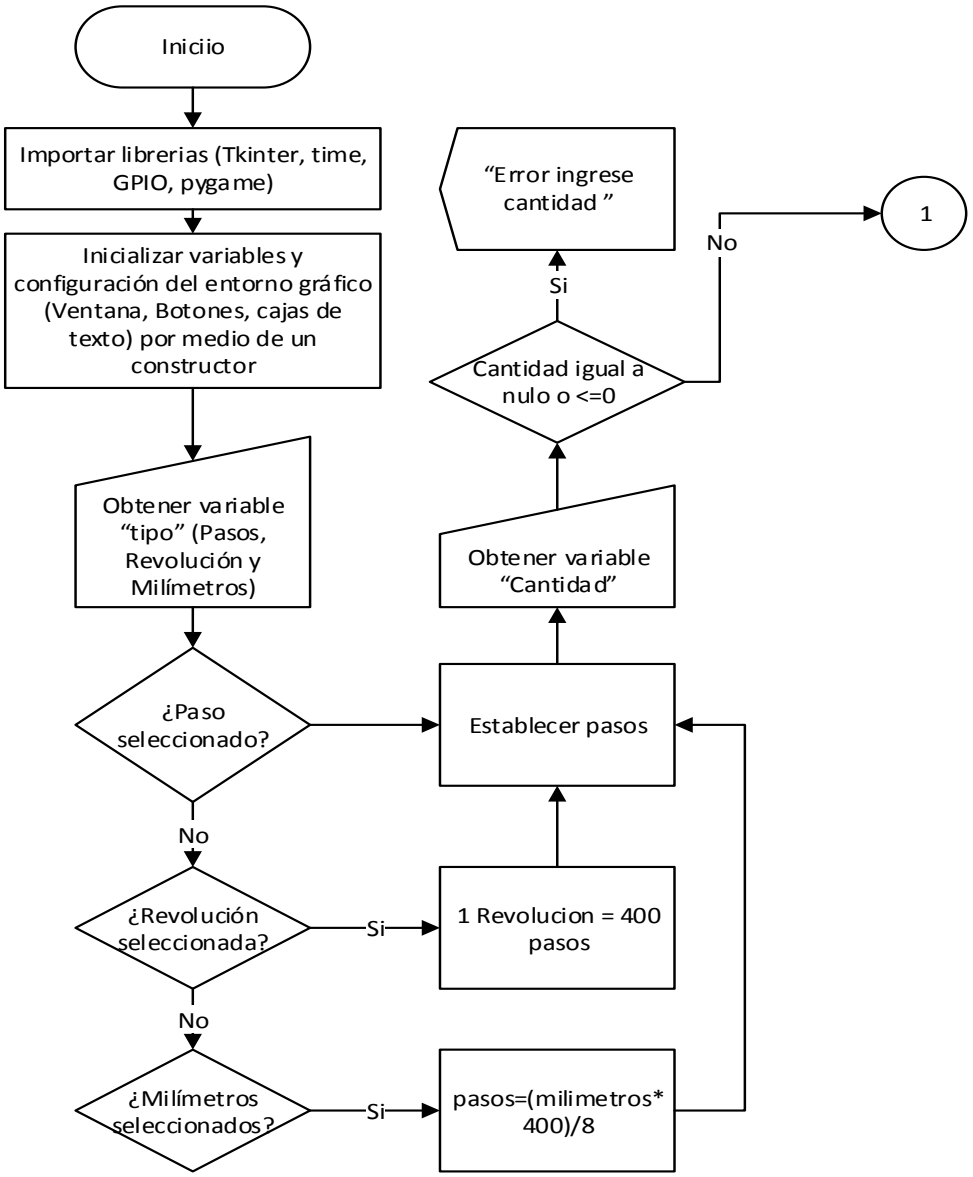


Figura 5.16 Diagrama de flujo general para la interfaz manual parte 1

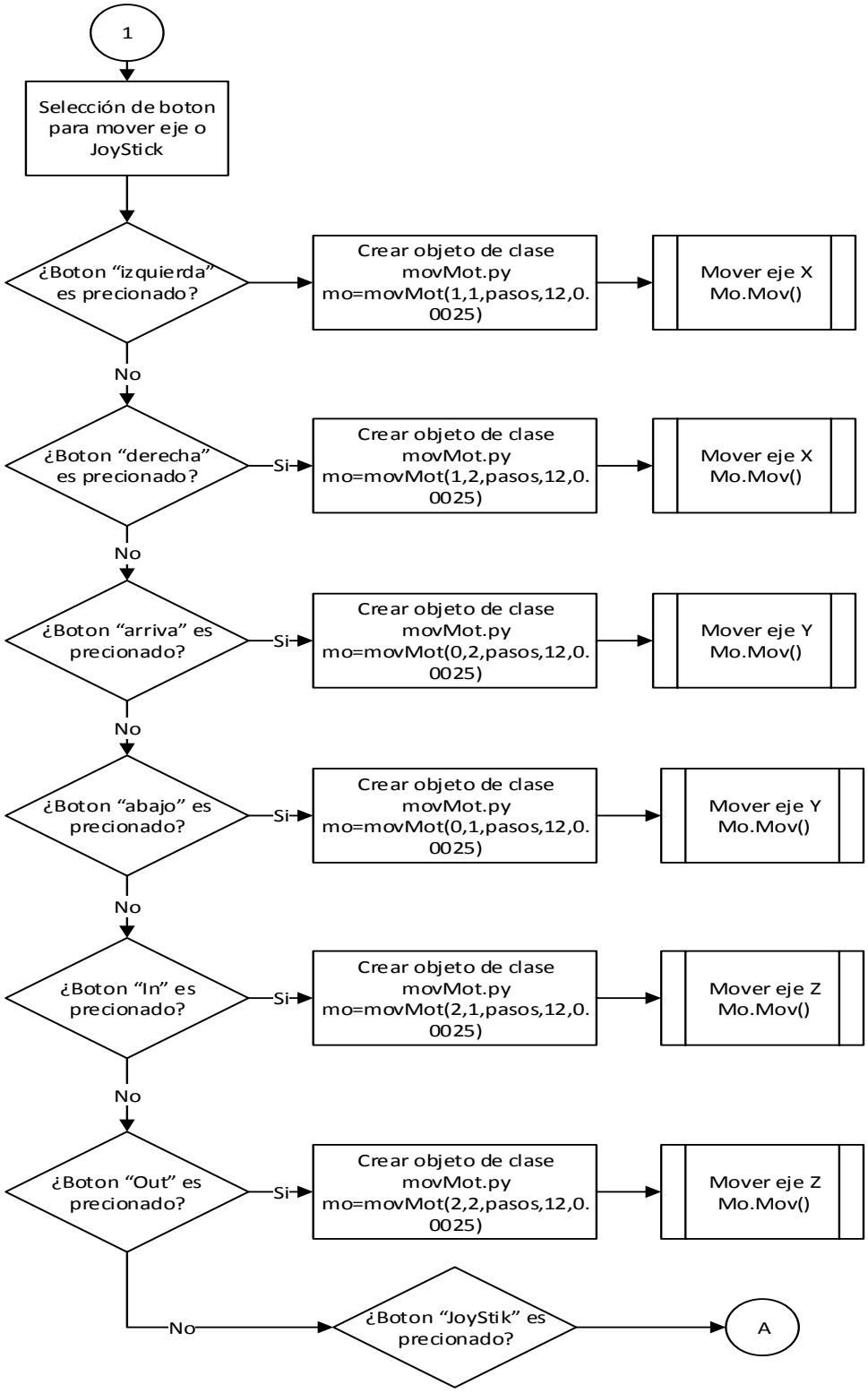


Figura 5.17 Diagrama de flujo general para la interfaz manual parte 2

De acuerdo a la Figura 5.17 , la interfaz es diseñada e implementada mediante el método constructor de la clase, en donde se definen y establecen los botones, botones radiales, cajas de texto y el diseño en general de la ventana. El usuario debe indicar por medio de botones radiales el tipo de desplazamiento a realizar (pasos, revolución y milímetros), pero para poder girar el motor, el sistema requiere convertir cualquiera de estas unidades a pasos:

- Si el usuario selecciona revolución de acuerdo a la cantidad ingresada:
    - 1 revolución = 400 pasos entonces:
        - Pasos = (cantidad * 400)
- Si el usuario selecciona milímetros de acuerdo con la cantidad ingresada:
    - 8 milímetros = 400 pasos entonces:
        - Pasos = (cantidad * 400) / 8mm

Establecido los pasos, el usuario puede seleccionar el eje a desplazar. El sistema incluye la clase movMot().py y como se mencionó anteriormente, esta clase contiene las instrucciones que permiten comunicar físicamente con el driver y este último con el motor para su rotación. Si el usuario selecciona el botón "JoyStick", se activa el manejo de los tres ejes mediante un control USB JoyStick y su funcionamiento de acuerdo al diagrama de clases en Figura 5.8 se extrae:

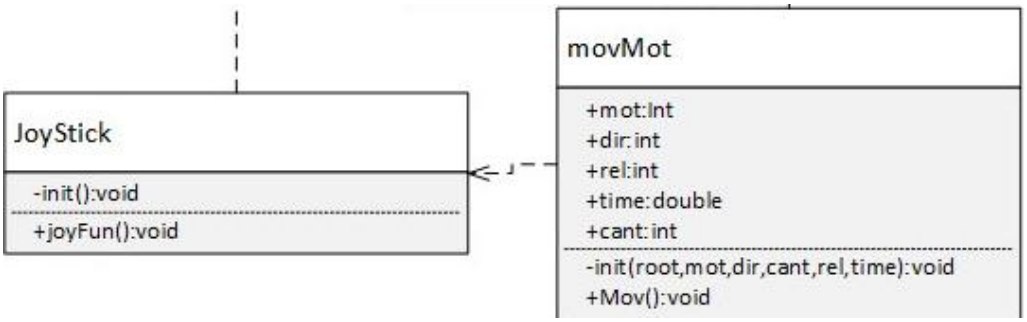


Figura 5.18 Clase JoyStick y dependencia de la clase movMot

En la Figura 5.18 se muestra la clase JoyStick, la cual depende directamente de la clase para comunicar con los motores de acuerdo al manejo en el control USB.

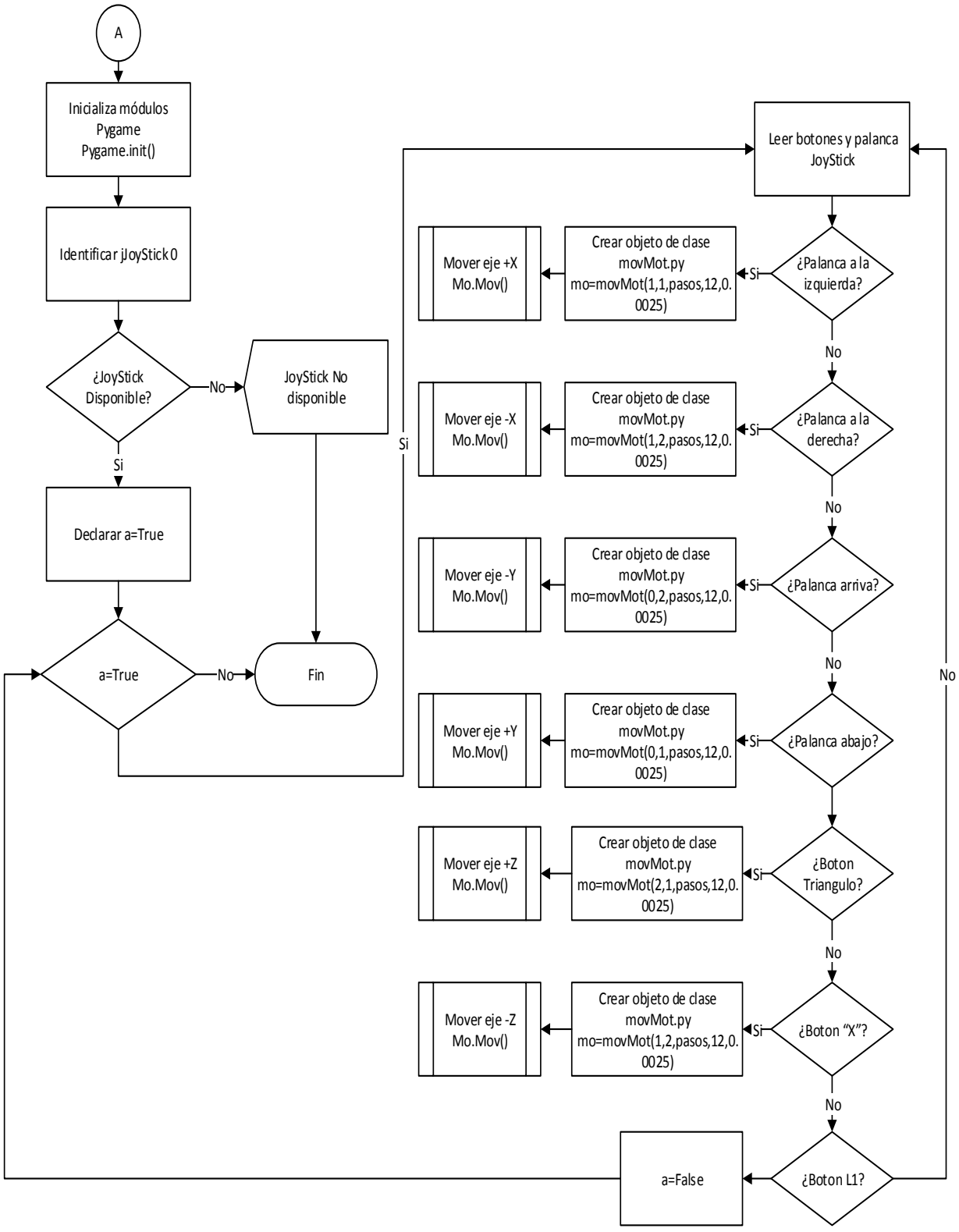


Figura 5.19 Continuación del diagrama para la interfaz manual, sección "A"

La Figura 5.19 muestra el funcionamiento de la clase JoyStick, en donde el sistema inicializa los métodos para la detección del control USB y su implementación. Si el

JoyStick no se encuentra disponible, le indica al usuario por medio de una advertencia que “el dispositivo no se encuentra disponible” y termina su ejecución. Si se encuentra disponible se establece un ciclo, que mientras “a=True” (Verdadero) lea los botones o la palanca manejados por el usuario en donde de acuerdo a la dirección que se le dé a la palanca (arriba, abajo, derecha, izquierda, in (adentro) y out (afuera)) sea el eje a desplazar, si el usuario aprieta la tecla (L1) se cambia el valor de “a=False” terminando la ejecución del programa.

El manejo automático es el proceso por el cuál, el usuario pueda establecer una rutina o una secuencia de desplazamientos entre los ejes y a diferencia del modo manual, el desplazamiento de los ejes se encuentra definido en milímetros. El sistema requiere de la cantidad de milímetros a desplazar y las veces que el usuario requiera repetir la secuencia.

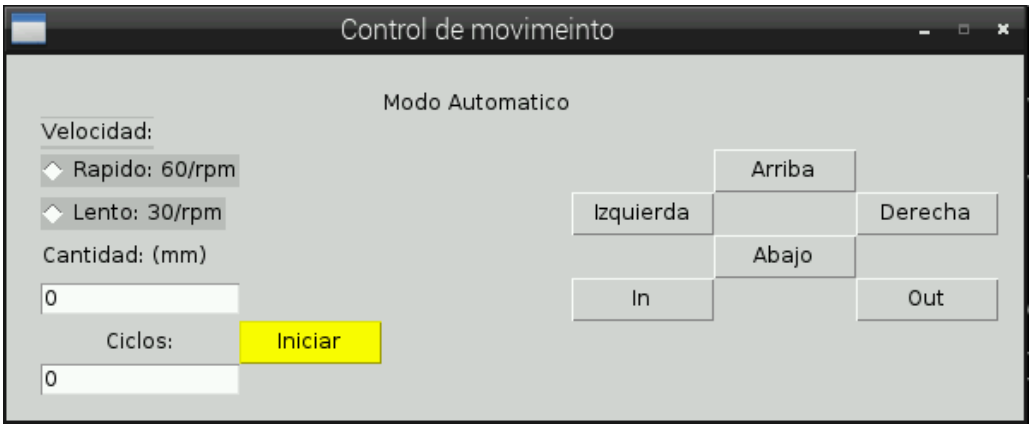


Figura 5.20 Interfaz gráfica para el manejo automático

En la Figura 5.20 se muestra la interfaz desarrollada para el manejo de automático, como se mencionó anteriormente, el sistema genera una secuencia que es establecida por el usuario mediante los botones. Los botones son identificados por un número asignado como se muestra en la Figura 5.21.

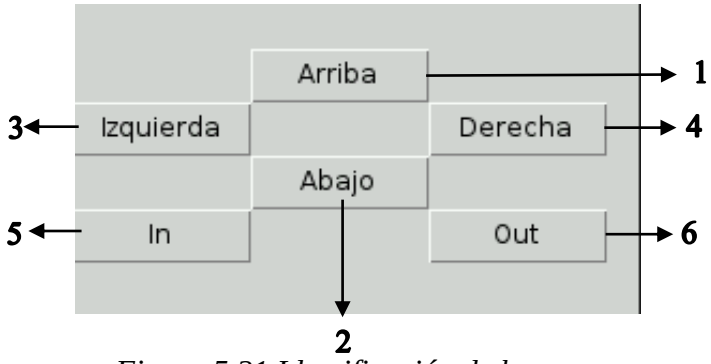


Figura 5.21 Identificación de botones

Los botones se encuentran identificados por números, el sistema genera una lista en el orden que se selecciona el botón y guarda el identificador del botón en esa lista por ejemplo si seleccionan los botones en el orden siguiente: Izquierda, Abajo, Arriba, Abajo y Out, generando:

lista_Mov= | 3 | 2 | 1 | 2 | 6 |

Este es el principio empleado para poder tres ejes de manera automática mediante la identificación de los ejes.

Para entender de mejor manera el funcionamiento, se presenta la estructura de la clase "interfazAutomática" y sus diagramas de flujo.

De acuerdo a la Figura 5.8 se extrae la clase interfaceAutomatica:

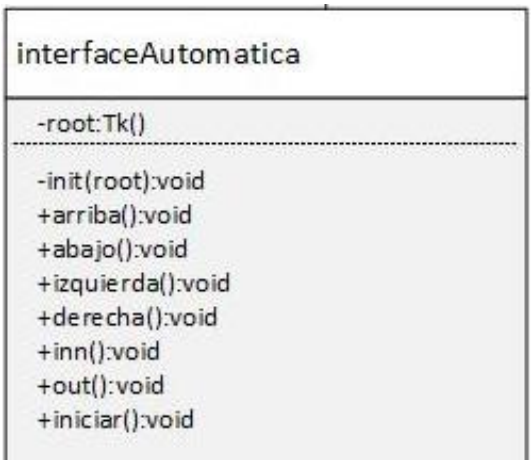


Figura 5.22 Clase interfaceAutomatica.py

La clase se encuentra conformado por 9 métodos y un constructor. El constructor inicializa los métodos gráficos para la creación de la interfaz gráfica. Los métodos (arriba, abajo,

izquierda, derecha, in y out) son activados mediante los botones correspondientes a la interfaz gráfica. El método iniciar determinar de acuerdo a la secuencia ya establecida, el motor a activar, cantidad de pasos, tiempo definido por el usuario mediante la interfaz y la cantidad de ciclos que se desea repetir la secuencia.

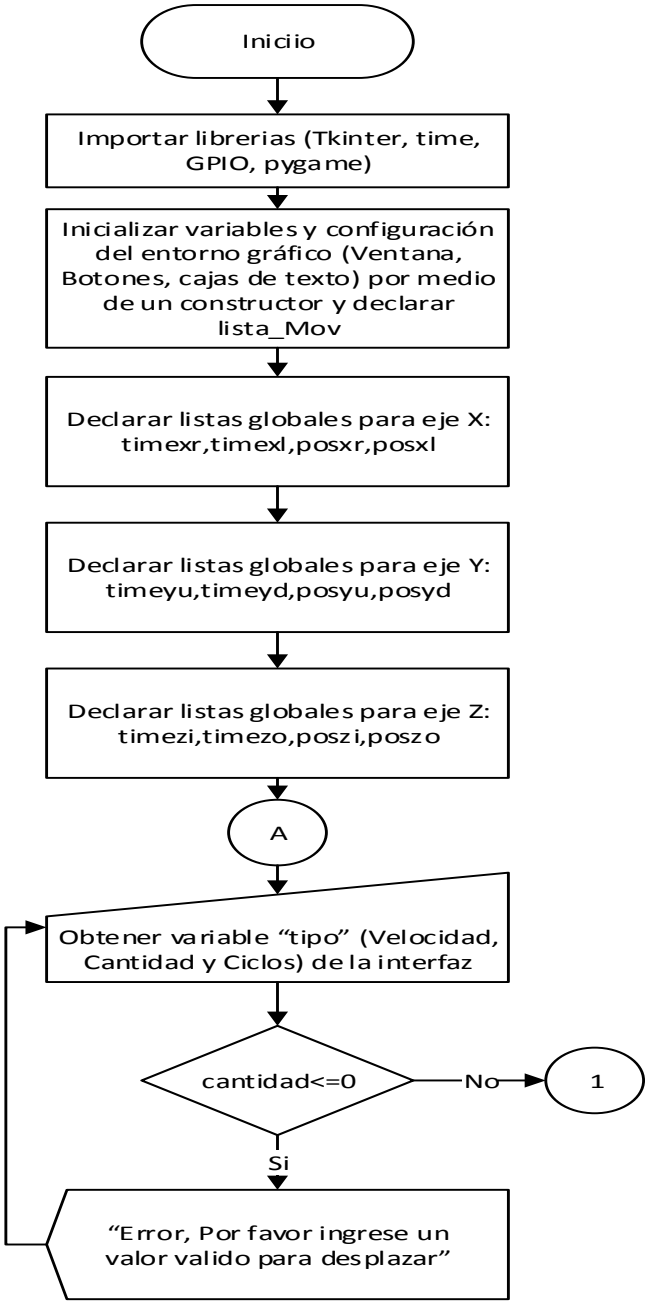


Figura 5.23 Diagrama General de la interfazAutomática.py parte 1

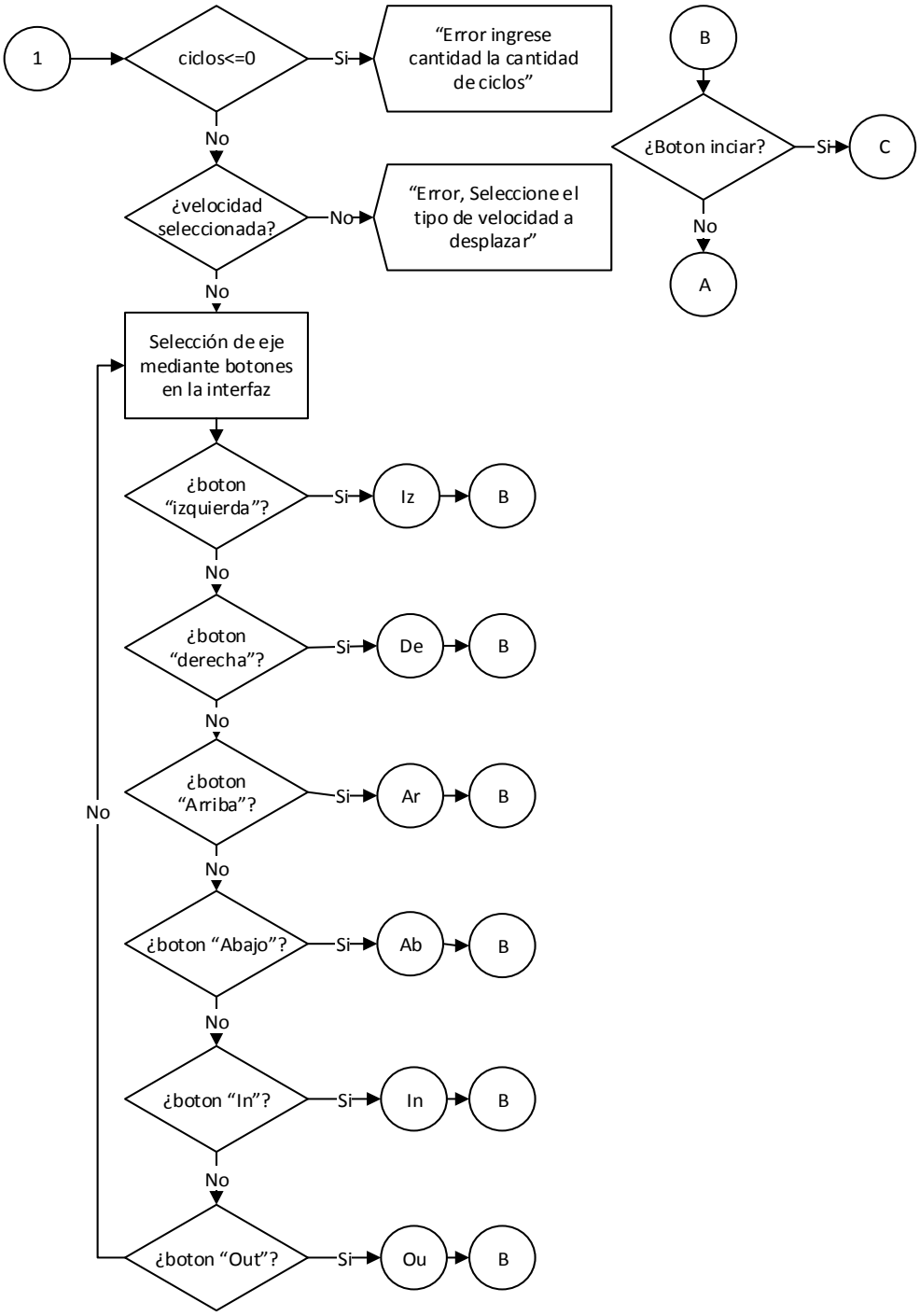


Figura 5.24 Diagrama General de la interfazAutomática.py parte 2

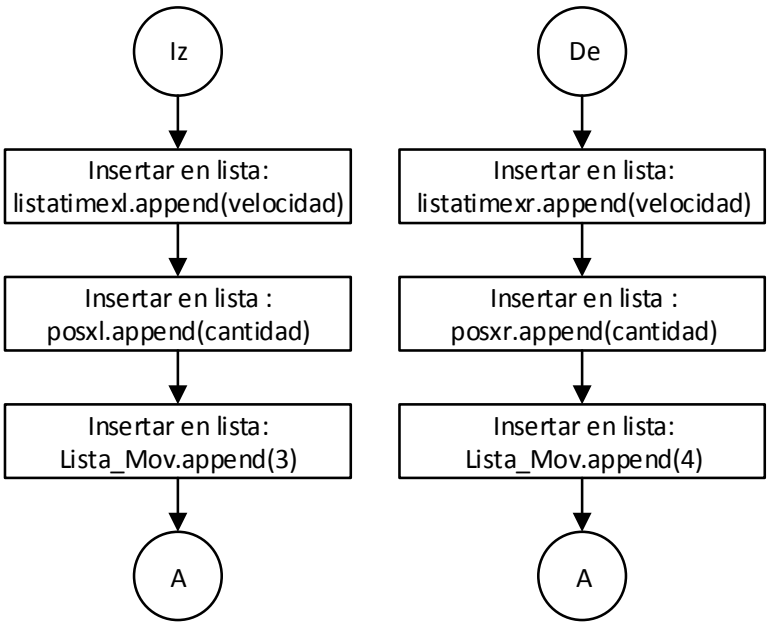


Figura 5.25 Índices "Iz" y "De" con respecto al diagrama anterior

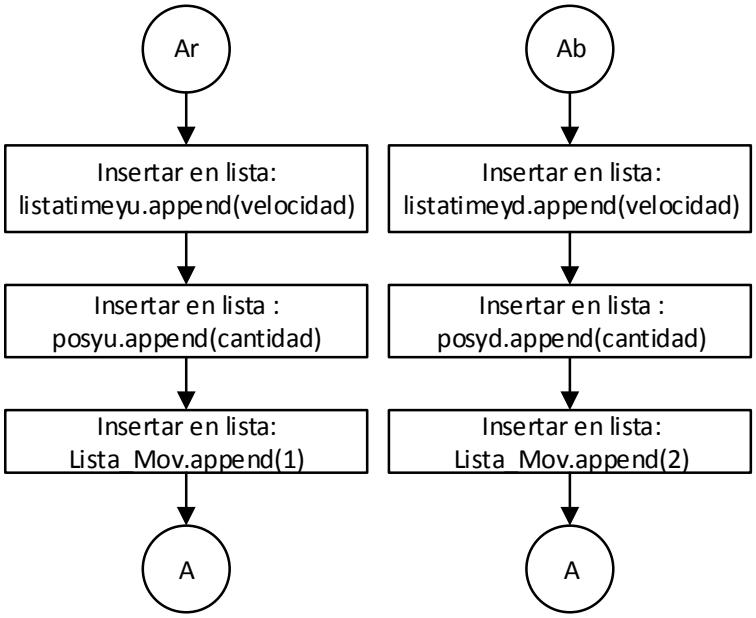


Figura 5.26 Índices "Ar" y "Ab" con respecto al diagrama anterior

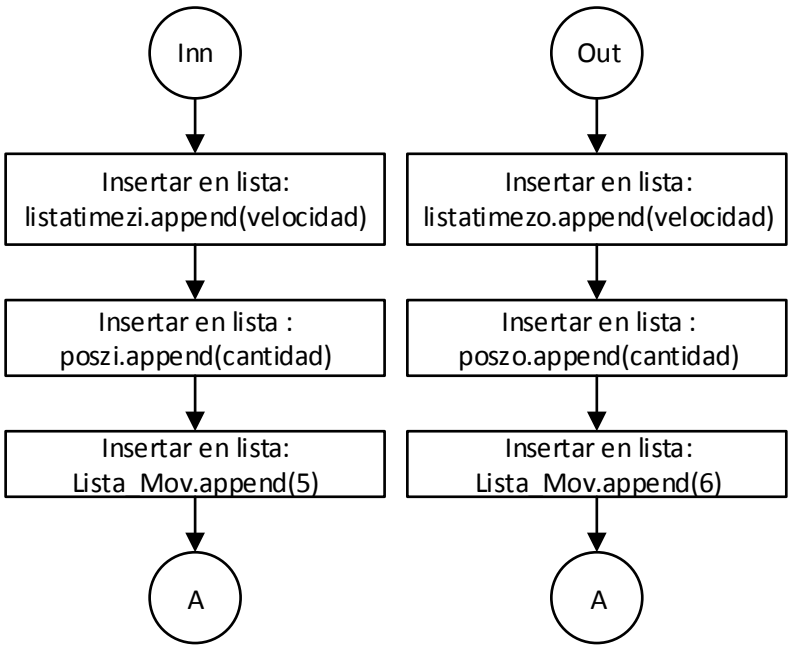


Figura 5.27 Índices "Inn" y "Out" con respecto al diagrama anterior

La Figura 5.23 muestra el proceso para automatizar movimientos en los ejes. Se define como el proceso inicial, la interfaz gráfica (botones, radio botnes y cajas de texto), se declaran las siguientes listas:

- Lista para la almacenar la cantidad de milímetros a desplazar por dirección de los ejes X (izquierda, derecha), Y (arriba, abajo) y Z (adentro, afuera), en total 6 listas.
- Lista para almacenar la posición la secuencia de los botones seleccionados.

Se implementan condiciones para la:

- Selección de la velocidad: Si el usuario no indica la velocidad a desplazar el sistema le informa mediante una ventana emergente.

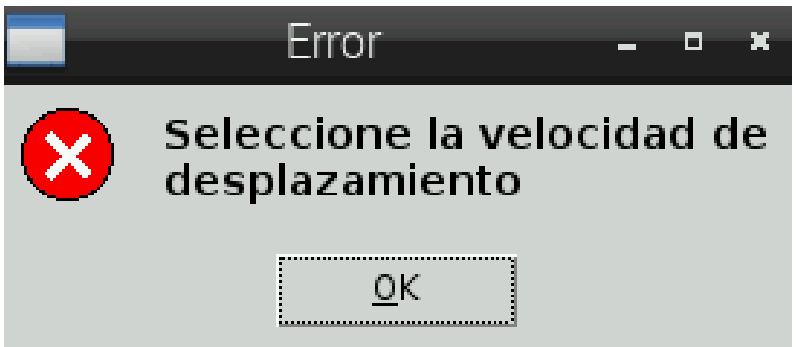


Figura 5.28 Ventana de error para la selección "Velocidad"

- Ingreso del valor para desplazar: Si el usuario no ingresa un valor mayor a cero, el sistema despliega una ventana de error para informarle al usuario

Figura 5.29 Ventana de error para el ingreso del valor "Desplazamiento"

- Ingreso del valor ciclos: Si el usuario no ingresa un valor mayor a cero, el sistema despliega una ventana de error para informarle al usuario.

Figura 5.30 Venta de error para el ingreso del valor "Cantidad"

Si el ingreso de los valores es correcto, se requiere seleccionar el eje e indicar a que dirección desplazar mediante los botones en la interfaz gráfica, el sistema guarda la dirección y los valores ingresados para esa dirección y este proceso continua llenando las listas con las secuencias de los botones seleccionados (Figura 5.25, Figura 5.26, Figura 5.27 hasta que el usuario seleccione la tecla "iniciar", la cual inicia el proceso de ejecución de la secuencia por medio del método "iniciar()".

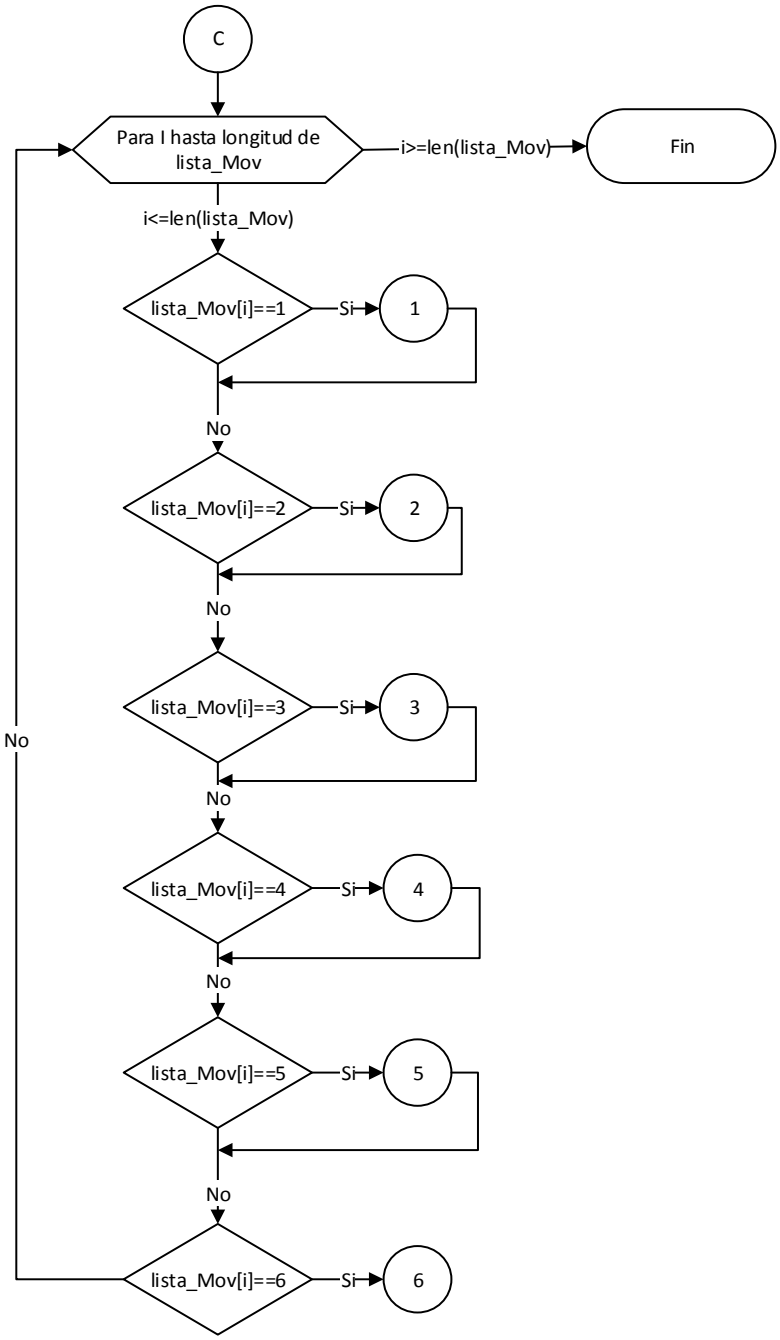


Figura 5.31 Método "iniciar"

La Figura 5.31 presenta el funcionamiento de la función "iniciar". Se recorre la lista "lista_Mov" hasta su longitud, ya que contiene la secuencia de todos los movimientos seleccionados, se extrae la secuencia y se verifica de uno a uno para identificar a que eje pertenece y desplazarlo. Se presenta las secciones de los índices dentro del diagrama

anterior. Las secciones procesan los datos obtenidos mediante la interfaz gráfica y genera la rotación en los ejes correspondientes a la secuencia establecida.

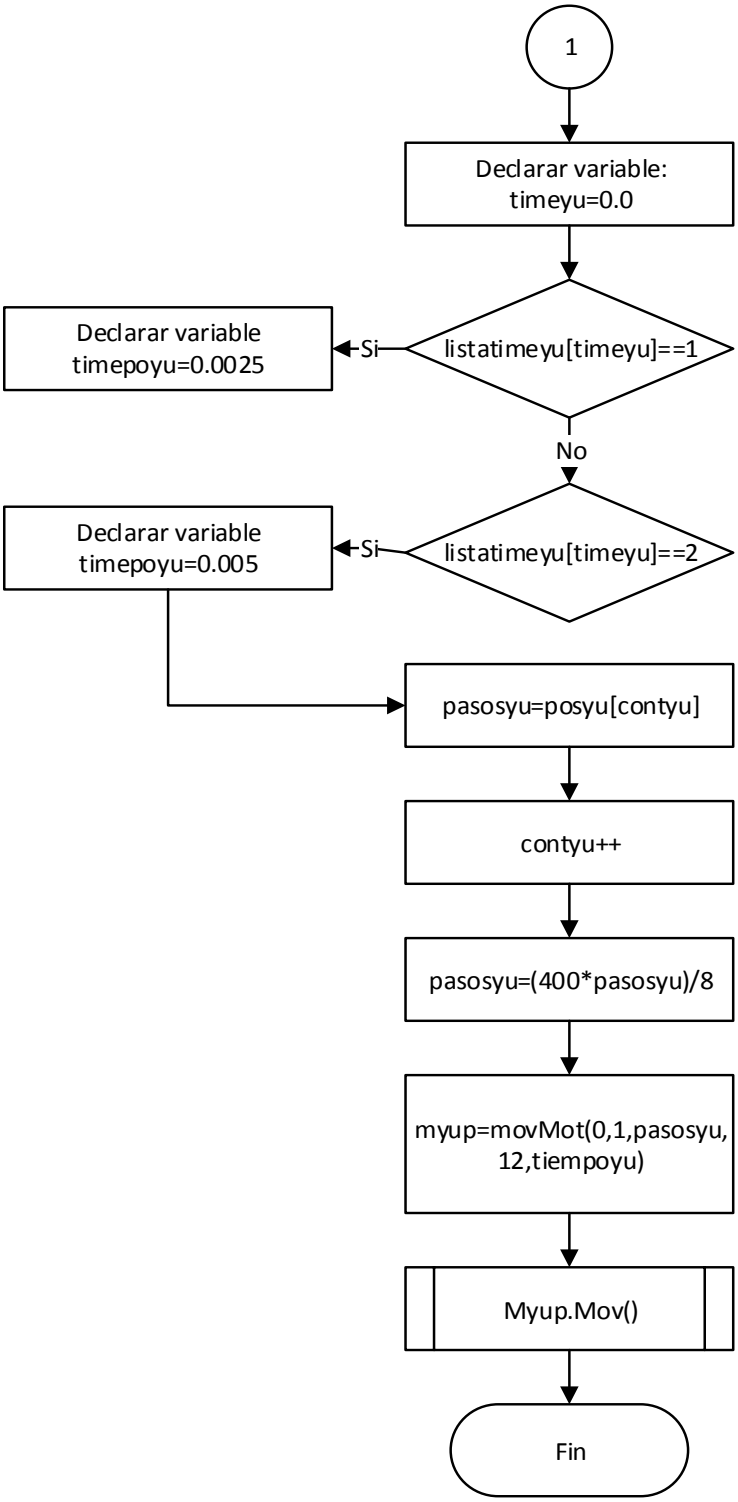


Figura 5.32 Sección 1 del diagrama método “iniciar”

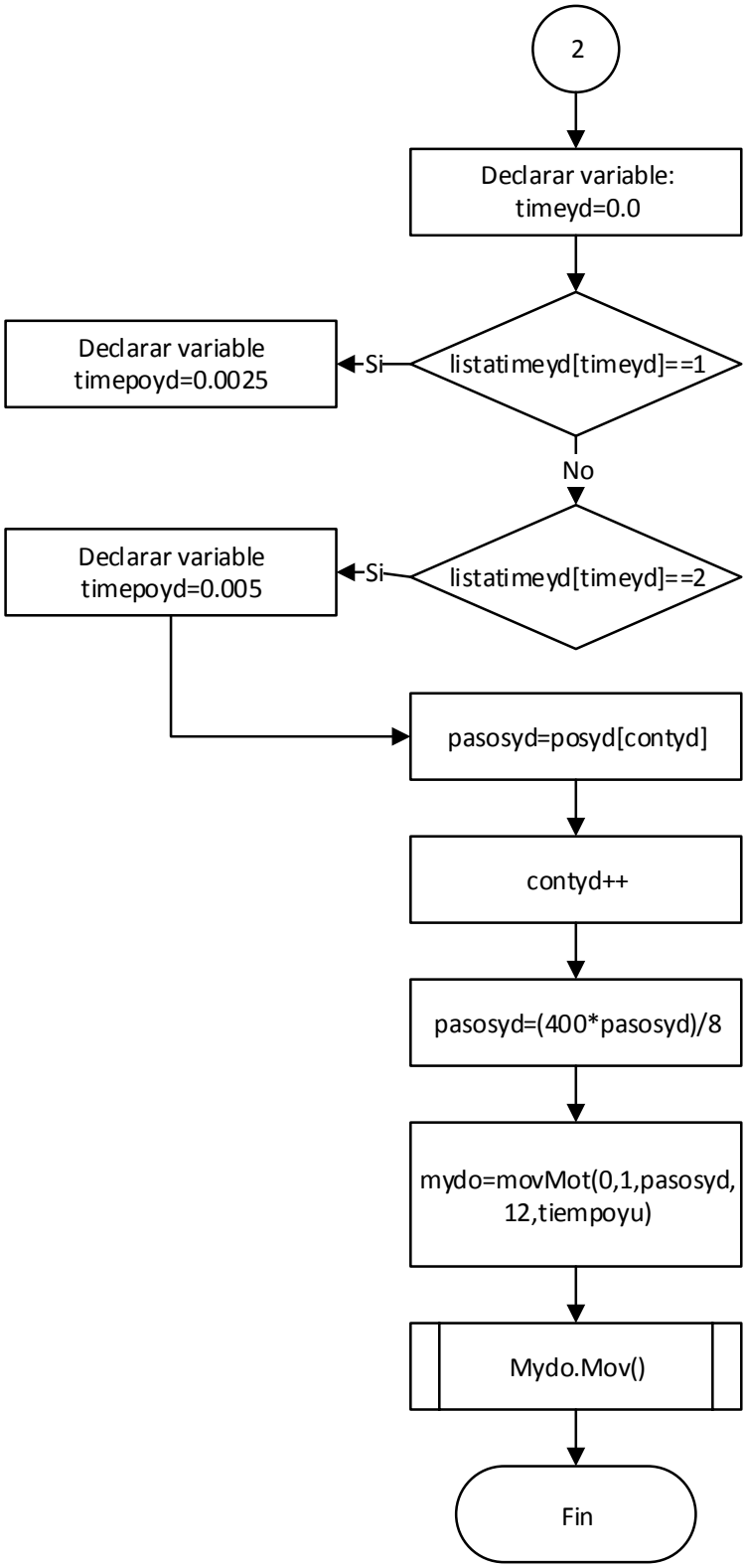


Figura 5.33 Sección 2 del diagrama método “iniciar”

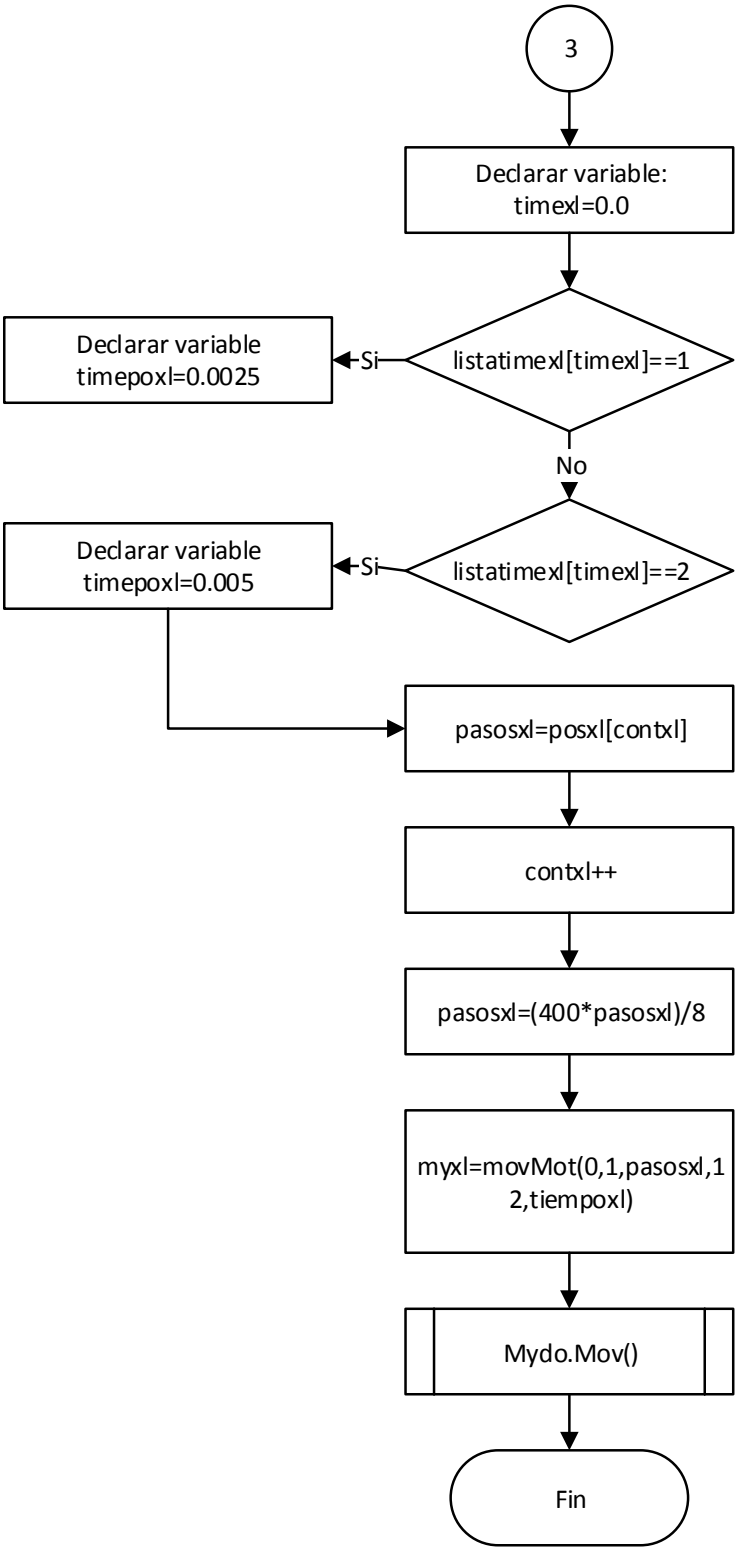


Figura 5.34 Sección 3 del diagrama método “iniciar”

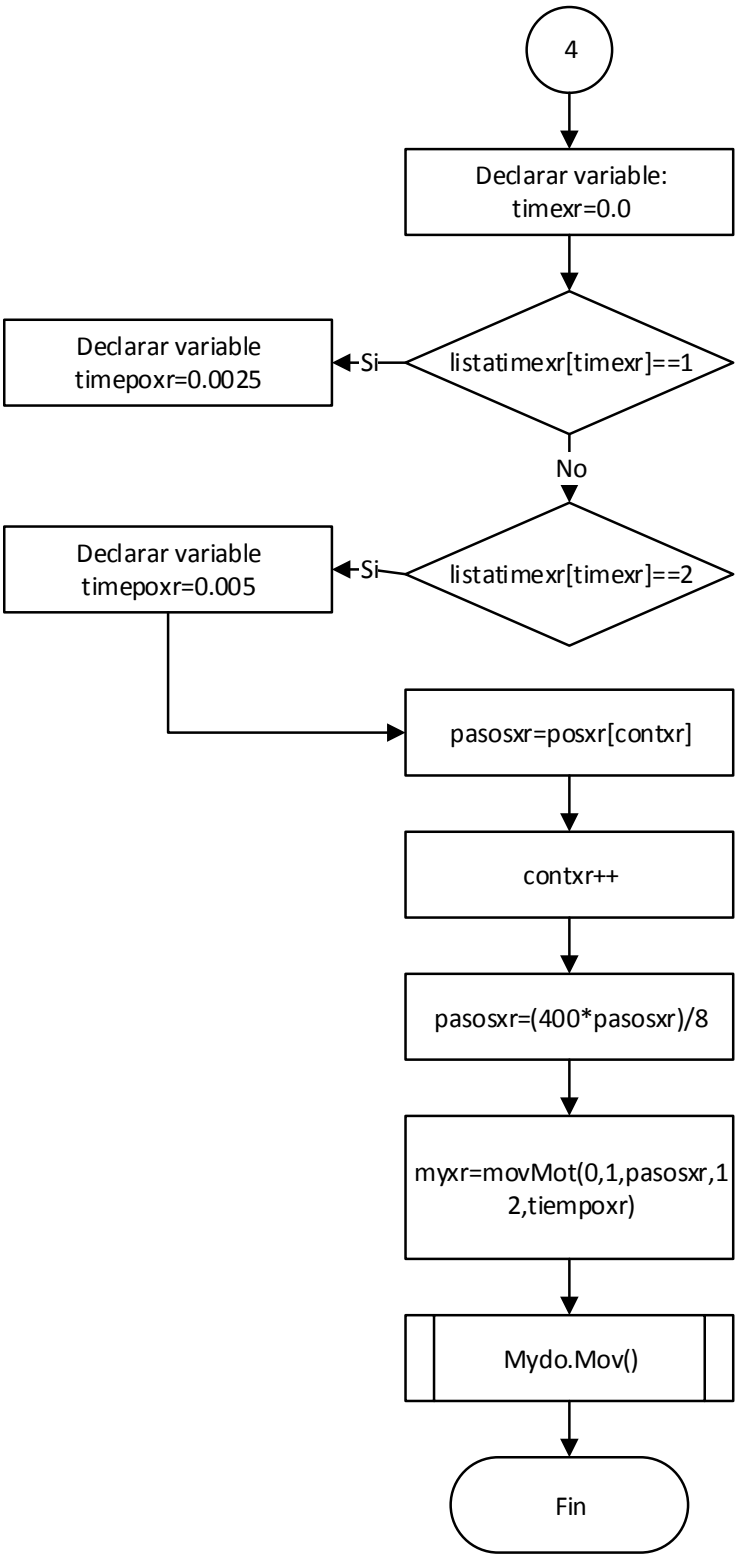


Figura 5.35 Sección 4 del diagrama método “iniciar”

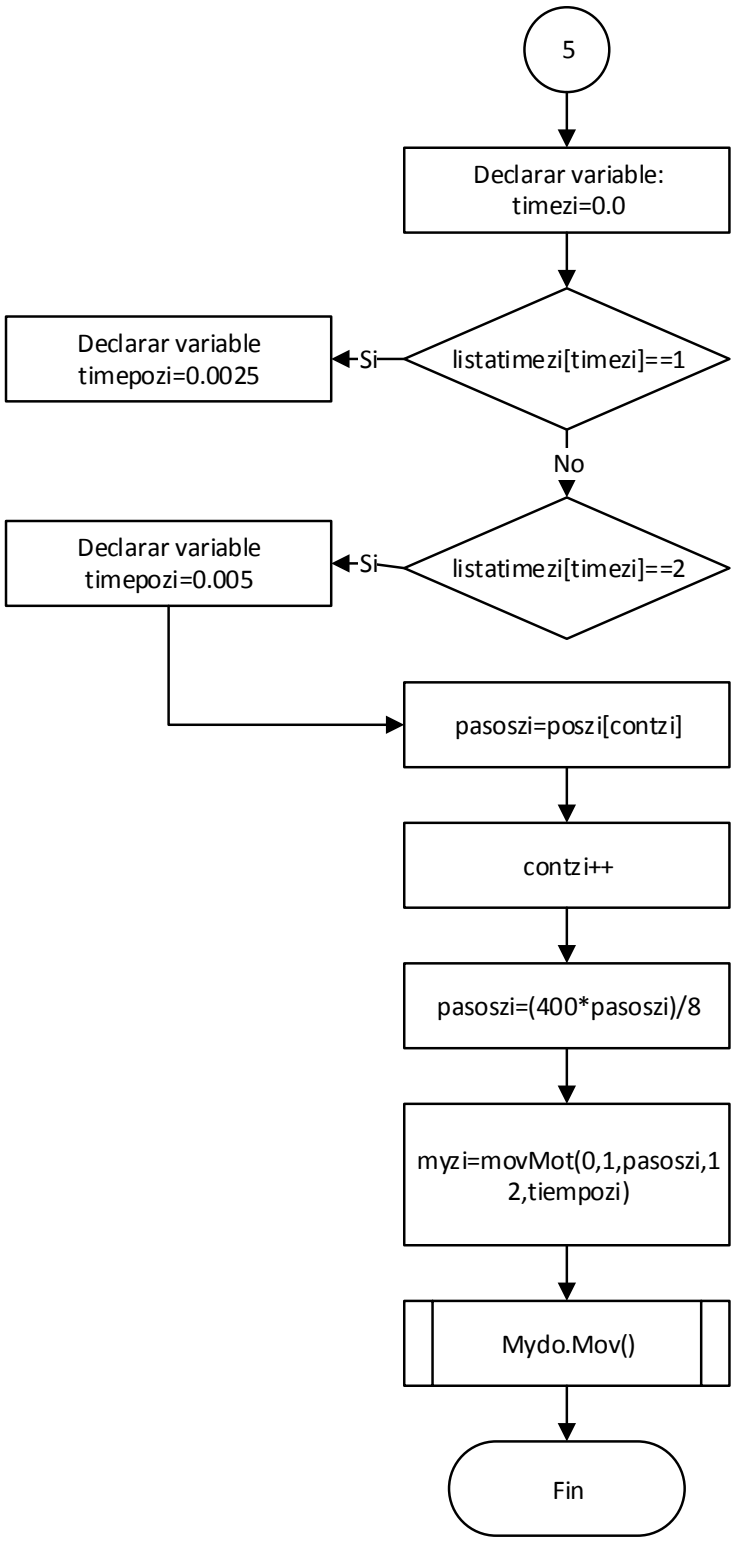


Figura 5.36 Sección 5 del diagrama método "iniciar"

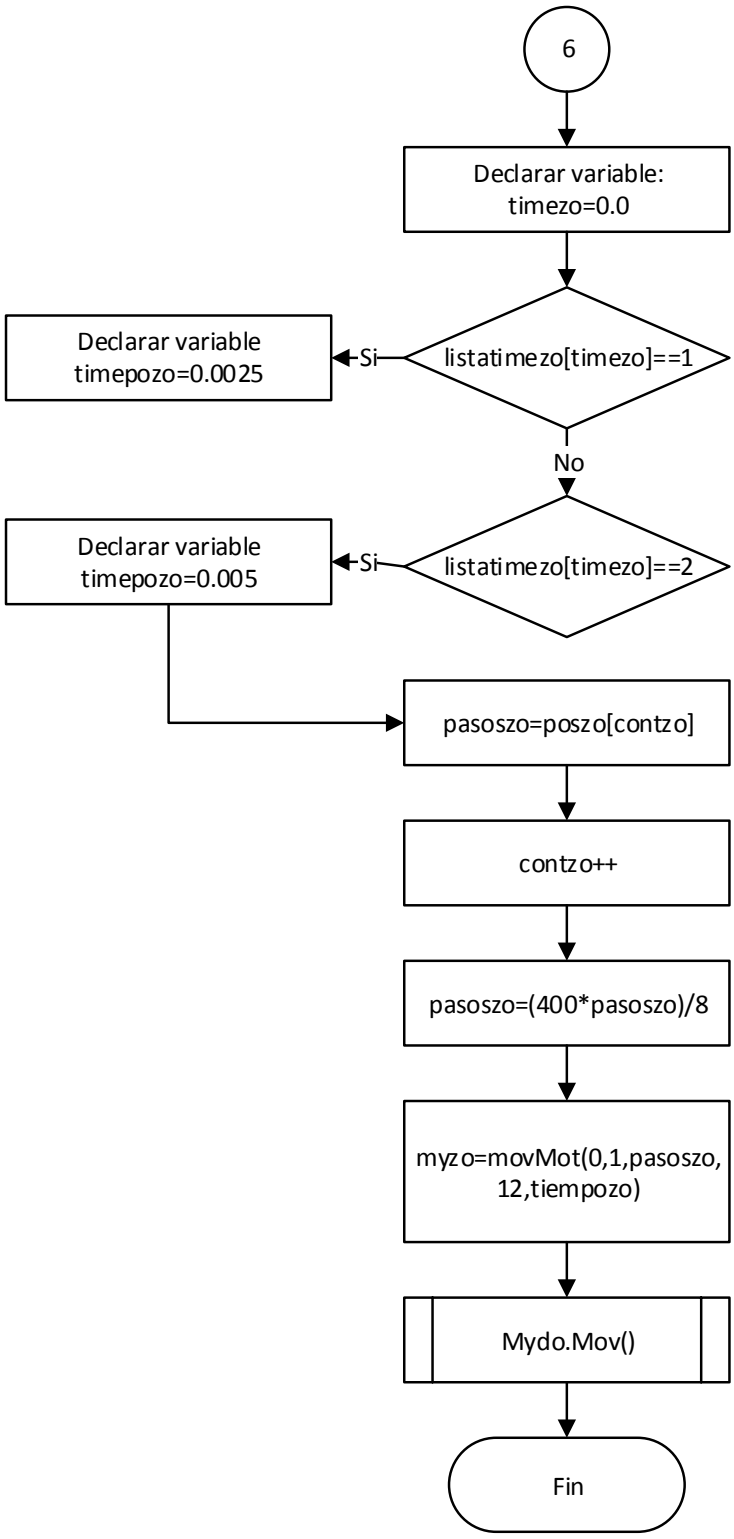


Figura 5.37 Sección 6 del diagrama método "iniciar"

5.3 Experimentación

Para verificar la precisión de desplazamiento del sistema y el tiempo que conlleva, se realizan las siguientes pruebas:

Se plantea un desplazamiento de 8 milímetros por segundo aproximadamente en cada uno de los ejes (“X”, “Y”, “Z”). Para comprobar dicho desplazamiento se emplea el uso de hojas milimétricas y un punzón, el punzón será empleado como el instrumento por el cual se realizarán marcas en la hoja milimétrica. Se indicara mediante la interfaz automática que:

- Se realicen desplazamientos en segmentos de 8 milímetros en el eje “X”, “Z”, y el eje “Y” realice una marca mediante el punzón en la hoja milimétrica, y se realice la medición de los segmentos entre punto a punto mediante un vernier.
- Se realicen desplazamientos aleatorios dentro del eje “X”, “Z”, y el eje “Y” realice una marca mediante el punzón en la hoja milimétrica y se realice la medición de los segmentos entre punto a punto mediante un vernier.
- Para el eje “Y” se coloque una tablilla vertical de unicel con una hoja milimétrica en segmentos de 8 milímetros y en desplazamientos aleatorios

Para obtener la cantidad de pasos que hay en cierta cantidad de milímetros con la siguiente ecuación:

$$Pasos = \frac{(milímetros * 400)}{8}$$

Para obtener el tiempo que tarda el eje en ejecutarse, para obtener dicho tiempo se emplea mediante software la función “time()”, que permite obtener la hora del sistema. La rotación del motor a pasos se encuentra definida dentro de la clase mov_Mot.py, dentro de ella se encuentra la función “Mov()” que contiene el proceso para ejecutar los pasos y comunicar mediante los pines de la Raspberry Pi al driver para los motores a pasos; entonces se obtiene la hora del sistema mediante la función “time()” antes la función “Mov()” que rota el motor a pasos y se obtiene de nuevo la hora del sistema, se resta el último tiempo con el tiempo que se adquirió antes de iniciar con la función “Mov()”.

Figura 5.38 Proceso para el contar tiempo de ejecución

Como se muestra en la Figura 5.38 se adquieren dos tiempos, uno antes de la función "Mov()" y otro al final, así se puede determinar mediante software el tiempo que tarda en rotar la cantidad milímetros desplazados.

6 RESULTADOS Y DISCUSIÓN

En este capítulo se presentan los resultados obtenidos en el desplazamiento de tres ejes “X”, “Y”, “Z” en segmentos de 8 milímetros y en diferentes distancias, como el tiempo que conllevó desplazar.

Se realizaron 3 pruebas con 10 distancias cada una en donde se realizó un desplazamiento en segmentos de 8 milímetros para el eje “X”. En la Prueba 1 se obtuvieron 7 distancias con un valor igual a 8 milímetros y 3 distancias con un error de 0.01 milímetros. En la Prueba 2 se obtuvieron 6 distancias con valor igual a 8 milímetros y 4 distancias con un error de 0.01 milímetros. En la Prueba 3 se obtuvieron 6 distancias con un valor de 8 milímetros y 4 distancias con un erro de 0.01 milímetros.

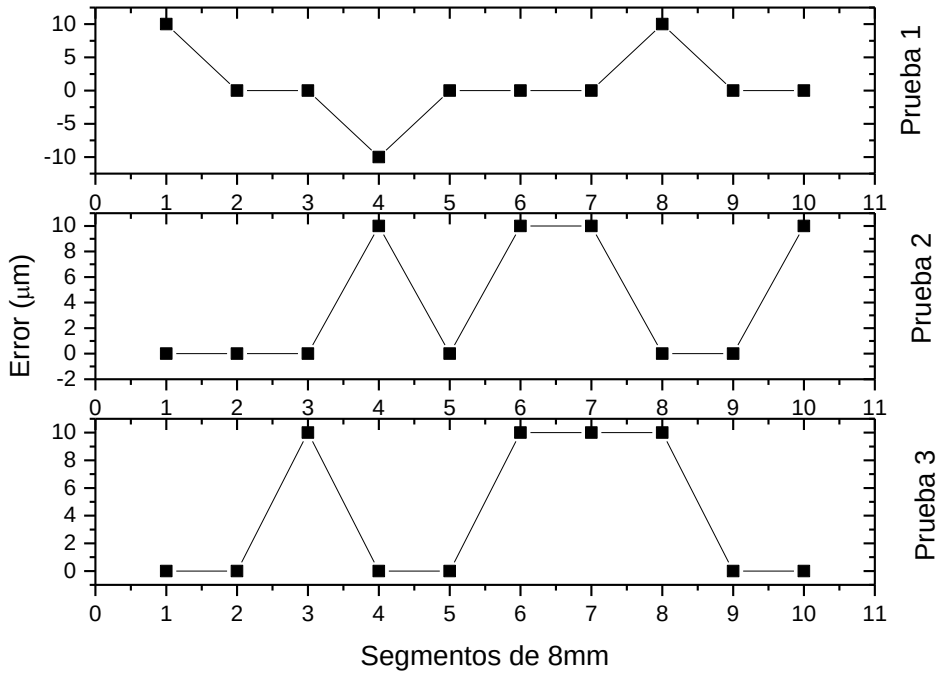


Figura 6.1 Error en pruebas de segmentos de 8 milímetros

En la Figura 6.1 el error obtenido en las 3 pruebas, donde aproximadamente cada 4 de 10 distancias para el desplazamiento en segmentos de 8 milímetros presentó un error de 10 micrómetros. Teniendo un promedio entre las 3 pruebas de 8.005 milímetros.

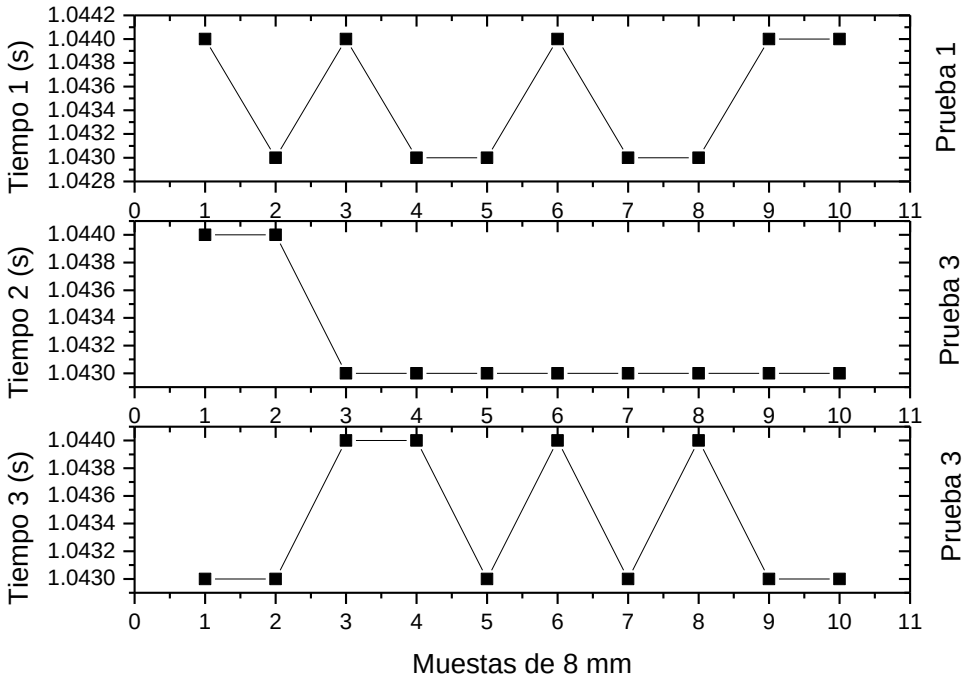


Figura 6.2 Pruebas de tiempo en desplazamiento de 8 mm

En la Figura 6.2 se presenta de acuerdo a las pruebas anteriores, sobre el desplazamiento en segmentos de 8 milímetros, se obtuvo el tiempo que requirió para dicho desplazamiento mediante software. En la primera prueba se obtuvo una variación de tiempo, donde en 5 distancias de 8 milímetros el tiempo de desplazamiento fue de 1.0440 segundos y las otras 5 restantes con un tiempo de 1.0430 segundos, siendo una diferencia entre ambas de 10 milisegundos. La segunda prueba presenta 2 distancias con un tiempo de 1.0440 segundos y los 8 restantes en 1.0430 segundos. Para la tercera prueba se obtuvo 4 distancias de 1.0440 segundos y 6 de 1.0430 segundos. En las 3 pruebas se encuentran variaciones de 10 milisegundos en las pruebas realizadas en el desplazamiento del eje “X” en segmentos de 8 milímetros y teniendo un tiempo promedio entre las tres pruebas de 1.04336 segundos.

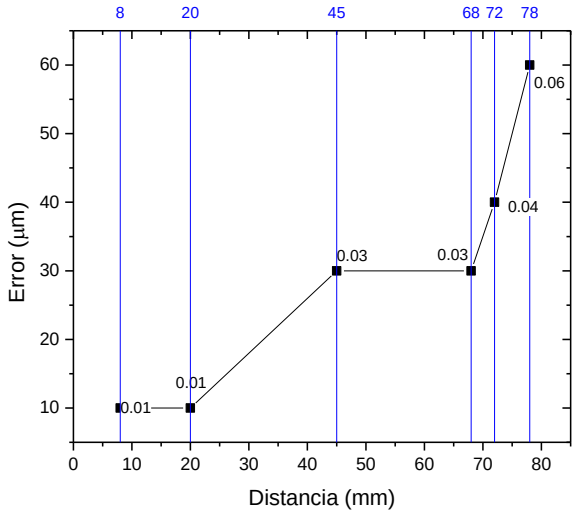


Figura 6.3 Error de desplazamiento de la prueba 1

En la Figura 6.3 se muestra el error de desplazamiento prueba 1 antes presentada, como se puede apreciar, en la primer distancia se obtiene un error de 10 micrómetros ó medio paso entre las distancias de 10 y 20 milímetros, ente las distancias de 45 y 68 milímetros se obtiene un error de 30 micrómetros o una pérdida de paso y medio, en un valor de desplazamiento de 72 milímetros se obtiene un error de 40 micrómetros o una pérdida de dos pasos y en un desplazamiento de 78 milímetros un error de 60 micrómetros ó 3 pasos de pérdida.

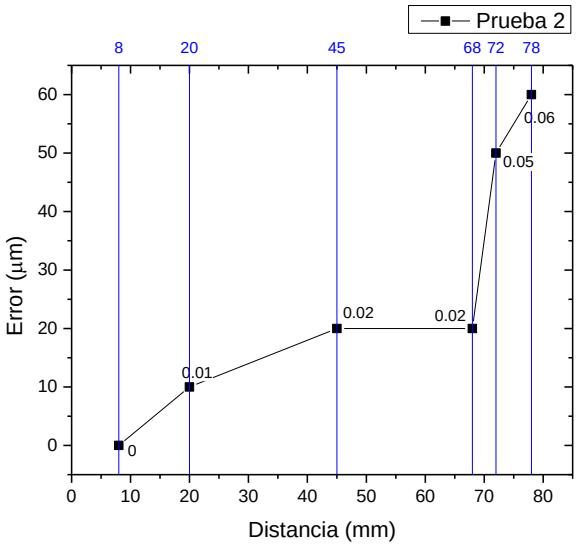


Figura 6.4 Error de la prueba 2 en el eje "X"

En la Figura 6.4 se presenta el error en la prueba 2, donde la primera distancia (8 mm) no presenta algún error, la segunda distancia (20 mm) presenta un error de 10 micrómetros, la tercer (45 mm) y cuarta distancia (68 mm) con un error de 20 micrómetros o la pérdida de un paso, la quinta distancia (72 mm) con un error de 50 micrómetros o una pérdida de dos pasos y la última distancia de 78 milímetros con un error de 60 micrómetros o una pérdida de tres pasos.

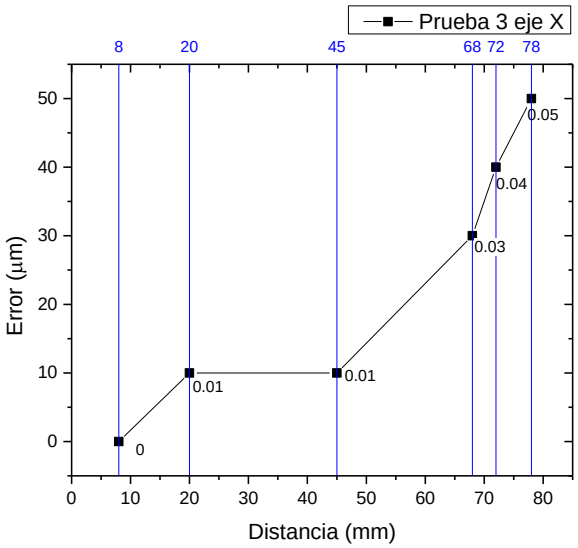


Figura 6.5 Error de prueba 3 en el eje "X"

La Figura 6.5 presenta el error de desplazamiento dentro de la tercer prueba realizado donde, la primer distancia de 8 mm no se obtuvo ningún error, la segunda de 20 mm se obtuvo un error 10 micrómetros al igual que la tercer prueba de 45 mm, la cuarta de 68 mm presenta un error de 30 micrómetros o una pérdida de un paso, la quinta prueba de 72 mm presenta un error de 40 micrómetros o una pérdida de dos pasos y la última prueba de 78 mm presenta un error de 50 micrómetros o una pérdida de dos pasos.

Entre las tres pruebas realizadas a diferentes distancias en el eje "X", se puede observar que mientras mayor sea la distancia a desplazar, entre una distancia menor a los 45 milímetros existe un error aproximado de 10 micrómetros, el cuál no se considera como pérdida ya que el paso mínimo del motor a pasos en el eje "X" es de 20 micrómetros.

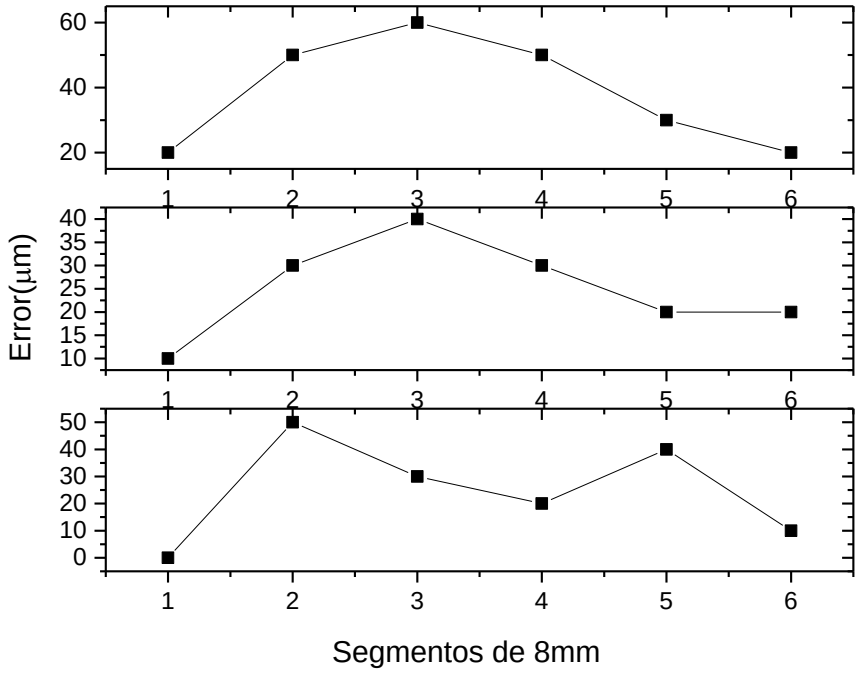


Figura 6.6 Pruebas en segmentos de 8mm en el eje Y

En la Figura 6.6 se presenta la realización de 3 pruebas con 10 muestras cada una en donde se realizó un desplazamiento en segmentos de 8 milímetros para el eje "Y". En la Prueba 1 se obtuvieron 4 distancias con un valor igual a 8 milímetros y 6 distancias con un error de 0.01 milímetros. En la Prueba 2 se obtuvieron 7 distancias con valor igual a 8

milímetros y 3 muestras con un error de 0.01 milímetros. En la Prueba 3 se obtuvieron 6 distancias con un valor de 8 milímetros y 3 distancias con un erro de 0.01 milímetros y una muestra con un error de 0.02 milímetros o la pérdida de un paso. Teniendo un promedio de las tres pruebas de 8.011 milímetros.

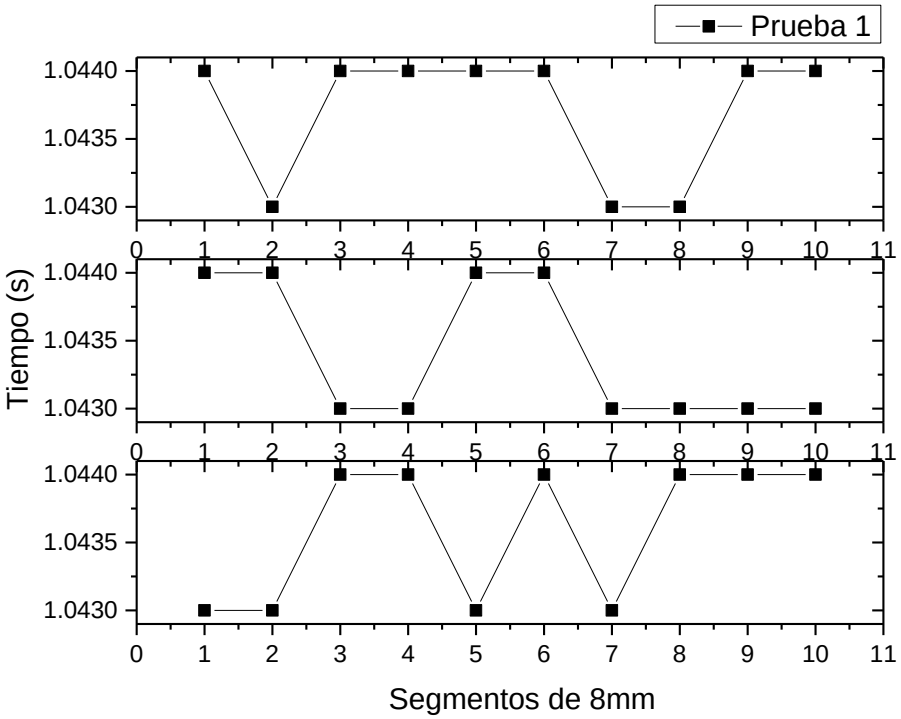


Figura 6.7 Prueba de tiempo en segmentos de 8 mm para el eje Y

De acuerdo a las pruebas anteriores sobre el desplazamiento en segmentos de 8 milímetros, se obtuvo el tiempo que requirió para dicho desplazamiento mediante software. En la primera prueba se obtuvo una variación de tiempo, donde en 7 distancias de 8 milímetros el tiempo de desplazamiento fue de 1.0440 segundos y las 3 restantes con un tiempo de 1.0430 segundos, siendo una diferencia entre ambas de 10 milisegundos. La segunda prueba presenta 4 distancias con un tiempo de 1.0440 segundos y las restantes en 1.0430 segundos. Para la tercera prueba se obtuvo 6 distancias de 1.0440 segundos y 4 de 1.0430 segundos. En las 3 pruebas se encuentran variaciones de 10 milisegundos en las pruebas realizadas en el desplazamiento del eje "X" en segmentos de 8 milímetros y teniendo un tiempo promedio entre las tres pruebas de 1.0439 segundos.

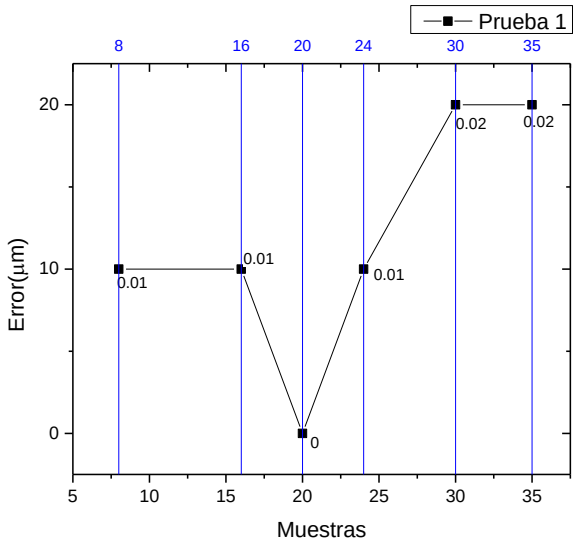


Figura 6.8 Error de prueba 1 de eje Y

La Figura 6.8 presenta el error de desplazamiento dentro de la primera prueba realizada donde, la primera distancia de 8 mm se obtuvo un error de 10 micrómetros al igual que la segunda distancia de 16 mm, en la tercera de 20 mm no se obtuvo error, la cuarta de 24 mm presenta un error de 10, la quinta de 30 mm presenta un error de 20 micrómetros o una pérdida de un paso al igual que la sexta distancia que representa una pérdida de un paso para las dos últimas distancias.

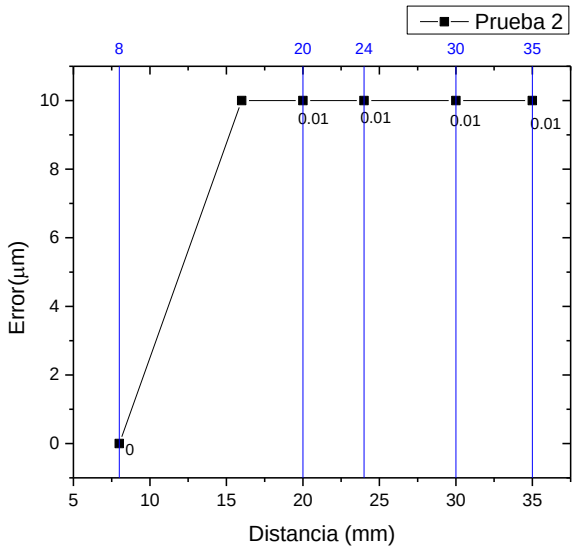


Figura 6.9 Error de la segunda prueba 2 en eje Y

La Figura 6.9 presenta el error de desplazamiento dentro de la segunda prueba realizada donde, la primera distancia de 8 mm no se presentó error, en la segunda distancia de 16mm, en la tercera, cuarta, quinta y sexta distancia se obtuvo un error de 10 micrómetros.

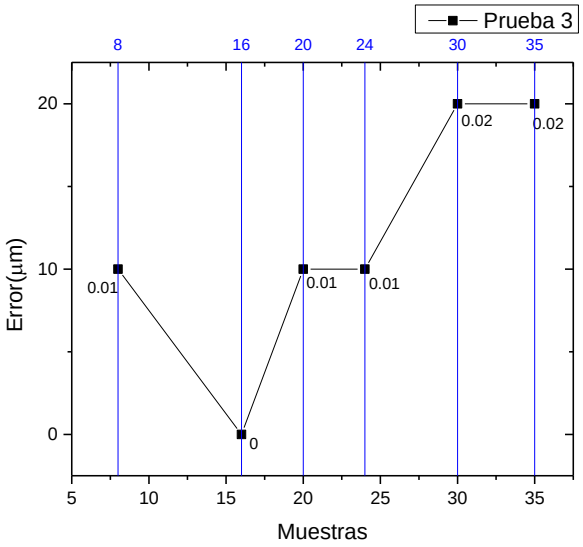


Figura 6.10 Error de la tercer prueba en eje Y

En la Figura 6.10 se muestra el error del desplazamiento, en donde se tienen tres distancias con un error de 10 micrómetros y dos distancias en donde el error es igual a la pérdida de un paso en las distancias desplazadas de 30 y 35 milímetros.

Dentro de las tres pruebas presentadas en el eje Y, dos de ellas presentan distancias con una pérdida de un paso en distancias mayores a 30 milímetros, siendo las demás distancias con valores que no intervienen con la perdida de paso del motor y genere una perdida en el desplazamiento en distancias menores a 30 milímetros.

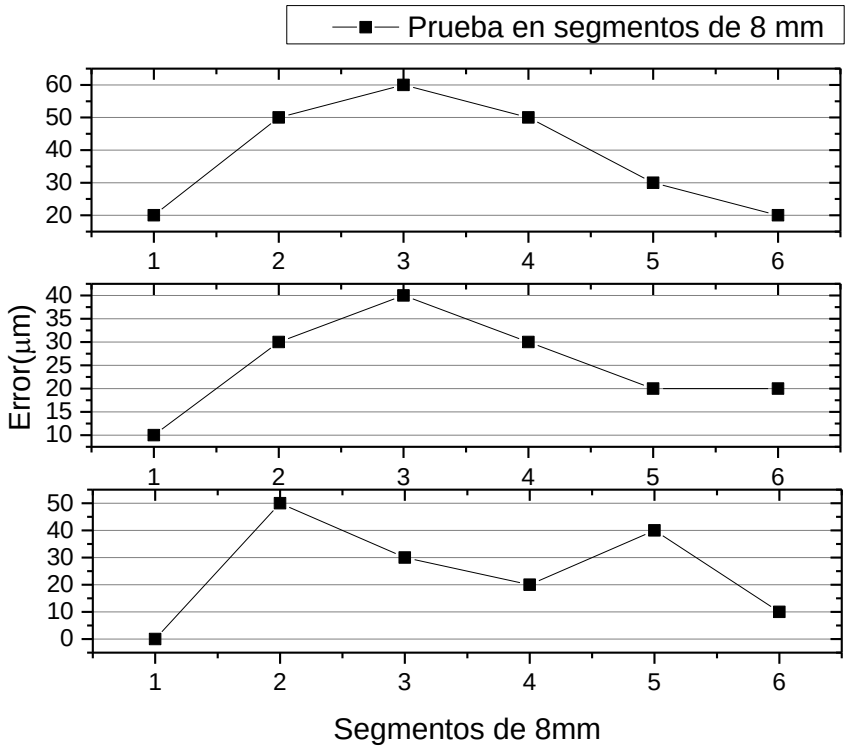


Figura 6.11 Error en las pruebas en segmentos de 8 mm

Como se presenta en la Figura 6.11, se presentan tres pruebas cada una con 6 distancias tomadas de desplazamientos en segmentos de 8 milímetros en el eje "Z". En la Figura 6.19 se presenta el error en el desplazamiento de dichas pruebas sobre cada distancia, se tiene un error máximo de 6 micrómetros o una pérdida de tres pasos, el promedio del error de las tres pruebas es de 8.0883 milímetros o 8 micrómetros4. que representa una pérdida promedio de 4 pasos en desplazamiento de 8 milímetros.

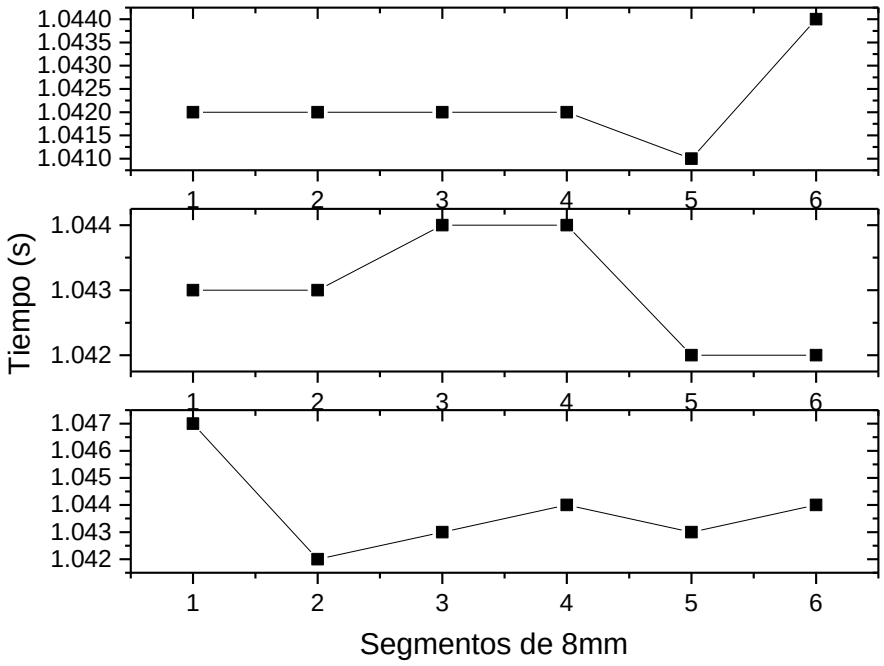


Figura 6.12 Prueba de tiempo en segmentos de 8 mm para el eje Z

En la Figura 6.12 se presenta de acuerdo a las pruebas realizadas anteriormente en segmentos de 8 milímetros para el eje "Z", se obtiene un promedio de acuerdo al desplazamiento en segmentos de 1.045 segundos.

También se realizaron pruebas del sistema automático en donde asignaron coordenadas para dibujar una figura cuadrada den 10x10 centímetros como se muestra en la siguiente figura:

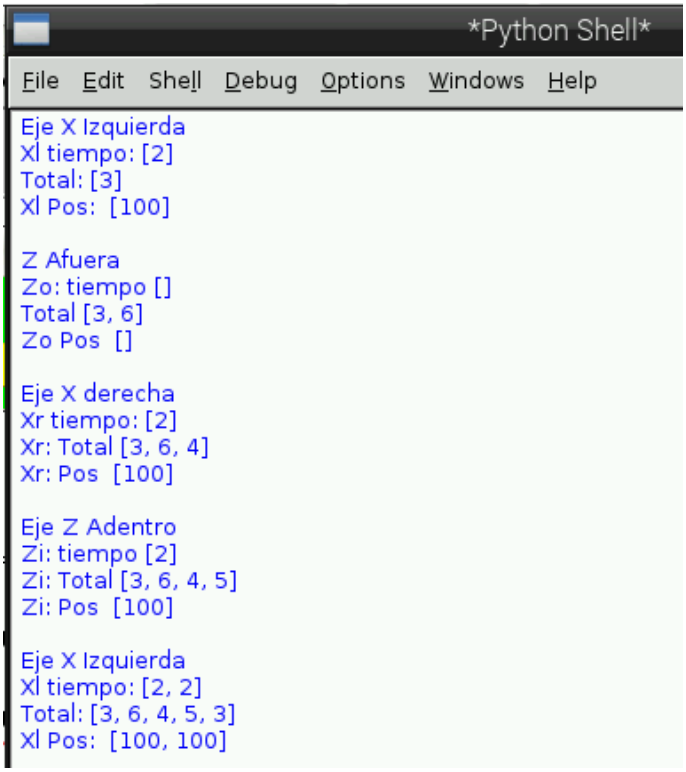


Figura 6.13 Secuencia del modo automático

La Figura 6.13 muestra una captura de pantalla sobre la secuencia que genera la interfaz automática al ir indicando el eje a desplazar su dirección, distancia y velocidad a desplazar. Primero se tiene un desplazamiento a la izquierda, afuera, a la derecha, adentro y al final a la izquierda para cerrar la figura cuadrada.

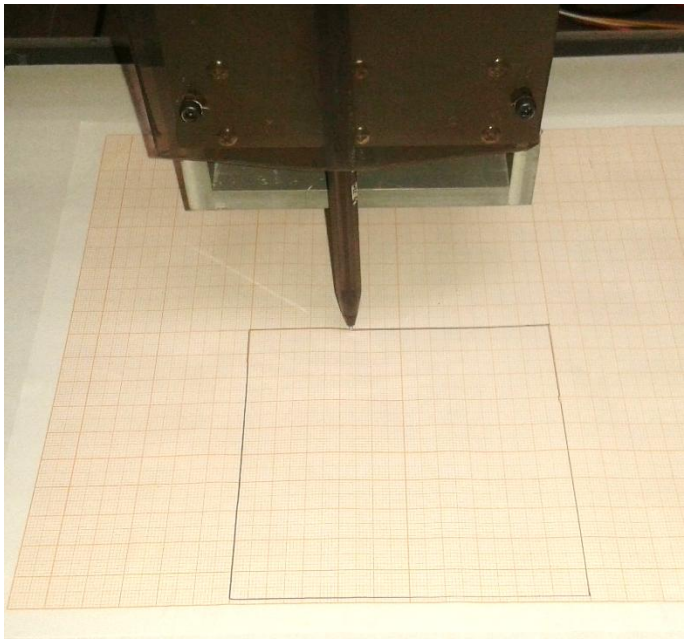


Figura 6.14 Operación del sistema automático

Se puede apreciar en la Figura 6.14 el dibujo de un diseño cuadrado instruida en el modo automático en una hoja milimétrica para posteriormente ser medida como se muestra en las siguientes figuras:

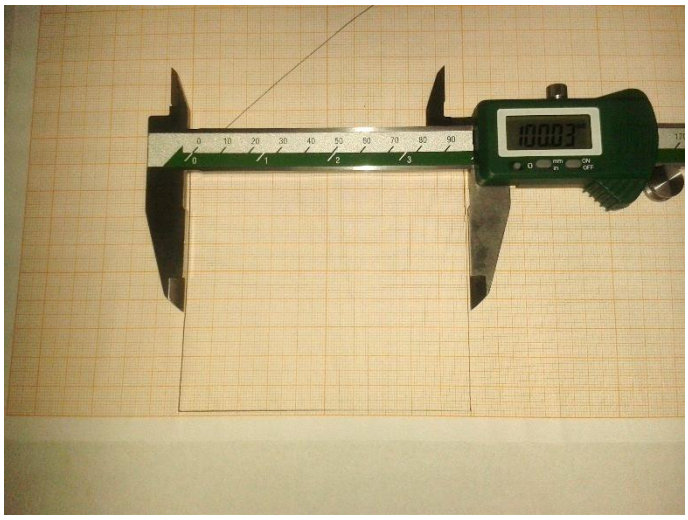


Figura 6.15 Medición del ancho de la figura cuadrada

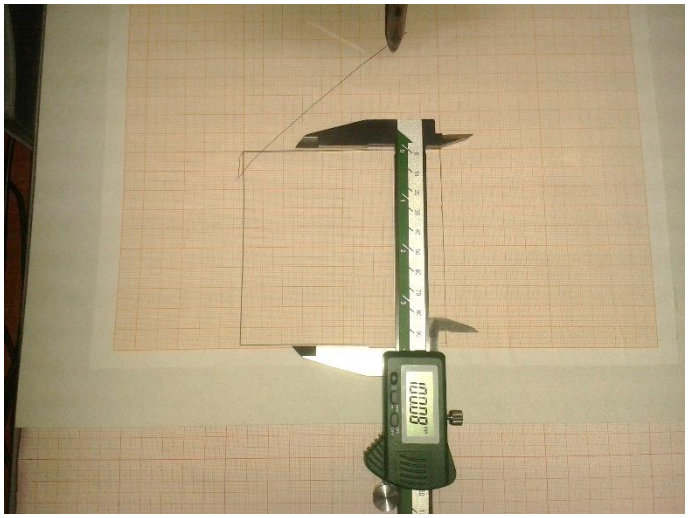


Figura 6.16 Medición del largo de la figura cuadrada

En las Figuras 6.15 y 6.16 se aprecia las mediciones obtenidas de la figura cuadrada fueron de:

- Largo:100.08 mm
- Ancho:100.03 mm

La distancia del lado largo fue por parte del desplazamiento del eje Z, mostrando un error de 8 micrómetros o una pérdida de 4 pasos. Del lado ancho de la figura correspondió el desplazamiento al eje “X” con un error de 3 micrómetros o una pérdida de 1 pasó del motor a pasos.

Apreciando los errores presentados en las pruebas, se considera que con respecto a una vuelta completa del motor que genera un desplazamiento de 8 milímetros y se requieren 400 pasos, se tomando la mayor pérdida presentada siendo de 8 micrómetros o 4 pasos perdidos y esto representa un 1% de pérdida de pasos con respecto a una vuelta.

CONCLUSIONES

El sistema permite el manejo de tres ejes (X, Y, Z) permite el desplazamiento de 8 milímetros en 1.044 segundos, presentándose un erro de 44 milésimas de segundo.

Los errores en cuanto a las posiciones (X, Y, Z) corresponden 5 micrómetros, 11 micrómetros y 8 micrómetros y se contemplan mediante software para realizar los ajustes necesarios en cada prueba.

El sistema permite operar mediante interfaces gráficas desarrolladas en Python 3.4 y en Raspberry Pi 2 para permitir la interacción con el usuario mediante botones, radio botones, cajas de texto y los datos obtenidos son procesados de acuerdo al modo de manejo en el que se encuentre el usuario. El modo de manejo manual permite al usuario operar por segmentos indicados en el sistema sobre algún eje hacia la dirección requerida en milímetros, pasos o revoluciones, también en esta interfaz permite manipular los ejes de manera libre mediante un control USB JoysStick y al dirigir las palancas analógicas hacia alguna dirección ya sea (Y+ , Y-, X+, X-, Z+ y Z-) el sistema detecta esta dirección y desplaza el eje con ese sentido durante el tiempo que se encuentre posicionada la palanca analógica. El modo de manejo automático permite al usuario indicar la velocidad en dos modalidades en lento y rápido (0.025 y 0.50 segundos), la distancia a desplazar para poder seleccionar el eje que se desplace con estas características y el sistema pueda ir generando la secuencia de acciones a realizar para cada uno de los ejes a indicados.

Para poder rotar cada motor a pasos se desarrolló un driver que permite activar las bobinas que conforman al motor en su interior de hasta 6Amperes.

El sistema se encuentra desarrollado para permitir ser una herramienta en poder posicionar y desplazar una antorcha de plasma con la finalidad de tratar materiales y modificar sus características físicas. Para esto se requiere contar con la mayor precisión posible en el desplazamiento. La antorcha de plasma se incluirá en futuros trabajos.

Para un posible trabajo a futuro se podría contemplar la modificación del driver electrónico para el motor a pasos desarrollado empleando un driver L297 como exitador o activador del Mosfet de potencia, esto permitirá reducir el uso de pines Gpio en la Raspberry Pi, la programación en la clase para mover el motor a pasos ya no sería

mediante una secuencia de pasos, sino mediante un pulso (PWM) y el uso de otro pin para indicar la dirección de giro al motor. También la implementación de sensores de distancia que permitan retroalimentar al sistema y pueda ajustar el desplazamiento a cualquier perdida de distancia, así como el monitoreo del driver en cuanto a su temperatura y evitar una posible falla de algún componente electrónico.

REFERENCIAS

Trabajos citados

[1] O. J. Leslie, E. Ú. Sequeira y Yamil, «Máquina de Control Numérico Computarizado (CNC) de cuatro ejes, con comunicación USB, para la automatización de los procesos de grabado en madera, plástico y materiales no convencionlaes,» de *Tesis de Licenciatura*, Managua, Universidad Nacional de Ingeneria, Facultad de Electrónica y Computación, 2011, p. 71.

[2] N. Londoño Ospina, P. León Simanca, J. Álvarez Díaz y E. Marín Zapata, «Descripción del diseño y construcción de un torno de control numérico,» *Ingenier´ıa y Ciencia, ISSN,* vol. 1, nº 2, pp. 41-51, 21 06 2005.

[3] electroensaimada, «electrosaimada,» 25 02 2012. [En línea]. Available: http://www.electroensaimada.com/cnc.html. [Último acceso: 30 05 2016].

[4] R. Pi, «Rasboerry Pi,» [En línea]. Available: https://www.raspberrypi.org/products/raspberry-pi-2-model-b/. [Último acceso: 14 06 2017].

[5] G. Van Rossum, «Interactively Testing Remote Servers Using the Python Programing Language,» HIO Enshede; P.O. Box, Enshede, 2014.

[6] F. Acuña, D. Rivas y C. Navarrete, *IEEE Latin America Transactions,* vol. 13, nº 6, pp. 1893-1898, 06 06 2015.

[7] Z. Toolworks, «CNC DIY Kits,» 2017. [En línea]. Available: http://www.zentoolworks.com/product_info.php?cPath=14&products_id=133. [Último acceso: 25 03 25].

[8] A. Codina García, «Posiconamiento y proyección actual del motor de paso en aplicaciones industriales.,» *Revista de Ingenería mecánica* , vol. 5, nº 2, pp. 1-10, 2002.

[9] S. Kenshi, «Shinano,» [En línea]. Available: http://www.dynetics.eu/CMS/Docs/Shinano%20Kenshi/SST59D.pdf. [Último acceso: 18 06 17].

[10] M. Roseman, «tkdocs/antecedentes,» 2017. [En línea]. Available: http://www.tkdocs.com/resources/backgrounder.html. [Último acceso: 2017 06 17].

[11] L. E. Úbeda Sequiera y Y. O. Jiménez López, «Google Académico,» 03 09 2011. [En línea]. Available: http://udpeo.uni.edu.ni/doc/MonografiaCNC_FINAL.pdf. [Último acceso: 28 03 2017].

[12] F. Acuña, D. Rivas y C. Navarrete, «IEEE,» 06 06 2015. [En línea]. Available: http://www.ewh.ieee.org/reg/9/etrans/ieee/issues/vol13/vol13issue06June2015/13TLA6_05Acuna.pdf. [Último acceso: 16 06 2016].

[13] A. Sasongko, A. Tulus Purnomo y F. Ishad Hariadi, «Ieee Xplorer,» *IEEE*, p. 6, 9 12 2015.

[14] S. Pandian y S. R. Pandian, «http://www.irdindia.in/journal_ijmer/pdf/vol2_iss1/2.pdf,» 2014. [En línea]. Available: http://www.irdindia.in/journal_ijmer/pdf/vol2_iss1/2.pdf. [Último acceso: 08 04 2017].

[15] O. J. Leslie, E. Ú. Sequeira y Yamil, «Google Scholar,» 03 09 2011. [En línea]. Available: udpeo.uni.edu.ni/doc/MonografiaCNC_FINAL.pdf. [Último acceso: 2016 05 30].

[16] N. Londoño Ospina, P. León Simanca, J. Álvarez Díaz y E. Marín Zapata, «Google Scholar,» 21 06 2005. [En línea]. Available: publicaciones.eafit.edu.co/index.php/ingciencia/article/view/494/470. [Último acceso: 30 05 2016].

[17] M. Eric, M. Cesar y V. Carlos, «Google Scholar,» 26 05 2014. [En línea]. Available: http://www.dspace.espol.edu.ec/xmlui/handle/123456789/25322. [Último acceso: 15 06 2016].

[18] G. Mondragón García , «Sistema de posicionamiento y barrido asistido por computadora para la aplicación de descargas superficiales,» Atlacomulco, 2013.

[19] P. Smid, CNC Programming Handbook, New York: Industrial Press Inc., 2003.

[20] A. Fernández Montoro, Python 3 al descubierto, España: RC Libros, 2012, p. 24.

[21] V. Milántiev y S. Temkó, «Introducción,» de *Física del Plasma*, Moscú, Mir, 1987, p. 6.

[22] M. García Tsai y J. Montero , «Plasma: una tecnología de gran potencial para la industria y la ciencia,» *INGENIUS*, vol. 1, nº 4, p. 7, 2010.

[23] D. Alciatore y M. Histand, «Mecatrónica y los sitemas de medición,» de *Motor a pasos*, Mc Graw Hill, 2008, pp. 413-419.

[24] R. Pi, «Raspberry Pi,» [En línea]. Available: https://www.raspberrypi.org/products/raspberry-pi-2-model-b/. [Último acceso: 2017 06 18].

[25] Toshiba, *Toshiba Photocoupler TLP250,* 2004, pp. 1-4.

[26] I. I. Rectifier, *Datasheet IRF1407PbF,* 2014, pp. 1-2.

[27] P.-c. A. Q. f. E. Needs, «Departamente de Ingenería Electríca y de Control,» Erasmus, 2011.

ANEXOS

Anexo A:

Hoja de datos del optocoplador TLP250.

TOSHIBA TLP250

TOSHIBA Photocoupler GaAlAs Ired & Photo-IC

TLP250

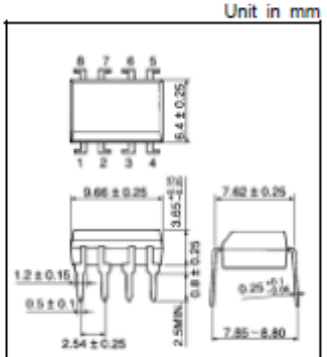
Transistor Inverter
Inverter For Air Conditionor
IGBT Gate Drive
Power MOS FET Gate Drive

The TOSHIBA TLP250 consists of a GaAlAs light emitting diode and a integrated photodetector.
This unit is 8-lead DIP package.
TLP250 is suitable for gate driving circuit of IGBT or power MOS FET.

- Input threshold current: I_F=5mA(max.)
- Supply current (I_{CC}): 11mA(max.)
- Supply voltage (V_{CC}): 10-35V
- Output current (I_O): ±1.5A (max.)
- Switching time (t_{pLH}/t_{pHL}): 1.5μs(max.)
- Isolation voltage: 2500V_{rms}(min.)
- UL recognized: UL1577, file No.E67349
- Option (D4) type
  VDE approved: DIN VDE0884/06.92,certificate No.76823
  Maximum operating insulation voltage: 630V_{PK}
  Highest permissible over voltage: 4000V_{PK}

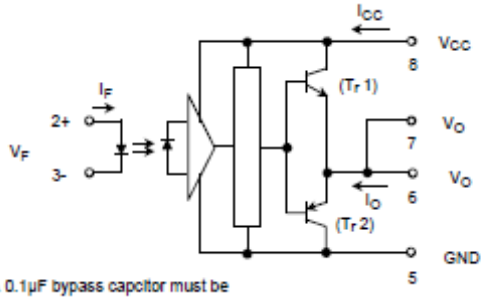
  (Note) When a VDE0884 approved type is needed, please designate the "option (D4)"
- Creepage distance: 6.4mm(min.)
  Clearance: 6.4mm(min.)

Unit in mm



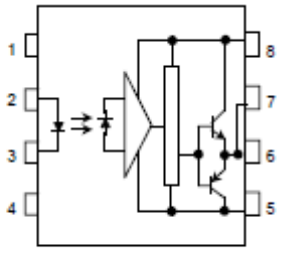
TOSHIBA 11-10C4
Weight: 0.54 g

Schmatic



A 0.1μF bypass capcitor must be connected between pin 8 and 5 (See Note 5).

Pin Configuration (top view)



1 : N.C.
2 : Anode
3 : Cathode
4 : N.C.
5 : GND
6 : V_O (Output)
7 : V_O
8 : V_{CC}

Truth Table

Absolute Maximum Ratings (Ta = 25°C)

	Characteristic		Symbol	Rating	Unit
LED	Forward current		I_F	20	mA
	Forward current derating (Ta ≥ 70°C)		$\Delta I_F / \Delta Ta$	−0.36	mA / °C
	Peak transient forward curent	(Note 1)	I_{FPT}	1	A
	Reverse voltage		V_R	5	V
	Junction temperature		Tj	125	°C
Detector	"H"peak output current (P_W ≤ 2.5μs,f ≤ 15kHz)	(Note 2)	I_{OPH}	−1.5	A
	"L"peak output current (P_W ≤ 2.5μs,f ≤ 15kHz)	(Note 2)	I_{OPL}	+1.5	A
	Output voltage	(Ta ≤ 70°C)	V_O	35	V
		(Ta = 85°C)		24	
	Supply voltage	(Ta ≤ 70°C)	V_{CC}	35	V
		(Ta = 85°C)		24	
	Output voltage derating (Ta ≥ 70°C)		$\Delta V_O / \Delta Ta$	−0.73	V / °C
	Supply voltage derating (Ta ≥ 70°C)		$\Delta V_{CC} / \Delta Ta$	−0.73	V / °C
	Junction temperature		Tj	125	°C
Operating frequency		(Note 3)	f	25	kHz
Operating temperature range			T_{opr}	−20~85	°C
Storage temperature range			T_{stg}	−55~125	°C
Lead soldering temperature (10 s)		(Note 4)	T_{sol}	260	°C
Isolation voltage (AC, 1 min., R.H.≤ 60%)		(Note 5)	BV_S	2500	Vrms

Note 1: Pulse width P_W ≤ 1μs, 300pps

Note 2: Exporenential wavefom

Note 3: Exporenential wavefom, I_{OPH} ≤ −1.0A(≤ 2.5μs), I_{OPL} ≤ +1.0A(≤ 2.5μs)

Note 4: It is 2 mm or more from a lead root.

Note 5: Device considerd a two terminal device: Pins 1, 2, 3 and 4 shorted together, and pins 5, 6, 7 and 8 shorted together.

Note 6: A ceramic capacitor(0.1μF) should be connected from pin 8 to pin 5 to stabilize the operation of the high gain linear amplifier. Failure to provide the bypassing may impair the switching proparty. The total lead length between capacitor and coupler should not exceed 1cm.

Recommended Operating Conditions

Characteristic		Symbol	Min.	Typ.	Max.		Unit
Input current, on	(Note 7)	$I_{F(ON)}$	7	8	10		mA
Input voltage, off		$V_{F(OFF)}$	0	—	0.8		V
Supply voltage		V_{CC}	15	—	30	20	V
Peak output current		I_{OPH}/I_{OPL}	—	—	±0.5		A
Operating temperature		T_{opr}	−20	25	70	85	°C

Electrical Characteristics (Ta = –20~70°C, unless otherwise specified)

Characteristic		Symbol	Test Cir-cuit	Test Condition		Min.	Typ.*	Max.	Unit
Input forward voltage		V_F	—	I_F = 10 mA , Ta = 25°C			1.6	1.8	V
Temperature coefficient of forward voltage		$\Delta V_F / \Delta Ta$	—	I_F = 10 mA		—	–2.0	—	mV / °C
Input reverse current		I_R	—	V_R = 5V, Ta = 25°C			—	10	μA
Input capacitance		C_T	—	V = 0 , f = 1MHz , Ta = 25°C		—	45	250	pF
Output current	"H" level	I_{OPH}	3	V_{CC} = 30V (*1)	I_F = 10 mA V_{8-6} = 4V	–0.5	–1.5	—	A
	"L" level	I_{OPL}	2		I_F = 0 V_{5-5} = 2.5V	0.5	2	—	
Output voltage	"H" level	V_{OH}	4	V_{CC1} = +15V, V_{EE1} = –15V R_L = 200Ω, I_F = 5mA		11	12.8	—	V
	"L" level	V_{OL}	5	V_{CC1} = +15V, V_{EE1} = –15V R_L = 200Ω, V_F = 0.8V		—	–14.2	–12.5	
Supply current	"H" level	I_{CCH}	—	V_{CC} = 30V, I_F = 10mA Ta = 25°C		—	7	—	mA
				V_{CC} = 30V, I_F = 10mA		—	—	11	
	"L" level	I_{CCL}	—	V_{CC} = 30V, I_F = 0mA Ta = 25°C		—	7.5	—	
				V_{CC} = 30V, I_F = 0mA		—	—	11	
Threshold input current	"Output L→H"	I_{FLH}	—	V_{CC1} = +15V, V_{EE1} = –15V R_L = 200Ω, V_O > 0V		—	1.2	5	mA
Threshold input voltage	"Output H→L"	I_{FHL}	—	V_{CC1} = +15V, V_{EE1} = –15V R_L = 200Ω, V_O < 0V		0.8	—	—	V
Supply voltage		V_{CC}	—			10	—	35	V
Capacitance (input–output)		C_S	—	V_S = 0 , f = 1MHz Ta = 25°		—	1.0	2.0	pF
Resistance(input–output)		R_S	—	V_S = 500V , Ta = 25°C R.H.≤ 60%		1×10^{12}	10^{14}	—	Ω

* All typical values are at Ta = 25°C (*1): Duration of I_O time ≤ 50μs

Anexo B:

Hoja de datos de Mosfet IRF1407.

International
IOR Rectifier

PD - 93907

AUTOMOTIVE MOSFET

IRF1407

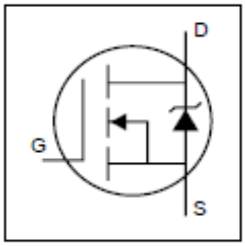
Typical Applications

- Integrated Starter Alternator
- 42 Volts Automotive Electrical Systems

Benefits

- Advanced Process Technology
- Ultra Low On-Resistance
- Dynamic dv/dt Rating
- 175°C Operating Temperature
- Fast Switching
- Repetitive Avalanche Allowed up to Tjmax

HEXFET® Power MOSFET



V_{DSS} = 75V

$R_{DS(on)}$ = 0.0078Ω

I_D = 130A⑥

Description

Specifically designed for Automotive applications, this Stripe Planar design of HEXFET® Power MOSFETs utilizes the lastest processing techniques to achieve extremely low on-resistance per silicon area. Additional features of this HEXFET power MOSFET are a 175°C junction operating temperature, fast switching speed and improved repetitive avalanche rating. These benefits combine to make this design an extremely efficient and reliable device for use in Automotive applications and a wide variety of other applications.

TO-220AB

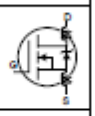
Absolute Maximum Ratings

	Parameter	Max.	Units
I_D @ T_C = 25°C	Continuous Drain Current, V_{GS} @ 10V	130⑥	A
I_D @ T_C = 100°C	Continuous Drain Current, V_{GS} @ 10V	92⑥	
I_{DM}	Pulsed Drain Current ①	520	
P_D @T_C = 25°C	Power Dissipation	330	W
	Linear Derating Factor	2.2	W/°C
V_{GS}	Gate-to-Source Voltage	± 20	V
E_{AS}	Single Pulse Avalanche Energy②	390	mJ
I_{AR}	Avalanche Current①	See Fig.12a, 12b, 15, 16	A
E_{AR}	Repetitive Avalanche Energy⑦		mJ
dv/dt	Peak Diode Recovery dv/dt ③	4.6	V/ns
T_J T_{STG}	Operating Junction and Storage Temperature Range	-55 to + 175	°C
	Soldering Temperature, for 10 seconds	300 (1.6mm from case)	
	Mounting Torque, 6-32 or M3 screw	10 lbf•in (1.1N•m)	

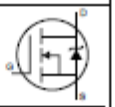
Thermal Resistance

	Parameter	Typ.	Max.	Units
$R_{\theta JC}$	Junction-to-Case	—	0.45	°C/W
$R_{\theta CS}$	Case-to-Sink, Flat, Greased Surface	0.50	—	
$R_{\theta JA}$	Junction-to-Ambient	—	62	

Electrical Characteristics @ T_J = 25°C (unless otherwise specified)

	Parameter	Min.	Typ.	Max.	Units	Conditions
$V_{(BR)DSS}$	Drain-to-Source Breakdown Voltage	75	—	—	V	V_{GS} = 0V, I_D = 250µA
$\Delta V_{(BR)DSS}/\Delta T_J$	Breakdown Voltage Temp. Coefficient	—	0.09	—	V/°C	Reference to 25°C, I_D = 1mA
$R_{DS(on)}$	Static Drain-to-Source On-Resistance	—	—	0.0078	Ω	V_{GS} = 10V, I_D = 78A ④
$V_{GS(th)}$	Gate Threshold Voltage	2.0	—	4.0	V	V_{DS} = 10V, I_D = 250µA
g_{fs}	Forward Transconductance	74	—	—	S	V_{DS} = 25V, I_D = 78A
I_{DSS}	Drain-to-Source Leakage Current	—	—	20	µA	V_{DS} = 75V, V_{GS} = 0V
		—	—	250		V_{DS} = 60V, V_{GS} = 0V, T_J = 150°C
I_{GSS}	Gate-to-Source Forward Leakage	—	—	200	nA	V_{GS} = 20V
	Gate-to-Source Reverse Leakage	—	—	-200		V_{GS} = -20V
Q_g	Total Gate Charge	—	160	250	nC	I_D = 78A
Q_{gs}	Gate-to-Source Charge	—	35	52		V_{DS} = 60V
Q_{gd}	Gate-to-Drain ("Miller") Charge	—	54	81		V_{GS} = 10V④
$t_{d(on)}$	Turn-On Delay Time	—	11	—	ns	V_{DD} = 38V
t_r	Rise Time	—	150	—		I_D = 78A
$t_{d(off)}$	Turn-Off Delay Time	—	150	—		R_G = 2.5Ω
t_f	Fall Time	—	140	—		V_{GS} = 10V ④
L_D	Internal Drain Inductance	—	4.5	—	nH	Between lead, 6mm (0.25in.) from package and center of die contact 
L_S	Internal Source Inductance	—	7.5	—		
C_{iss}	Input Capacitance	—	5600	—	pF	V_{GS} = 0V
C_{oss}	Output Capacitance	—	890	—		V_{DS} = 25V
C_{rss}	Reverse Transfer Capacitance	—	190	—		f = 1.0KHz, See Fig. 5
C_{oss}	Output Capacitance	—	5800	—		V_{GS} = 0V, V_{DS} = 1.0V, f = 1.0KHz
C_{oss}	Output Capacitance	—	560	—		V_{GS} = 0V, V_{DS} = 60V, f = 1.0KHz
C_{oss} eff.	Effective Output Capacitance ⑤	—	1100	—		V_{GS} = 0V, V_{DS} = 0V to 60V

Source-Drain Ratings and Characteristics

	Parameter	Min.	Typ.	Max.	Units	Conditions
I_S	Continuous Source Current (Body Diode)	—	—	130⑥	A	MOSFET symbol showing the integral reverse p-n junction diode. 
I_{SM}	Pulsed Source Current (Body Diode) ①	—	—	520		
V_{SD}	Diode Forward Voltage	—	—	1.3	V	T_J = 25°C, I_S = 78A, V_{GS} = 0V ④
t_{rr}	Reverse Recovery Time	—	110	170	ns	T_J = 25°C, I_F = 78A
Q_{rr}	Reverse RecoveryCharge	—	390	590	nC	di/dt = 100A/µs ④
t_{on}	Forward Turn-On Time	Intrinsic turn-on time is negligible (turn-on is dominated by L_S+L_D)				

Notes:

① Repetitive rating; pulse width limited by max. junction temperature. (See fig. 11).

② Starting T_J = 25°C, L = 0.13mH R_G = 25Ω, I_{AS} = 78A. (See Figure 12).

③ I_{SD} ≤ 78A, di/dt ≤ 320A/µs, V_{DD} ≤ $V_{(BR)DSS}$, T_J ≤ 175°C

④ Pulse width ≤ 400µs; duty cycle ≤ 2%.

⑤ C_{oss} eff. is a fixed capacitance that gives the same charging time as C_{oss} while V_{DS} is rising from 0 to 80% V_{DSS} .

⑥ Calculated continuous current based on maximum allowable junction temperature. Package limitation current is 75A.

⑦ Limited by T_{Jmax} , see Fig.12a, 12b, 15, 16 for typical repetitive avalanche performance.