PROYECTO FIN DE GRADO

TÍTULO:
Diseño y desarrollo de una máquina de control numérico controlada mediante el uso de un microcontrolador para la fabricación de placas de circuito impreso (PCBs).

AUTOR/A: Sergio Sanz López

TITULACIÓN: Grado en Electrónica de Comunicaciones

TUTOR/A: Neftalí Núñez (EFF)

DEPARTAMENTO: Electrónica Física, Ingeniería Eléctrica y física Aplicada

VºBº TUTOR/A

Miembros del Tribunal Calificador:

PRESIDENTE/A: Antonio Carpeño (DTE)

TUTOR/A: Neftalí Núñez (EFF)

SECRETARIO/A: Manuel Vázquez (EFF)

Fecha de lectura:

Calificación:

El Secretario/La Secretaria,

A mis padres, por confiar en mi en todo momento, cuidarme y quererme por como soy, os quiero 10.

A mis hermanos Daniel y Ricardo, por ser un apoyo incondicional en cada etapa de mi vida.

A mi pareja Adriana, por aparecer en mi vida en el mejor momento llenándola de recuerdos felices, te quiero.

A todos ellos, darles las gracias por aguantarme durante esta etapa, un camino largo y lleno de retos difíciles, pero no imposibles.

"Aquellos que busquen la gloria eterna, que sepan que ninguna tarea es imposible, que si somos recordados en los corazones y las mentes de aquellos que nos quedan atrás, jamás seremos olvidados"

Resumen

El objetivo del presente proyecto ha sido el diseño, desarrollo y validación de una Máquina de Control Numérico por Computadora (CNC) que sea accesible y versátil, la cual estará destinada a la automatización de tareas de mecanizado durante el proceso de fabricación de prototipos de PCBs tanto en entornos académicos como industriales. Este proceso se lleva a cabo considerando múltiples condicionantes que abarcan aspectos tecnológicos, económicos y ambientales.

Desde una perspectiva tecnológica, el proyecto se enfoca en el meticuloso diseño y ensamblaje de la CNC, donde se seleccionan componentes de alta calidad para garantizar una gran robustez y precisión en su funcionamiento. El control de la máquina se implementa mediante una placa Arduino y se establece una interfaz de usuario en Visual Studio que facilita la interacción con el equipo. Esto implica la utilización de algoritmos de control específicos y la comunicación efectiva entre la placa Arduino y la interfaz de usuario.

En el contexto económico, se busca crear una solución asequible para la automatización de tareas de mecanizado. Se realiza un análisis de costos detallado de los componentes y materiales utilizados en el diseño y construcción de la CNC, además del costo de la mano de obra durante el ensamblaje de los componentes de la máquina, garantizando de esta manera que la solución sea viable desde un punto de vista financiero.

El aspecto ambiental se tiene en cuenta mediante la selección de componentes y materiales que minimicen el impacto ambiental y cumplan con las normativas correspondientes. Se promueve la eficiencia energética y se reducen los residuos en la fabricación (el ataque químico y el cobre eliminado mediante ese ataque) y operación de la máquina CNC.

En cuanto al impacto del proyecto, tiene en cuenta las implicaciones sociales, implicaciones de salud y seguridad e implicaciones ambientales, además de realizar aportaciones a los Objetivos de Desarrollo Sostenible.

La metodología empleada en el proyecto incluye el diseño, la fabricación, la programación tanto de la placa como de la interfaz de usuario, integración del conjunto y las pruebas de funcionamiento de la CNC. Se llevan a cabo pruebas con diversos circuitos de distintas complejidades para verificar la precisión y confiabilidad de la máquina, y se realizan demostraciones prácticas que permiten comparar los resultados obtenidos con las expectativas teóricas.

En cuanto a los resultados y conclusiones del proyecto se puede garantizar que el producto final cumple con las restricciones de diseño estipulados y es capaz de automatizar los procesos de mecanizado, siendo controlado desde la interfaz de usuario. Como añadido, durante la fase de diseño se optó por el empleo de un eje Z con cabezal intercambiable, esto quiere decir que la máquina es capaz de dibujar (Plotter) si se le equipa un rotulador, de taladrar si se le equipa una broca, de eliminar el cobre si se le

equipa con un cabezal giratorio y una fresa, o incluso sería capaz de extruir plástico si se equipase con un extrusor. Esto hace que la máquina no sólo sea capaz de cumplir las necesidades de este proyecto, sino que, además, con mínimas modificaciones tanto en el hardware como en el software, este modelo de CNC tendría la capacidad de ser funcional en diversas tareas más allá del mecanizado.

En resumen, este proyecto constituye una contribución significativa al ámbito de la automatización y la fabricación de prototipos de PCBs, al proporcionar una solución de bajo costo, pero altamente versátil y precisa para tareas de mecanizado, respaldada por una interfaz de usuario amigable y eficiente.

Abstract

The objective of this project has been the design, development and validation of an accessible and versatile Computer Numerical Control (CNC) machine, which will be used for the automation of machining tasks during the manufacturing process of PCB prototypes in both academic and industrial environments. This process is carried out considering multiple conditioning factors that include technological, economic and environmental aspects.

From a technological perspective, the project focuses on the meticulous design and assembly of the CNC, where high quality components are selected to ensure high robustness and precision in its operation. The machine control is implemented through an Arduino board and a Visual Studio user interface is established to facilitate the interaction with the equipment. This involves the use of specific control algorithms and effective communication between the Arduino board and the user interface.

In the economic context, we seek to create an affordable solution for the automation of machining tasks. A detailed cost analysis of the components and materials used in the design and construction of the CNC, in addition to the cost of labor during the assembly of the machine components, is performed, thus ensuring that the solution is viable from a financial point of view.

The environmental aspect is taken into account by selecting components and materials that minimize environmental impact and comply with the corresponding regulations. Energy efficiency is promoted and waste is reduced in the manufacture (chemical etching and copper removed by etching) and operation of the CNC machine.

Regarding the impact of the project, it takes into account social implications, health and safety implications and environmental implications, in addition to making contributions to the Sustainable Development Goals.

The methodology employed in the project includes design, manufacturing, programming of both the board and the user interface, integration of the assembly and testing of the CNC. Tests are carried out with various circuits of different complexities to verify the accuracy and reliability of the machine, and practical demonstrations are performed to compare the results obtained with the theoretical expectations.

As for the results and conclusions of the project, it can be guaranteed that the final product complies with the stipulated design restrictions and is capable of automating the machining processes, being controlled from the user interface. In addition, during the design phase we opted for the use of a Z axis with interchangeable head, this means that the machine is capable of drawing (Plotter) if equipped with a marker, drilling if equipped with a drill, removing copper if equipped with a rotating head and a milling cutter, or even be able to extrude plastic if equipped with an extruder. This makes the machine not only capable of meeting the needs of this project, but also, with minimal

modifications to both hardware and software, this CNC model would have the ability to be functional in a variety of tasks beyond machining.

In summary, this project is a significant contribution to the field of PCB automation and prototyping by providing a low-cost, yet highly versatile and accurate solution for machining tasks, supported by a friendly and efficient user interface.

Índice de contenido

Índice de ilustraciones

Lista de acrónimos

ACK	Acknowledgment
CNC	Computer Numerical Control
FDC	Final De Carrera
GRBL	G-Code Reference Block Library
GUI	Graphic User Interface
IA	Inteligencia Artificial
IDE	Integrated Development Environment
NEMA	National Electrical Manufacturers Association
ODS	Objetivos de Desarrollo Sostenible
PFG	Proyecto de Fin de Grado
PCB	Printed Circuit Board
PLA	Polylactic Acid
ACK	Acknowledgment
PWM	Pulse With Modulation
PYME	Pequeña Y Mediana Empresa

Índice de tablas

1. Introducción, antecedentes y objetivos

1.1. Introducción

En la actualidad, el control numérico por computadora ha revolucionado la industria de la fabricación al ofrecer una precisión y versatilidad inigualables en la producción de piezas y componentes. Esta tecnología ha permitido la automatización de procesos, reduciendo errores y tiempos de producción, además de brindar nuevas oportunidades para la innovación en el diseño y la manufactura.

Adicionalmente, se debe destacar que en los últimos años, la tecnología se ha encaminado por el mundo de la impresión 3D, en sus diversas formas. Esta tendencia ha ampliado aún más las posibilidades de fabricación, permitiendo con ello la creación de nuevos prototipos.

El propósito fundamental de este PFG es explorar las posibilidades y ventajas de utilizar una máquina de control numérico manejada por Arduino, un microcontrolador de código abierto ampliamente accesible y adaptable. Ha de tenerse en cuenta las restricciones de funcionamiento referentes a Arduino, así como las posibles restricciones de diseño del proyecto, tales como tamaño de pistas, tamaños de broca, superficie de trabajo, diseño hardware etc... .La combinación de la tecnología de control numérico por computadora con Arduino no solo brinda una solución asequible, sino que también permite una mayor flexibilidad en la implementación de proyectos de control numérico.

Este trabajo no se centra únicamente en el diseño, fabricación y configuración de una máquina CNC, sino que también aborda el diseño y posterior desarrollo de una interfaz de control en Visual Studio, lo que añade un componente de software esencial a la ecuación.

La investigación y desarrollo en este proyecto incluyen la selección y ensamblaje de componentes, la programación de Arduino para el control de motores y herramientas, la creación de diseños 3D para producir piezas específicas, y la implementación de una interfaz de usuario amigable. A lo largo de esta memoria, se detallarán los pasos cruciales y los resultados obtenidos en cada una de estas áreas.

Esta máquina estará compuesta principalmente por una estructura de aluminio, que cuenta con motores y finales de carrera además de una base de apoyo de las placas, todo ello conectado a un microprocesador Arduino el cual cuenta con una placa de expansión CNC Shield v3 y una conexión mediante puerto serie a la interfaz de control.

El componente de Arduino, junto con la placa de expansión CNC Shield v3 son elementos de código abierto, es decir, su código fuente está disponible para su acceso y

modificación por parte del público en general, sin requerir el pago de ninguna tarifa. Gracias a esto, ha permitido que el desarrollo de código fuese más fluido debido a la facilidad de encontrar la resolución de posibles errores en el código, las cuales están presentes en internet por parte de la comunidad de usuarios.

En las secciones que siguen, se detallarán los antecedentes de la tecnología CNC, los fundamentos de Arduino y su aplicación en el control numérico, así como la metodología y los resultados obtenidos a lo largo de este proyecto. Este trabajo representa un esfuerzo dedicado a la investigación y desarrollo en un campo con un potencial significativo y abre la puerta a futuras exploraciones y aplicaciones de la tecnología CNC controlada por Arduino.

1.2. Antecedentes

Los antecedentes juegan un papel de suma importancia en cualquier proyecto de investigación o desarrollo, proporcionando un contexto relevante que demuestra la necesidad y la justificación del proyecto en cuestión. En el caso de este trabajo centrado en el diseño y desarrollo de una Máquina de Control Numérico por Computadora controlada mediante un microcontrolador Arduino y con una interfaz implementada en Visual Studio, los antecedentes se extienden a lo largo de diversas áreas de importancia.

El campo de la automatización y la robótica ha experimentado un crecimiento significativo en los últimos años. Las CNC se han convertido en piezas fundamentales de esta revolución tecnológica, permitiendo la fabricación precisa y eficiente de piezas y componentes en una amplia gama de industrias, desde la manufactura hasta la medicina. El impacto de la automatización en la industria no solo se traduce en una mayor productividad, sino también en una mayor precisión y reducción de costos.

Arduino, como plataforma de desarrollo de hardware de código abierto, ha desempeñado un papel crucial en la democratización de la electrónica y la automatización. Ha permitido a ingenieros, estudiantes y entusiastas de la tecnología crear soluciones personalizadas para una variedad de aplicaciones. Su versatilidad y facilidad de uso lo convierten en una elección popular para el control de sistemas, desde pequeños proyectos hasta aplicaciones industriales.

En paralelo al desarrollo de hardware, la creación de interfaces de usuario intuitivas y efectivas se ha vuelto esencial para permitir a los usuarios interactuar con sistemas automatizados de manera eficiente. Las aplicaciones de software, como las desarrolladas en Visual Studio, han facilitado la creación de interfaces de usuario personalizadas y amigables.

El desarrollo de una CNC controlada por Arduino con una interfaz en Visual Studio no solo satisface una necesidad tecnológica creciente, sino que también promueve la

accesibilidad y la versatilidad en la automatización de tareas de mecanizado y fabricación. Los antecedentes resaltan la relevancia de este proyecto en el contexto actual y anticipan contribuciones significativas en el campo de la automatización industrial.

Además, en un contexto donde la eficiencia y la optimización de los procesos industriales son cruciales para la competitividad, la integración de esta tecnología ofrece una solución que no solo mejora la precisión y la calidad del producto final, sino que también agiliza los tiempos de producción y reduce los costes asociados. Esta combinación de hardware y software proporciona una plataforma flexible y adaptable que puede ser implementada en una variedad de entornos, desde usuarios en su hogar hasta pequeñas y medianas empresas (PYMES).

1.3. Objetivos

El objetivo general de este proyecto es diseñar, desarrollar y validar una Máquina de Control Numérico por Computadora que sea accesible y versátil, destinada a automatizar tareas de mecanizado y fabricación en entornos académicos e industriales. Este objetivo se deriva de la necesidad de contar con una herramienta eficiente y asequible que brinde una solución precisa para esta tarea. Los objetivos específicos son los siguientes:

1. **Diseño de la CNC**. Desarrollar el diseño de la máquina teniendo en cuenta los requisitos y las especificaciones de diseño, seleccionando componentes de alta calidad.

2. **Montaje de la CNC**. Llevar a cabo el ensamblado de los componentes y materiales seleccionados durante la fase de diseño.

3. **Implementación de Control con Arduino**. Programar algoritmos de control en una placa Arduino para gestionar el movimiento de los motores y la comunicación eficaz con la interfaz de usuario en Visual Studio mediante comunicación puerto serie.

4. **Interfaz de usuario**. Crear una interfaz de usuario intuitiva en Visual Studio que permita a los usuarios cargar diseños, previsualizar resultados y controlar la máquina de manera sencilla.

5. **Pruebas y validación**. Llevar a cabo pruebas exhaustivas para verificar la precisión y la fiabilidad tanto de la máquina CNC como de la interfaz desarrollada, demostrando su capacidad para realizar tareas de mecanizado con alta precisión y eficiencia.

6. **Análisis de Resultados y Conclusiones**. Evaluar los resultados obtenidos en relación con los objetivos planteados y elaborar conclusiones que destaquen las

contribuciones del proyecto y ofrezcan recomendaciones para futuras mejoras y desarrollos relacionados.

2. Marco tecnológico

El ámbito tecnológico de este proyecto abarca diversas áreas relacionadas con la automatización, la electrónica y el desarrollo de software.

2.1. Máquina de control numérico (CNC)

Una máquina de control numérico es un dispositivo electromecánico que se utiliza para la fabricación automatizada de piezas a partir de datos digitales, utilizando instrucciones precisas codificadas en un lenguaje de programación específico. Este tipo de máquinas son ampliamente utilizadas en la industria manufacturera para producir componentes con alta precisión y repetibilidad.

La estructura de una máquina de control numérico comprende varios componentes clave, cada uno desempeñando un papel fundamental en el proceso de fabricación. A continuación, se detalla una descripción detallada de estos componentes:

- **Controlador Numérico:** El corazón de la máquina de control numérico, este dispositivo recibe instrucciones de un programa de computadora y las traduce en movimientos precisos de los ejes de la máquina. Utiliza algoritmos complejos para coordinar el movimiento de los motores y garantizar la exactitud de las operaciones de mecanizado.

- **Ejes de la Máquina:** La mayoría de las CNC operan con tres o más ejes de movimiento que definen la posición tridimensional de la herramienta de corte o la pieza de trabajo. Estos ejes pueden incluir el movimiento lineal a lo largo de los ejes X, Y y Z, así como rotaciones alrededor de estos ejes.

- **Motores:** Los motores eléctricos controlan el movimiento de los ejes de la máquina en respuesta a las instrucciones del controlador numérico. Estos motores pueden ser de diferentes tipos, como motores paso a paso o servomotores, cada uno con sus propias características de precisión y velocidad.

- **Herramientas de Corte:** Las herramientas de corte, como brocas, fresas o cuchillas, se montan en la máquina y se utilizan para dar forma a la pieza de trabajo según las especificaciones del diseño. La selección de la herramienta adecuada depende del material de trabajo, la geometría requerida y otras consideraciones de mecanizado.

- **Sistema de Sujeción:** Para mantener la pieza de trabajo en su lugar durante el proceso de mecanizado, se utilizan dispositivos de sujeción que pueden incluir

mordazas, platos de sujeción o sistemas de sujeción especializados. Estos sistemas deben ser lo suficientemente robustos como para soportar las fuerzas de corte generadas durante la operación.

- **Sistema de Refrigeración y Lubricación:** Durante el mecanizado, es crucial mantener las herramientas y la pieza de trabajo a una temperatura adecuada y lubricada para evitar el sobrecalentamiento y el desgaste prematuro. Los sistemas de refrigeración y lubricación se encargan de este aspecto, proporcionando refrigerantes y lubricantes según sea necesario.

- **Panel de Control:** Los operadores de la máquina interactúan con el sistema a través de un panel de control que permite la entrada de datos, la monitorización del proceso y el ajuste de parámetros como la velocidad de corte, la profundidad de corte y otros ajustes específicos del trabajo en curso.

- **Computadora y Software de Control:** En muchas máquinas de control numérico modernas, una computadora integrada ejecuta el software de control que gestiona todas las funciones del sistema, desde la interpretación de los programas de mecanizado hasta la comunicación con los dispositivos periféricos y la supervisión del rendimiento de la máquina.

En conjunto, estos componentes forman una máquina de control numérico altamente sofisticada y versátil, capaz de producir una amplia variedad de piezas con precisión y eficiencia, lo que la convierte en un elemento esencial en la industria manufacturera moderna.

2.2. Arduino

Arduino es una plataforma de hardware y software de código abierto que ha ganado una enorme popularidad en el ámbito de la electrónica, la robótica y la creación de proyectos interactivos. Esta plataforma se caracteriza por su accesibilidad, versatilidad y facilidad de uso, lo que la convierte en una opción ideal tanto para principiantes como para expertos en electrónica. Durante el transcurso de este proyecto se ha escogido emplear la plataforma Arduino debido a su alto uso por parte de la comunidad en aplicaciones CNC.

La plataforma Arduino consiste en varios elementos clave que trabajan juntos para permitir la creación de una amplia gama de dispositivos y sistemas electrónicos. A continuación, se ofrece una descripción detallada de estos componentes:

- **Placa Arduino**: El corazón de la plataforma es la placa Arduino en sí misma. Esta placa es un microcontrolador programable que integra un procesador,

memoria, puertos de entrada/salida y otros componentes necesarios para ejecutar programas. Hay varios modelos de placas Arduino disponibles, cada uno con sus propias características y capacidades únicas.

- **Microcontrolador:** La placa Arduino está basada en un microcontrolador, que es el chip principal responsable de ejecutar el programa cargado en la placa. Los microcontroladores más comunes utilizados en las placas Arduino son de la familia AVR de Atmel (ahora parte de Microchip) y la familia ARM.

- **Puertos de Entrada/Salida (E/S):** Las placas Arduino están equipadas con una variedad de pines de entrada/salida que pueden ser utilizados para conectar sensores, actuadores, pantallas y otros dispositivos electrónicos. Estos pines pueden ser configurados como entradas digitales, salidas digitales o entradas/salidas analógicas, lo que permite una amplia variedad de aplicaciones.

- **Conexiones de Alimentación:** Las placas Arduino pueden ser alimentadas a través de una fuente de alimentación externa o a través del puerto USB de un ordenador. También suelen incluir un regulador de voltaje que permite alimentar dispositivos externos a través de los pines de alimentación.

- **Conexión USB:** La mayoría de las placas Arduino vienen equipadas con un puerto USB que permite la comunicación con un ordenador y la carga de programas en el microcontrolador. Esta conexión también se puede utilizar para enviar y recibir datos entre la placa Arduino y el ordenador, lo que facilita el desarrollo y depuración de proyectos.

- **Entorno de Desarrollo Integrado (IDE):** Arduino proporciona un entorno de desarrollo integrado (IDE) que permite escribir, compilar y cargar programas en las placas Arduino de una manera sencilla y eficiente. El IDE Arduino es multiplataforma y está disponible para Windows, macOS y Linux, lo que lo hace accesible para una amplia audiencia de usuarios.

- **Lenguaje de Programación:** Los programas para Arduino se escriben en un lenguaje de programación simplificado basado en C/C++, que ha sido adaptado para facilitar su aprendizaje y uso. El lenguaje de programación Arduino incluye una serie de funciones y bibliotecas predefinidas que simplifican la interacción con los pines de E/S, los sensores y otros dispositivos.
- **Bibliotecas y Shields**: Arduino cuenta con una amplia variedad de bibliotecas y shields (placas de expansión) que amplían las capacidades de las placas Arduino y facilitan la conexión de sensores, actuadores y otros dispositivos externos. Estas bibliotecas y shields abarcan una amplia gama de aplicaciones, desde la robótica y la domótica hasta el arte interactivo y la Internet de las cosas (IoT).

- **Especificaciones técnicas**. El Arduino empleado durante el desarrollo del presente proyecto cuenta con las siguientes especificaciones técnicas:

    - **Microcontrolador**. Emplea el microcontrolador ATmega328P.
    - **Voltaje de funcionamiento**. El voltaje de funcionamiento es de 5V.
    - **Memoria Flash**. Dispone de una memoria flash de 32 kB, de la cual 0.5 kB son empleados por el gestor de arranque (bootloader).
    - **SRAM**. Cuenta con una SRAM de 2Kb.
    - **EEPROM**. Cuenta con una EEPROM de 1Kb.
    - **Velocidad del reloj**. La velocidad del reloj del Arduino empleado es de 16 MHz.
    - **Pines de E/S digitales**. Cuenta con 14 pines de E/S digitales de los cuales 6 proporcionan salida PWM.
    - **Pines de E/S analógica**. Cuenta con 6 pines de E/S analógica.
    - **Corriente continua por pin**. La corriente continua por pin es de 20 mA.

2.3. Visual Studio 2022

Visual Studio 2022 es la última versión de la conocida suite de desarrollo integrado (IDE) de Microsoft, diseñada para ayudar a los desarrolladores a crear una amplia gama de aplicaciones, desde software de escritorio hasta aplicaciones web y móviles, utilizando una variedad de lenguajes de programación y tecnologías. Durante el transcurso del proyecto se ha escogido el empleo de Visual Studio 2022 para el desarrollo de la interfaz debido a su gran facilidad para implementar interfaces, además de una preferencia de desarrollo mediante el lenguaje C#.

A continuación, proporciono una descripción detallada de las características y capacidades clave de Visual Studio 2022:

- **Entorno de Desarrollo Integrado (IDE):** Visual Studio 2022 proporciona un entorno de desarrollo integrado altamente personalizable y potente que permite a los desarrolladores escribir, depurar y compilar código de manera eficiente. Su interfaz de usuario intuitiva y fácil de usar incluye herramientas y ventanas para la edición de código, la depuración, la gestión de proyectos y la colaboración en equipo.

- **Soporte Multiplataforma**: Visual Studio 2022 es compatible con una amplia gama de plataformas de desarrollo, incluyendo Windows, macOS y Linux. Esto permite a los desarrolladores crear aplicaciones para diferentes sistemas operativos y dispositivos utilizando un único entorno de desarrollo.

- **Lenguajes de Programación**: Visual Studio 2022 admite una variedad de lenguajes de programación populares, incluyendo C#, Visual Basic, C++, JavaScript, TypeScript, Python, y muchos más. Esto permite a los

desarrolladores elegir el lenguaje que mejor se adapte a sus necesidades y preferencias.

- **Frameworks y Tecnologías**: La suite de Visual Studio 2022 incluye soporte para una amplia gama de frameworks y tecnologías de desarrollo, como .NET, ASP.NET, Xamarin, Node.js, React, Angular, y muchos otros. Esto permite a los desarrolladores crear una variedad de aplicaciones, desde aplicaciones web y móviles hasta aplicaciones de escritorio y servicios en la nube.

- **Herramientas de Depuración Avanzadas**: Visual Studio 2022 proporciona potentes herramientas de depuración que permiten a los desarrolladores detectar y corregir errores en su código de manera eficiente. Estas herramientas incluyen capacidades avanzadas como puntos de interrupción, inspección de variables, seguimiento de la pila de llamadas, y mucho más.

- **Integración con Azure**: Visual Studio 2022 está estrechamente integrado con Microsoft Azure, la plataforma de servicios en la nube de Microsoft. Esto permite a los desarrolladores construir, implementar y administrar aplicaciones en la nube de manera eficiente directamente desde el IDE.

- **Desarrollo Colaborativo**: Visual Studio 2022 incluye herramientas que facilitan la colaboración en equipo, como Git integrado, control de versiones, seguimiento de problemas, y revisión de código. Esto permite a los desarrolladores trabajar de manera efectiva en proyectos compartidos y colaborar con colegas en tiempo real.

- **Extensiones y Personalización**: Visual Studio 2022 es altamente extensible y personalizable, permitiendo a los desarrolladores ampliar las capacidades del IDE mediante la instalación de extensiones y la creación de herramientas personalizadas. Esto permite a los desarrolladores adaptar el entorno de desarrollo a sus necesidades específicas y mejorar su productividad.

En resumen, Visual Studio 2022 es una herramienta poderosa y versátil que ofrece un conjunto completo de características y capacidades para ayudar a los desarrolladores a crear aplicaciones de alta calidad de manera eficiente y efectiva. Con su soporte multiplataforma, amplia gama de lenguajes y tecnologías, y herramientas avanzadas de desarrollo y depuración, Visual Studio 2022 sigue siendo una opción popular entre los desarrolladores de todo el mundo.

3. Especificaciones y restricciones de diseño

El sistema cuenta con las siguientes especificaciones de funcionamiento:

- El sistema será diseñado y ensamblado de forma autónoma, adquiriendo todas las piezas y componentes por separado para su posterior montaje y programación.

- El sistema contará con alta precisión a la hora de realizar el dibujo tanto de las pistas en la PCB como durante el taladrado de los agujeros de esta.

- El sistema dispondrá de una interfaz sencilla y amigable desde la cual será posible realizar la monitorización y el control de la CNC.

    - El lenguaje de programación empleado para su desarrollo será C#.

    - El IDE de desarrollo utilizado durante el transcurso del proyecto será Visual Studio 2022.

    - Desde la interfaz será posible tanto controlar los motores, como procesar los ficheros de tipo Gcode mediante los cuales serán enviados comandos al Arduino.

- El sistema contará con un software de control de la CNC basado en el microcontrolador Arduino.

    - El lenguaje de programación empleado para su desarrollo será el propio de Arduino.

    - El IDE de desarrollo utilizado durante el transcurso del proyecto será el IDE propio de Arduino.

    - El software de control será capaz de recibir cada uno de los comandos por parte de la interfaz y realizar con los motores los movimientos pertinentes.

- Se empleará un generador de PCBs comercial, realizando tanto el diseño esquemático del circuito como el diseño de la propia PCB. Se ha decidido optar por el software de diseño *Kicad*, debido a sus altas prestaciones y su multitud de librerías.

- Se empleará un generador de Gcode comercial para la obtención del fichero con extensión ".gcode" el cuál será legible por el sistema. Se ha decidido optar por el software *FlatCAM*, debido a su alto uso en este tipo de ficheros.

- La comunicación entre la interfaz y el Arduino será mediante puerto serie, en formato Full-dúplex y con sistema de ACKs.

- Durante el transcurso del proyecto, se empleará un repositorio de Git con el objetivo de obtener un control de versiones. La URL es la siguiente:

  https://gitlab.com/sergio.sanz.lopez/tfg_cnc_machine

El sistema cuenta con las siguientes restricciones de diseño:

- Con el objetivo de facilitar el diseño y reducir los tiempos de trabajo, se ha decidido realizar el dibujo de las pistas, sin tener en cuenta el plano de masa, facilitando así el diseño y reduciendo considerablemente los tiempos de fabricación, minimizando por ello los posibles errores durante la ejecución.

- Debido a la decisión anterior, es necesario diseñar las placas de circuito impreso con un tamaño superior al estándar.

4. Descripción de la solución propuesta

4.1. Metodología y entorno de trabajo

4.1.1. Metodología seguida

Para la ejecución del presente proyecto, se ha seguido meticulosamente una metodología de desarrollo de cascada. Este enfoque conocido por su rigor y estructura secuencial, proporcionando un marco de trabajo claro y lineal que facilita la gestión eficiente del proceso de desarrollo. La metodología de cascada se distingue por su división en fases claramente definidas, donde cada etapa debe completarse de manera integral antes de proceder hacia la siguiente, asegurando así una progresión ordenada y controlada. Dadas las características del proyecto, se dividió en 6 fases diferenciadas: elección de objetivos, diseño de la CNC, diseño de la interfaz, fabricación de la CNC, desarrollo del software y, por último, pruebas y validaciones.

Durante el transcurso del proyecto, la mayor parte se ha llevado a cabo siguiendo el principio del código abierto, lo que implica el uso de plataformas accesibles para todos, sin ningún costo asociado. Este enfoque se refleja en las herramientas empleadas durante el transcurso del presente proyecto, tales como Visual Studio y IDE de Arduino, debido a que ambas cuentan con una gran comunidad de usuarios activos.

4.1.2. Entorno de trabajo

El entorno de trabajo está basado en una máquina de control numérico controlada mediante Arduino, el cual recibe órdenes desde la interfaz generada en Visual Studio.

Para poder establecer una conexión entre ambas partes, ha sido necesario el empleo de una comunicación puerto serie conocida comúnmente como *full-dúplex*, la cual permite la transmisión de datos en ambas direcciones. Dicha comunicación cuenta además con un sistema de *Acknowledgment*, nombrado popularmente como *ACKs* cuyo propósito es la confirmación de la recepción y posterior procesado de mensajes por parte del controlador con el objetivo de habilitar la transmisión del siguiente paquete de datos disponible.

A continuación, en la Ilustración 1 se muestra el diagrama de bloques de la solución completa:

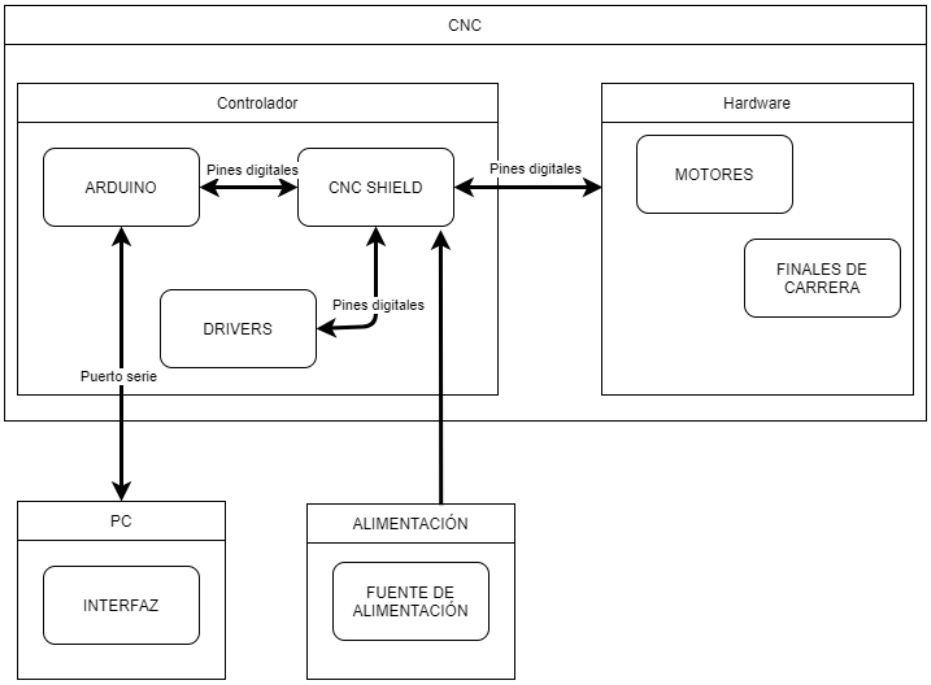


Ilustración 1. Diagrama de bloques del sistema

4.2. Máquina de control numérico

4.2.1. Diseño

Para formalizar el diseño de la máquina, se ha empleado el software de diseño 3D "*Sketchup*" en su versión sin costo en formato Web. Se ha optado por esta herramienta debido a la innumerable cantidad de modelos 3D diseñados por la comunidad que posee, tales como perfiles de aluminio y esquinas de montaje, ambos requeridos para el diseño.

Los componentes 3D gratuitos desarrollados por la comunidad y empleados en este diseño han sido los siguientes:

- **Perfil de extrusión 3030.** Se puede apreciar su diseño en la Ilustración 2.

Ilustración 2. Perfil de extrusión 3030

- **Esquina de montaje 3030.** Se puede apreciar su diseño en la Ilustración 3.

Ilustración 3. Esquina de montaje 3030

Mediante el empleo de estos dos componentes, se ha realizado el diseño de la estructura de la CNC. Para la base, se ha optado por un diseño robusto, de 400 mm x 400 mm, permitiendo así una amplia superficie de trabajo, además, ha sido necesario el montaje de un perfil intermedio con el objetivo de soportar la carga y el peso del eje X una vez montado.

Los resultados se pueden apreciar en las siguientes Ilustraciones 4 y 5:

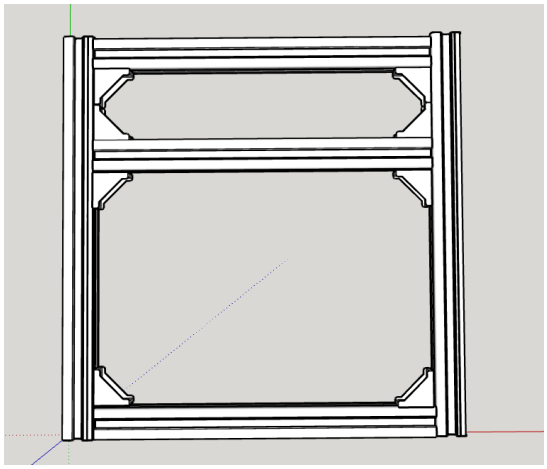


Ilustración 4. Base de la estructura (Planta)

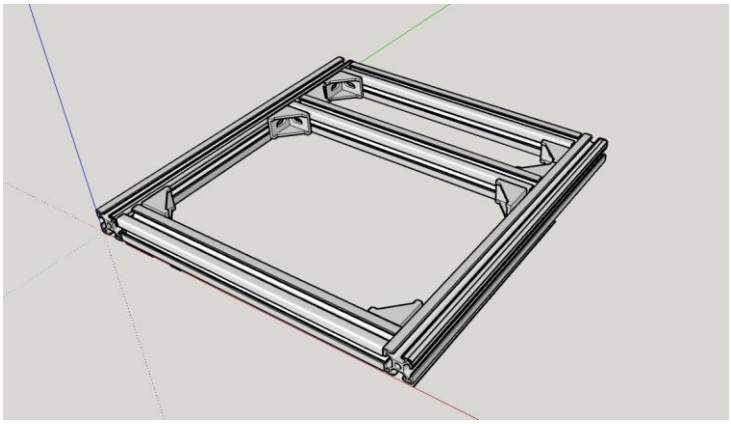


Ilustración 5. Base de la estructura (Isométrica)

Para el eje X, se ha optado por repetir el diseño recortando en altura para evitar un excedente de tamaño innecesario en este eje. Los resultados han sido los siguientes, visibles en las Ilustraciones 6 y 7:

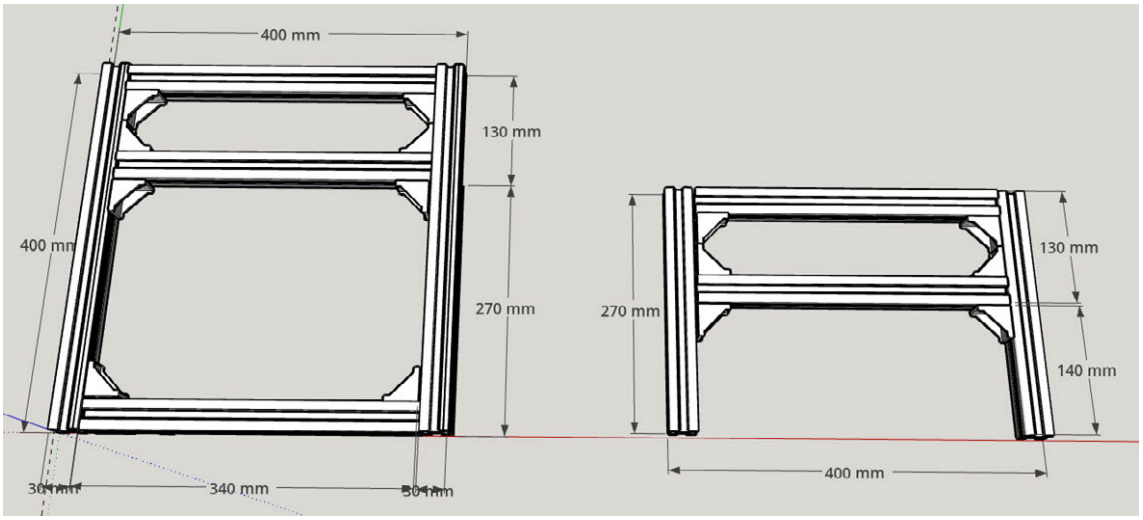


Ilustración 6. Estructura completa

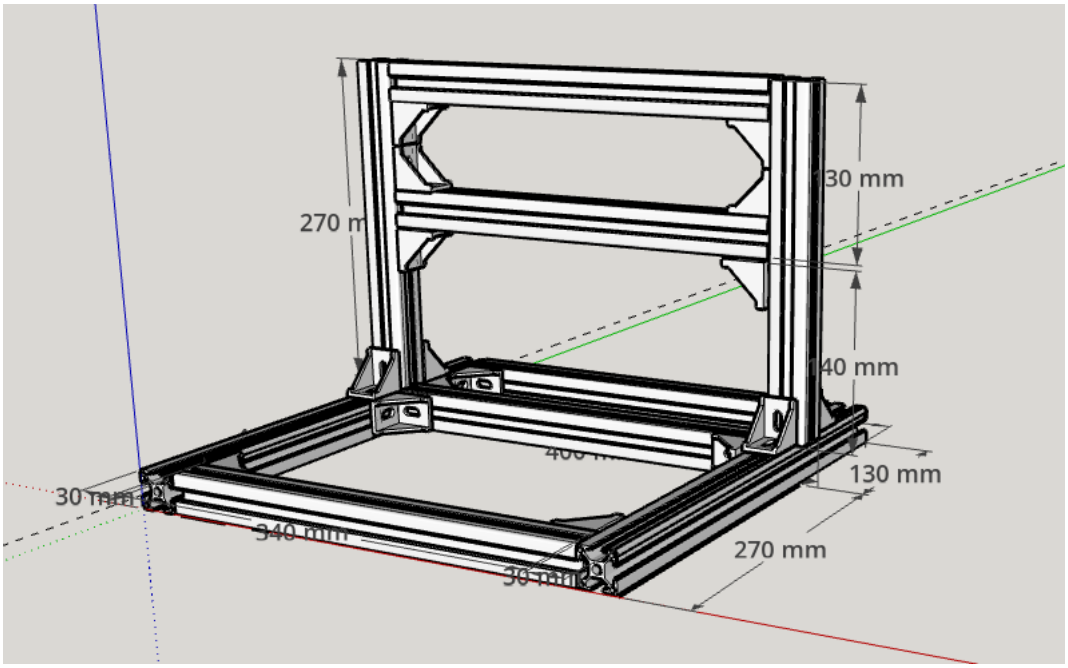


Ilustración 7. Ensamblaje final

Este diseño representa una estructura sólida y robusta, capaz de soportar varios ejes unidos con motores y con la posibilidad de utilizar una base de hasta 300 mm de largo, permitiendo así la generación de piezas de gran tamaño si fuese necesario.

4.2.2. Componentes

A lo largo de este apartado, se explicarán con detalle los componentes empleados, tanto para la estructura como para la electrónica del proyecto.

- **Arduino UNO**. Es una placa de hardware de código abierto basada en el microcontrolador *ATmega*. Es parte de la familia de placas Arduino y es utilizada ampliamente en proyectos de electrónica y robótica. Con entradas y salidas tanto digitales como analógicas, interfaz USB y capacidad de programación sencilla, el Arduino UNO es altamente popular para prototipado rápido y educación en electrónica. Se programa mediante el entorno de desarrollo (IDE) Arduino, permitiendo a los usuarios cargar y ejecutar fácilmente código en la placa. En la Ilustración 8 se puede apreciar un ejemplo de Arduino UNO.

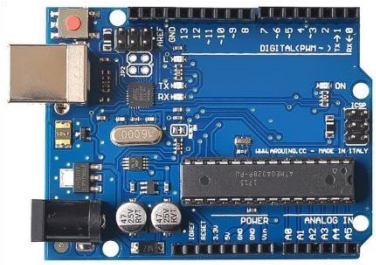


Ilustración 8. Arduino UNO

- **Motores NEMA17**. Son motores paso a paso estándar que cumplen con las especificaciones del NEMA (National Electrical Manufacturers Association), ampliamente utilizados en aplicaciones de impresión 3D y CNC. Su medida es de 43,2 x 43,2 mm, son conocidos por su precisión en el posicionamiento y su capacidad para generar un par adecuado para aplicaciones de baja carga. En la Ilustración 9 se puede apreciar un ejemplo de motor NEMA 17.

Ilustración 9. Motor NEMA17

- **Drivers DRV8825**. Es un controlador de motor paso a paso que se utiliza comúnmente en sistemas de control de movimiento, como impresoras 3D y máquinas CNC. Este chip puede manejar motores paso a paso bipolares y unipolares, ofreciendo control de corriente y microstepping ajustable. Su capacidad para proporcionar un alto rendimiento y una resolución fina de movimiento lo hace popular para aplicaciones que requieren precisión en la posición del motor. Además, ofrecen protección contra sobrecorriente, lo cual garantiza la seguridad y eficiencia. En la Ilustración 10 se puede apreciar un ejemplo de driver DRV8825.

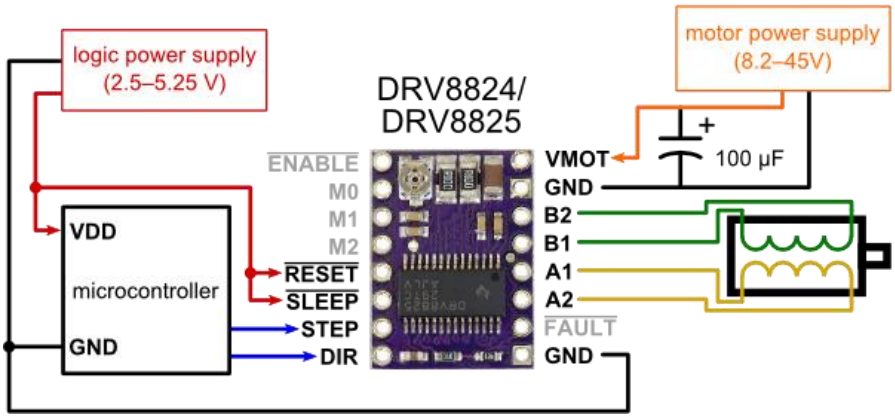


Ilustración 10. Driver DRV8825

- **CNC Shield V3**. Es una placa de expansión diseñada para ser utilizada con placas Arduino. Está diseñada para facilitar la creación de sistemas de control numérico por computadora, comúnmente utilizados en CNC. Facilita el control de motores paso a paso mediante el uso de controladores específicos que se insertan directamente en la placa. Además, proporciona conectores para finales de carrera y otros periféricos, simplificando la creación y mejora de sistemas CNC mediante Arduino. En la Ilustración 11 se puede apreciar un ejemplo de CNC Shield V3.

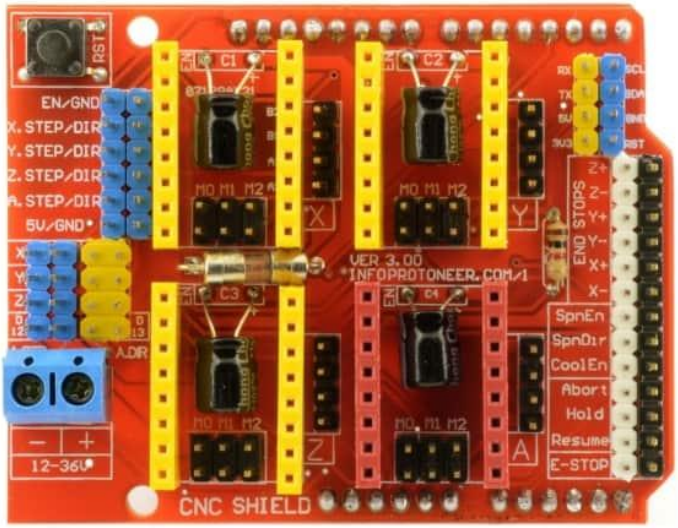


Ilustración 11. CNC Shield v3

- **Kit Eje Z**. Para este proyecto, se ha utilizado un soporte para el Eje Z comercial, la elección fue debida a la compatibilidad de este componente con el diseño de la estructura. En la Ilustración 12 se puede observar un ejemplo de Kit Eje Z empleado.

Ilustración 12. Kit eje Z

- **Fuente de alimentación 12V 10A**. Esta fuente supone un voltaje y corrientes más que suficientes para la alimentación del conjunto de la electrónica que compone la máquina, a excepción del Arduino y la fresadora, que tienen alimentaciones independientes. En la Ilustración 13 se puede apreciar un ejemplo de la fuente de alimentación empleada.

Ilustración 13. Fuente de alimentación 12V 10A

- **Finales de carrera**. Es un interruptor mecánico diseñado para detectar los límites físicos de movimiento en un sistema. Se activa al alcanzar el extremo del recorrido de un componente, deteniendo o modificando la operación en consecuencia. En el caso de este proyecto, se emplean para limitar los recorridos de los ejes X, Y y Z. En la Ilustración 14 se puede apreciar un ejemplo de final de carrera empleado.

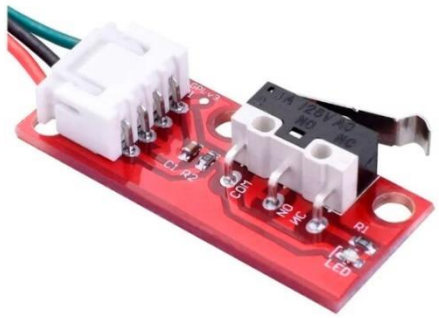


Ilustración 14. Finales de carrera

- **Perfiles de Aluminio (Varios tamaños)**. Se ha optado por perfiles de aluminio 3030 debido a su ligero peso y robustez. En el apartado de diseño de este mismo documento se puede observar el tamaño escogido para cada uno de los perfiles empleados durante el ensamblaje de la estructura. En la Ilustración 15 se puede apreciar un ejemplo de perfiles de aluminio empleados.

Ilustración 15. Perfiles alumnio 3030

- **Acoplamientos eje 5mm a 8mm**. Empleados para poder conectar los ejes de los motores NEMA17 a las varillas lisas y roscadas, con el fin de poder transmitir el giro. Estos acoplamientos cuentan con distinto tamaño de boquilla en cada uno de sus laterales, permitiendo así encajar los ejes de los motores por el lateral de 5 mm y transmitir el giro encajando las varillas por el lateral de 8 mm. En la Ilustración 16 se puede apreciar un ejemplo de los acoplamientos empleados.

Ilustración 16. Acoplamientos ejes

- **Soportes SK8/SK10/KP08**. Conjunto de soportes empleados en la sujeción tanto de las varillas lisas como de las varillas roscadas. Los soportes SK8 y SK10 se encargan de la sujeción de las varillas lisas, las cuales debes ser fijas y los soportes KP08 a su vez se encargar de la sujeción de las varillas roscadas, debido a que estos permiten la continuidad del giro. En las Ilustraciones 17 y 18 se puede apreciar un ejemplo de los tipos de soportes empleados.

Ilustración 18. Soportes SK8/SK10

Ilustración 17. Soportes KP08

- **Escuadra para perfil de aluminio**. Con 90º de ángulo, empleadas en la unión entre perfiles 3030. En la Ilustración 19 se puede apreciar un ejemplo de las escuadras empleadas.

Ilustración 19. Escuadras perfiles de alumnio

- **Tuercas en T**. Este tipo de tuerca son idóneas para su empleo conjunto con los perfiles de aluminio mencionados anteriormente, debido a su forma encajan a la perfección con dichos perfiles, permitiendo así una mejor sujeción y agarre, limitando el número y la intensidad de vibraciones producidas por la máquina en funcionamiento. En la Ilustración 20 se puede apreciar un ejemplo de las tuercas empleadas.

Ilustración 20. Tuercas en T

- **Abrazaderas en C**. Empleadas para la sujeción de la placa con la base, disponen del tamaño perfecto para la sujeción de una plancha de madera (o de cualquier otro material) a la base de la máquina, obteniendo la altura de trabajo necesaria tanto para el rotulador como para el taladro. En la Ilustración 21 se puede apreciar un ejemplo de las abrazaderas empleadas.

Ilustración 21. Abrazadera en C

- **Fresadora eléctrica**. Se ha optado por emplear un modelo de fresadora muy comercial y popular como es la Dremel. Esta fresadora no sólo se encargará de realizar el dibujo de las pistas en la placa mediante una punta metálica, sino que mediante el empleo de unas brocas del tamaño correspondiente llevará a cabo los agujeros necesarios en la PCB. En la Ilustración 22 se puede apreciar un ejemplo de una fresadora.

Ilustración 22. Fresadora

- **Varillas lisas y roscadas**. Se ha optado por el empleo de varillas de 8mm de diámetro. En la Ilustración 23 se puede apreciar un ejemplo de las varillas empleadas.

Ilustración 23. Ejemplo de varillas

- **Piezas 3D**. Durante el transcurso del proyecto, se han diseño y empleado piezas en 3D para solucionar problemáticas encontradas. Este punto será detallado más adelante en este mismo documento.
- **Tornillos/Tuercas/Arandelas (Varios tamaños)**. Han sido necesarios en abundancia para poder ensamblar todos los componentes de la máquina.

4.2.3. Montaje y ensamblaje

Una vez completada la etapa de diseño, se ha previsto una etapa de montaje y ensamblaje de la CNC. En esta fase se dará vida a la visión de una máquina precisa y personalizada. Este proceso no sólo implica unir componentes mecánicos y electrónicos, sino también a la integración cuidadosa de cada elemento para garantizar un funcionamiento preciso y eficiente. Las fases de la etapa de montaje serán las siguientes:

- **Preparación de componentes**. A lo largo de esta fase, será necesario comprobar la disponibilidad de todos los componentes seleccionados durante la etapa de diseño del proyecto. Además, se realizará la revisión de todas las piezas para garantizar su integridad.

  De manera adicional, se organizarán las piezas en dos grupos. El primer grupo contendrá las piezas que formarán la estructura de la base de la CNC, mientras que el segundo grupo contendrá las piezas que formarán parte del eje X de la máquina. Esta separación inicial logrará obtener una mayor sencillez durante el montaje, evitando de esta manera posibles fallos.

- **Estructura mecánica**. Esta fase se dividirá en dos grupos de montaje, mencionados en el punto anterior:
  - **Estructura mecánica de la base**. Durante el montaje de la base, será de suma importancia ensamblar el marco principal de la propia base, ajustando las escuadras lo máximo posible para obtener así una base robusta y nivelada. Posibles desniveles en esta temprana etapa de montaje podrán derivar en un fallo grave de precisión en la CNC. En la Ilustración 24 se puede apreciar el aspecto de la base de la CNC una vez terminado esta fase de montaje.

Ilustración 24. Estructura mecánica de la base

  - **Estructura mecánica del eje X**. Será recomendado ensamblar el eje independiente de la base, para poder cuadrar y nivelar todos los perfiles de aluminio de manera cómoda, evitando así posibles fallos. En la Ilustración 25 se puede apreciar la estructura mecánica del eje X una vez terminada la fase de montaje.

Ilustración 25. Estructura mecánica del eje X

- **Estructura completa.** Una vez completado el montaje mecánico, será el momento de unir las dos etapas del montaje.

- **Montaje de componentes electrónicos.** Durante el transcurso de esta fase, se llevará a cabo el montaje de todos los componentes electrónicos, tales como los motores, los finales de carrera, el Arduino y la fuente de alimentación. Cabe destacar que la unión Arduino + CNC Shield es un componente completo que únicamente será necesario unir a la máquina. En la Ilustración 26 se puede apreciar el montaje de uno de los finales de carrera disponibles en la CNC, montado sobre una pieza diseñada en 3D, que será mencionada en siguientes apartados de este documento.

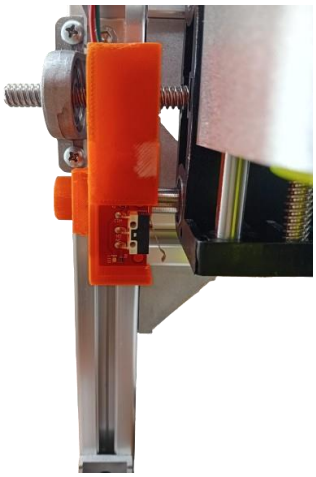


Ilustración 26. Final de carrera negativo eje X

En la Ilustración 27 se puede apreciar el montaje de uno de los motores que componen el sistema, en este caso concreto se trata del motor que controla el eje X.

Ilustración 27. Motor eje X

- **Cableado y conexiones**. Durante el transcurso de esta fase, el objetivo será realizar todo el conexionado de pines en la CNC Shield, los provenientes de los motores, de los finales de carrera y de la fuente de alimentación. Será necesaria una correcta señalización y organización de cables para permitir así rapidez a la hora de solucionar problemas y/o modificar componentes en el futuro. En la Ilustración 28 se puede ver un ejemplo del cableado relativo al eje Z junto con el final de carrera positivo del eje X. El cableado del final de carrera estará fijo a la estructura, mientras que el cableado del eje Z debe ser móvil debido al desplazamiento producido por el giro del eje X.

Ilustración 28. Ejemplo de cableado

- **Montaje de la base**. En esta fase, se unirá la base sobre la cual se harán los trabajos a la estructura final, la base estará unida mediante unos soportes ubicados en las varillas metálicas que forman la estructura. Dicha base de madera irá atornillada a estos soportes, de manera que quede fija y se desplace acorde con el movimiento del eje. En la Ilustración 29 se puede apreciar la base de madera sujeta a los soportes, sobre esta se encuentra una segunda plancha de madera en la cuál reposarán las placas a la hora de hacer el dibujo de las pistas y los taladros. Además, esta segunda plancha de madera cuenta con abrazaderas metálicas, que aportan la sujeción necesaria para que no exista movimiento ninguno teniendo en cuenta las posibles vibraciones generadas por los motores.

Ilustración 29. Base de madera de la CNC

- **Configuración y ajustes iniciales**. Durante el transcurso de esta fase, se ajustarán los límites de los motores teniendo en cuenta la posición de los finales de carrera, además se ajustará la altura del eje Z de tal manera que la posición de la punta del rotulador sea la misma que la posición de la punta de la broca del taladro. De esta manera, se logrará mayor precisión durante el trabajo.

Una vez finalizada la fase de montaje y ensamblaje y solucionados todos los posibles problemas derivados de dicho montaje, la CNC estará completa. En la Ilustración 30 se puede apreciar la imagen de la máquina en su totalidad, en la cual se aprecia la estructura, la base, los motores, los finales de carrera y todo el cableado. Además, en esta figura se puede ver que una prueba de dibujo realizada por la máquina, comprobando así el correcto funcionamiento del sistema.

Ilustración 30. CNC completa

4.2.4. Programación de Arduino

Una parte muy importante de este proyecto será el desarrollo de código en Arduino, el cual será el encargado de recibir los comandos *GCODE*, interpretarlos y mover los motores en consecuencia.

Durante el transcurso de este apartado, se tratarán los aspectos más relevantes de dicho código. Los aspectos por tratar serán:

- **Instancia de variables y constantes**. Será necesario el empleo de variables y constantes que serán empleadas a lo largo del código. Además, se definirán los pines asociados a los motores y los finales de carrera.

- **Función setup()**. Es la parte de código que contiene la configuración inicial de la máquina, como la configuración de los pines, el inicio de la comunicación puerto serie y un primer auto home de la CNC, con el objetivo de desplazar los motores inicialmente a las coordenadas (X=0 , Y=0, Z=0).

- **Función loop()**. Esta función consistirá en un bucle infinito de recepción y procesamiento de comandos, la máquina estará constantemente esperando un comando desde la interfaz, los cuales serán:
    - **Movimiento de motor.** La CNC interpretará el comando y girará el motor seleccionado el número de pasos enviados.
    - **Autohome.** La CNC interpretará el comando y llevará a los motores a la posición inicial (X=0, Y=0, Z=0).
    - **Líneas Gcode.** La CNC entrará en un bucle de recepción de comandos Gcode, los cuales serán procesados y la máquina actuará en consecuencia. Dichos comandos van seguidos de ACKs de confirmación de recepción y movimiento.

- **Función processIncomingLine(char* line, int charNB).** Esta función será la encargada de procesar la línea de Gcode recibida por puerto serie, y en base a la información que contenga, realizará el movimiento de los motores en consecuencia para que avancen a la posición deseada. Las posibles líneas por recibir serán las siguientes:

    - **M300 S50**. Este comando indica que el eje Z deberá realizar un movimiento positivo, es decir, deberá subir hasta la posición inicial (Z=0).

    - **M300 S30**. Este comando indica que el eje Z deberá realizar un movimiento negativo, es decir, deberá bajar hasta la posición de dibujo/taladro.

    - **G1 Xxx Yyy**. Este comando indica que el eje X deberá desplazarse hasta la coordenada X=xx y el eje Y deberá desplazarse hasta la coordenada Y = yy. La CNC será capaz de desplazarse en el eje X e Y tanto linealmente como en diagonal, interpretando la posición actual y la posición final (Empleando el algoritmo de Bresenham, detallado más adelante en este mismo apartado).

    - **G2 Xxx Yyy Iii Jjj.** Este comando indica que el eje X deberá desplazarse hasta la coordenada X=xx y el eje Y deberá desplazarse hasta la coordenada Y = yy, pero en este caso, deberá hacerlo realizando un arco desde la posición actual de los ejes (X = Xactual, Y=Yactual) hasta la posición final (X=xx, Y=yy) con centro en (X = Xactual + ii, Y= Yactual + jj) en **sentido horario**.

    - **G3 Xxx Yyy Iii Jjj.** Este comando indica que el eje X deberá desplazarse hasta la coordenada X=xx y el eje Y deberá desplazarse hasta la coordenada Y = yy, pero en este caso, deberá hacerlo realizando un arco desde la posición actual de los ejes (X = Xactual, Y=Yactual) hasta la posición final (X=xx, Y=yy) con centro en (X = Xactual + ii, Y= Yactual + jj) en **sentido antihorario**.

    - **G4.** Este comando indicia la finalización del trabajo, tanto de dibujo como de taladrado.

- **Función drawLine(float x1, float y1).** Esta función trazará una línea recta desde la posición actual de los motores (X=Xactual, Y=Yactual) hasta la nueva posición recibida (X=x1, Y=y1).
- **Función drawArc(float startX, float startY, float endX, float endY, float centerX, float centerY, float radius, bool clockwise).** Esta función trazará un arco, en sentido horario o antihorario (clockwise = true implica **sentido horario**,

clockwise = false implica **sentido antihorario**) desde la posición actual de los motores (X=startX, Y=startY) hasta la posición final (X=endX, Y=endY) con centro en (X=centerX, Y=centerY).

- **Función turnPositive*Axis* (float length).** Esta función desplazará en sentido positivo el eje *Axis* recibido una distancia = length.
- **Función turnNegative*Axis* (float length).** Esta función desplazará en sentido negativo el eje *Axis* recibido una distancia = length.
- **Función autohome().** Esta función desplazará los ejes hasta su posición inicial (X=0, Y=0, Z=0).
- **Algoritmo de Bresenham.** Este algoritmo selecciona, empleando incrementos, el punto o píxel más cercano a la posición real del punto en la recta deseada.
  Las distancias de cada uno de los puntos a la recta real se resumen en:

$$\Delta y = m\Delta x$$
$$\Delta y = \frac{\Delta y}{m}$$

Si el punto seleccionado es el más cercano a la posición real de la recta buscada, es posible elegir dicho punto únicamente teniendo en cuenta el signo de la diferencia existente entre $d_1 y d_2$:

$$d_1 = (y+1) - mx = (x+1) - x\frac{dy}{dx}$$
$$d_2 = mx - b = x\frac{dy}{dx} - 2y - 1$$

Particularizando en el punto inicial, se puede obtener el parámetro de decisión ($p = 2dy - dx$). En la siguiente Ilustración 31 se puede observar de forma gráfica.

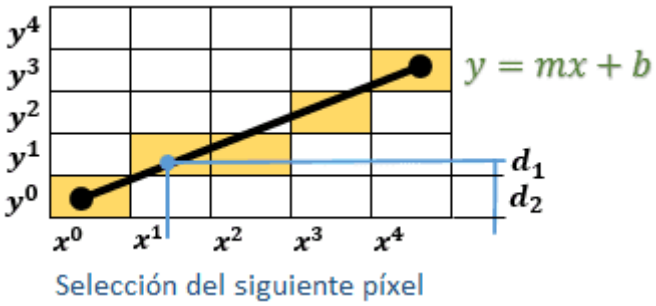


Ilustración 31. Algoritmo de Bresenham

4.2.5. Diseño de piezas en 3D

A lo largo de este proyecto, se ha tomado la decisión de emplear piezas 3D para solventar problemáticas encontradas durante el desarrollo. Las piezas diseñadas han sido las siguientes:

- **Soporte rotulador Eje Z.** Esta pieza será la encargada de sustituir a la fresadora en los momentos en los que necesitemos dibujar con el rotulador permanente. El

diseño de esta pieza se ha realizado mediante la herramienta *FreeCAD*, ha sido necesario realizar numerosas medidas en el Kit del Eje Z para poder transmitirlas al diseño. En las Ilustraciones 32, 33 y 34 es posible apreciar varios planos de la pieza.

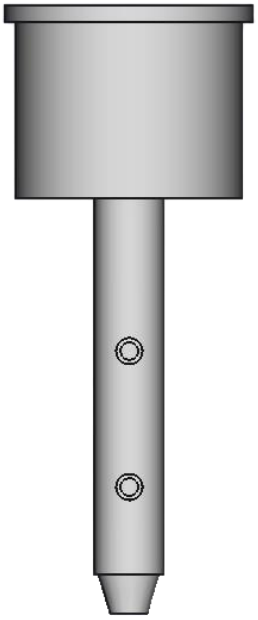


Ilustración 32. Figura 3D eje Z (Alzado)

La figura consta de una estructura cilíndrica, con el diámetro suficiente para poder sujetar el rotulador, este cilindro tiene, además, una pareja de cilindros de menor tamaño perpendiculares en su extremo inferior, los cuales han sido ideados para sujetar mediante dos tornillos el rotulador y obtener así mejor precisión al limitar el movimiento del mismo. Por último, en su extremo inferior está dotado también de un cono con abertura, diseñado para sujetar siempre el rotulador en la misma altura.

Para completar la figura, el cilindro de mayor tamaño que abarca la mitad de la pieza está diseñado para encajar a la perfección en el Kit del eje Z, obteniendo así una sujeción robusta del rotulador, manteniendo la misma posición y altura en todo caso e incrementando de esta manera la precisión de la máquina.

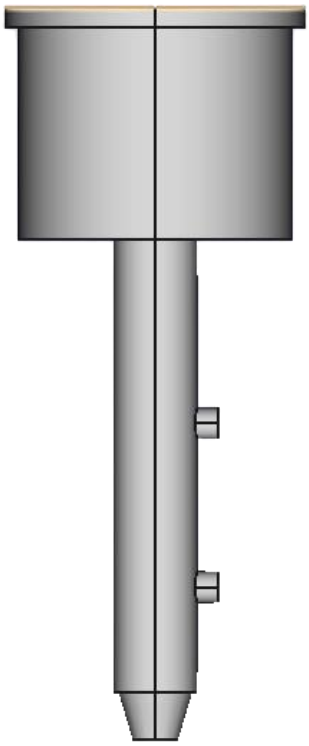


Ilustración 33. Figura 3D eje Z (Perfil)

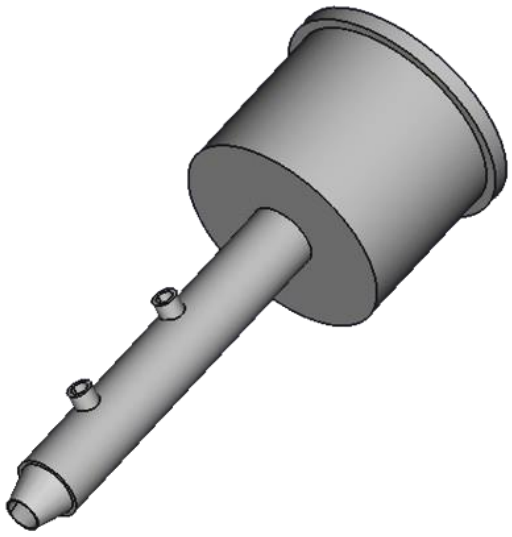


Ilustración 34. Figura 3D eje Z (Isométrica)

En la Ilustración 35 se puede apreciar el resultado obtenido, una pieza impresa en 3D, con un filamento PLA de color amarillo.

Ilustración 35. Pieza 3D eje Z

- **Soporte para los finales de carrera.** Serán diseñados distintos modelos de soporte de los finales de carrera, dependiendo del eje y de la orientación. En la Ilustración 36 se puede apreciar un ejemplo de uno de los modelos diseñados para soportar a los finales de carrera, en este caso es el soporte del eje Y. En la Ilustración 37 se puede apreciar un ejemplo de otro modelo diseñado.

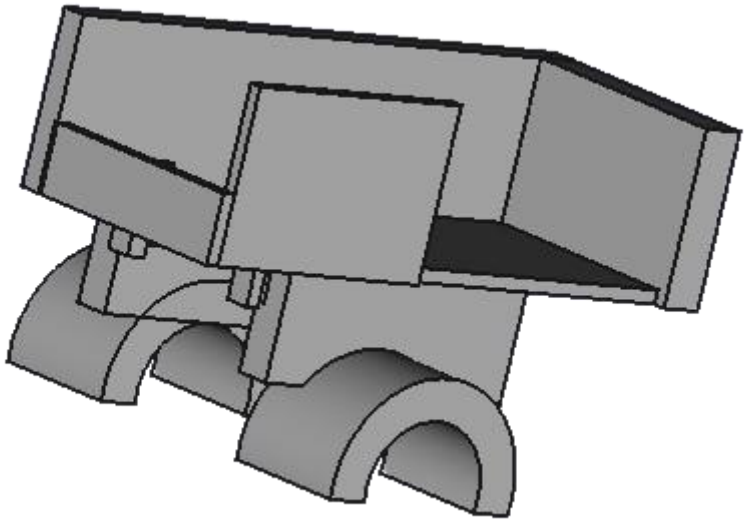


Ilustración 36. Soporte FDC eje Y

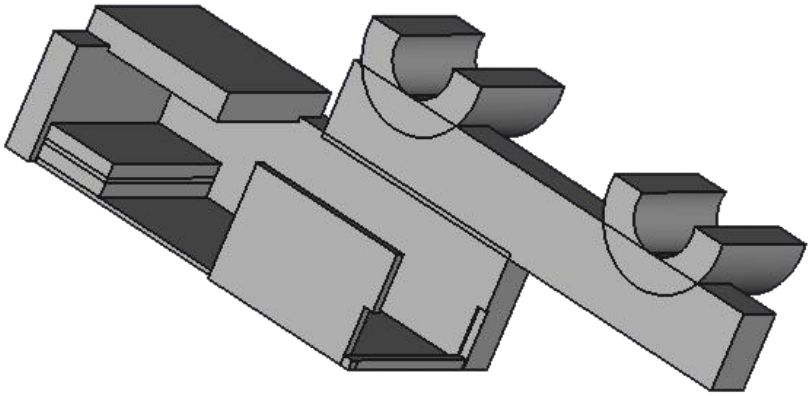


Ilustración 37. Soporte FDC eje Z

Tanto en la Ilustración 38 como en la Ilustración 39 se pueden apreciar los resultados obtenidos de la impresión de estas dos últimas piezas mencionadas.

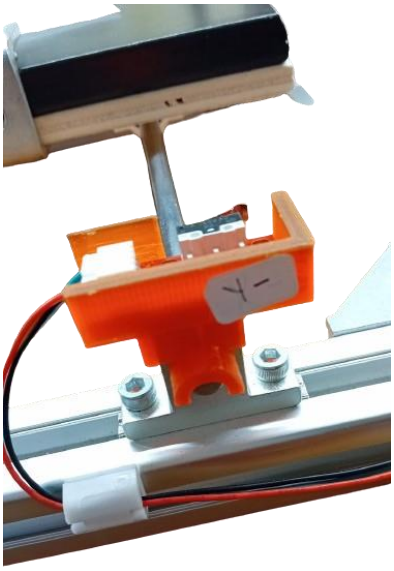


Ilustración 38. Pieza 3D soporte FDC eje Y

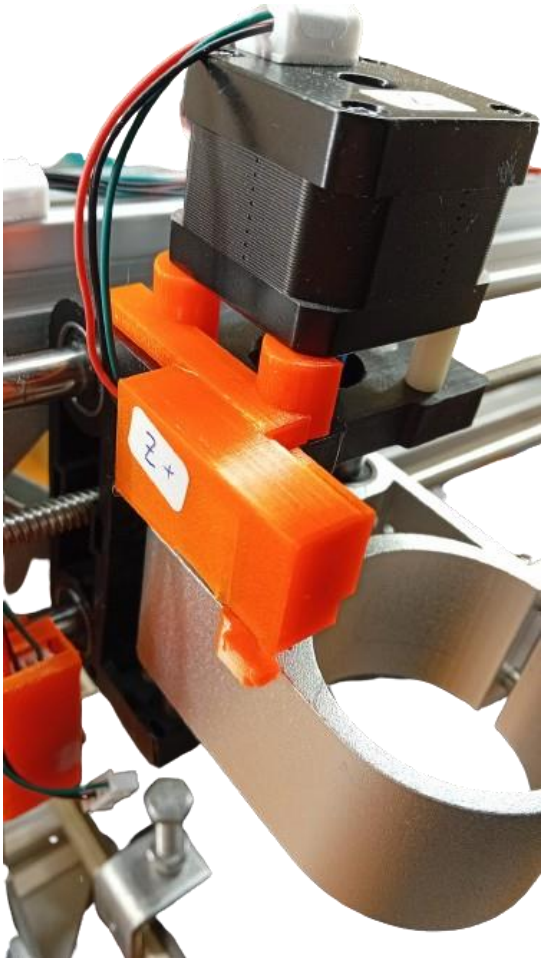


Ilustración 39. Pieza 3D soporte FDC eje Z

- **Soporte de apoyo final de carrera eje Z.** Será diseñada una pieza con el objetivo de ir encajada en el kit eje Z, con el fin de obtener mejor base de apoyo para el final de carrera empleado. La pieza diseñada se puede apreciar en la Ilustración 40. Además, en la Ilustración 41 se puede apreciar el resultado de la pieza obtenida.

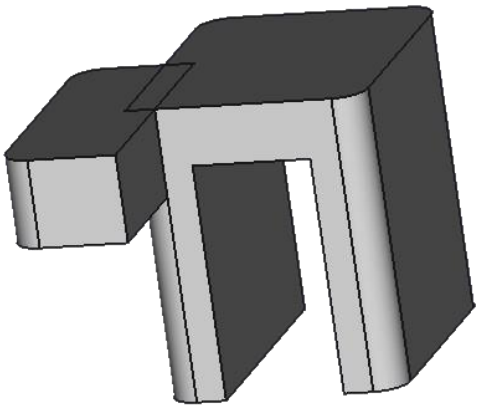


Ilustración 40. Pieza 3D soporte de apoyo FDC eje Z

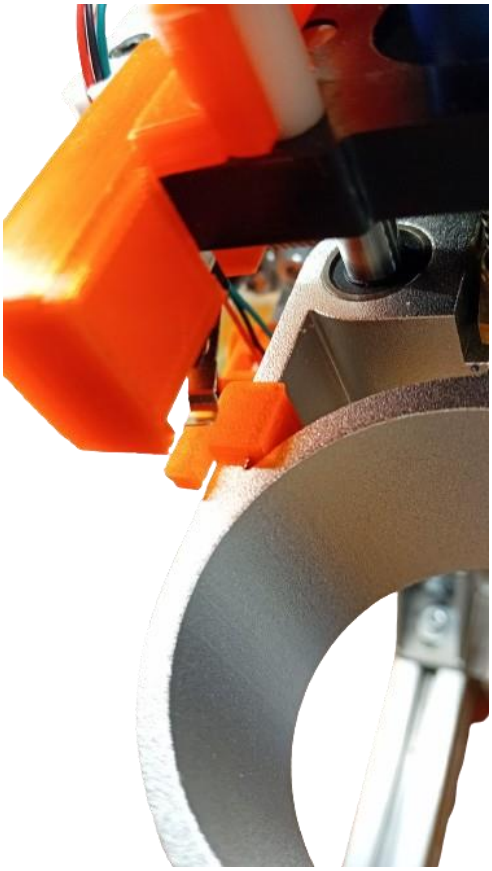


Ilustración 41. Pieza 3D soporte apoyo FDC eje Z resultado

- **Soporte para el Arduino.** Será diseñado un soporte para el Arduino, con el objetivo de anclarlo a un lateral de la estructura mecánica**.** Para el soporte propio del Arduino, se ha obtenido un soporte comercial disponible en la plataforma de diseños 3D '*Thingiverse'*, al cual se le ha añadido un soporte longitudinal con agujeros para tornillos. En la Ilustración 42 se puede apreciar el diseño de la pieza mencionada.

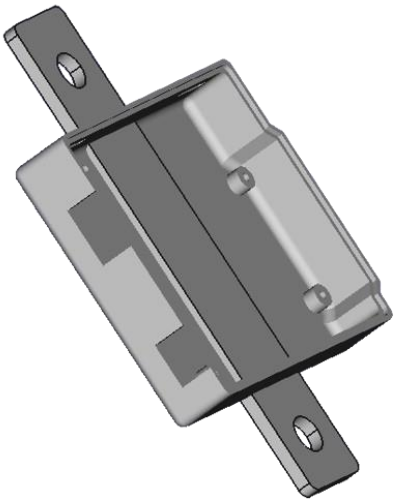


Ilustración 42. Pieza 3D soporte para Arduino

El resultado obtenido se puede apreciar en la Ilustración 43, en la cual se puede observar como el conjunto Arduino y CNC Shield reposan en un lateral de la estructura mecánica gracias al empleo de esta pieza.

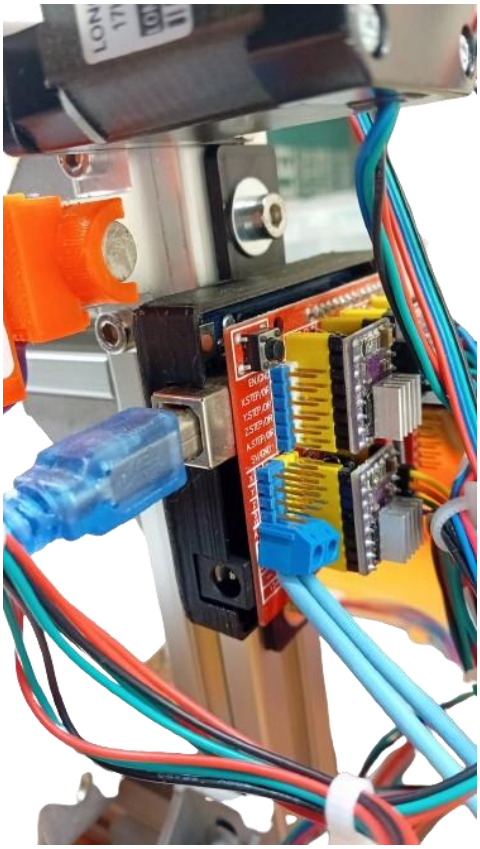


Ilustración 43. Resultado soporte para Arduino

- **Soporte placas de prototipado** Será diseñado un soporte con el objetivo de colocar las placas siempre en el mismo lugar de cara a seguir un orden durante los trabajos. En la Ilustración 44 se puede apreciar el diseño 3D del mismo.

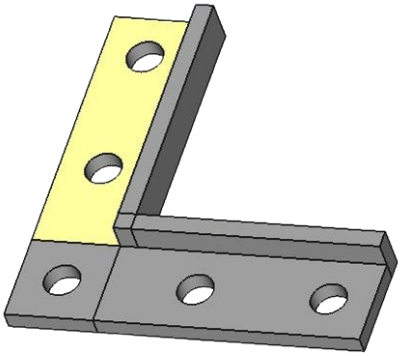


Ilustración 44. Pieza 3D soporte para las placas

El resultado obtenido se puede apreciar en la Ilustración 45, en la cual se puede observar la colocación de una de las placas desarrolladas en el mismo.

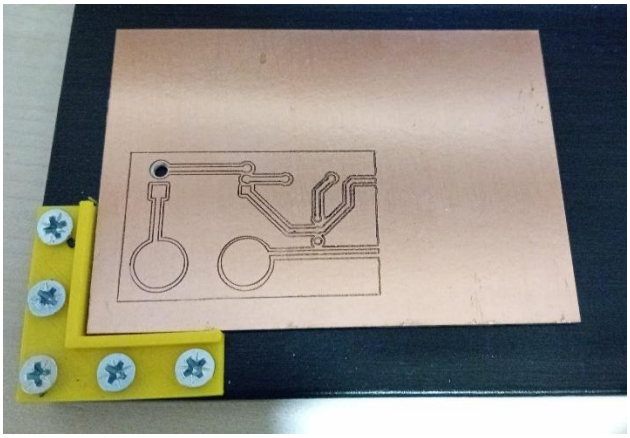


Ilustración 45. Resultado soporte para placas

4.2.6. Pruebas y ajustes

Para el desarrollo de las pruebas, ha sido necesario ajustar el movimiento de los motores programáticamente, por decisiones de diseño el motor X y el motor Y cuentan con varillas de distinto paso, esto implica que el mismo número de giros en ambos motores, otorgarán un desplazamiento lineal distinto de los mismos.

Debido al empleo de estas varillas, se ha modificado en el código dicho movimiento para permitir desplazar la máquina en unidades de milímetros teniendo en cuenta el número de pasos de cada motor. Siendo x1 e y1 variables en milímetros:

```
//  Convierte coordenadas a pasos
  x1 = (int)(x1 * 200.0);  //Full-Stepping: StepsPerMmX = 50.0  --> Micro-Stepping 1/4 --> 200 stepspermm
  y1 = (int)(y1 * 100.0);  // Full-Stepping StepsPerMmY = 25.0 --> Micro-Stepping 1/4 --> 100 stepspermm
```

Se ha decidido emplear microstepping para el desarrollo del proyecto, ajustando los drivers del controlador mediante jumpers en pines específicos se ha seleccionado un

microstepping de ¼ para los motores, esta decisión de diseño brinda una gran precisión a los motores sin ningún tipo de pérdida de potencia. En la Tabla 1 se pueden apreciar las distintas configuraciones posibles de microstepping disponibles en los drivers DRV8825 empleados en el proyecto.

MODE0	MODE1	MODE2	Microsteps
Low	Low	Low	Full step
High	Low	Low	Half step
Low	High	Low	1/4 step
High	High	Low	1/8 step
Low	Low	High	1/16 step
High	Low	High	1/32 step
Low	High	High	1/32 step
High	High	High	1/32 step

Tabla 1. Tipos de Microstepping

De forma adicional, en la Ilustración 46 se puede observar la distribución de dichas parejas de pines en la CNC Shield v3.

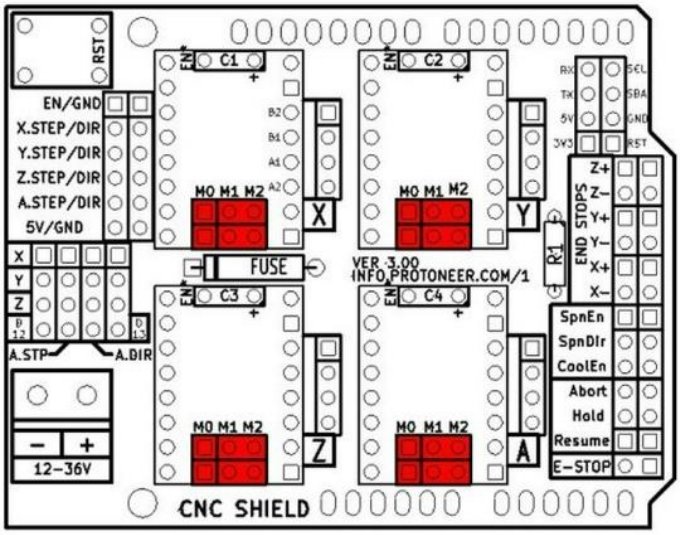


Ilustración 46. Pines microstepping CNC Shield v3

4.3. Interfaz de control en Visual Studio

Durante el transcurso de este apartado se mostrará en detalle el proceso de desarrollo de la interfaz de la CNC en Visual Studio. El objetivo ha sido proporcionar una interfaz de usuario intuitiva y funcional que facilite el control y la supervisión de la CNC, otorgando una experiencia eficiente y accesible.

4.3.1. Consideraciones de diseño

Durante el transcurso de la fase de diseño de la interfaz, se han tenido en cuenta diversos factores importantes a la hora de mejorar la experiencia de usuario. Las claves de diseño serán las siguientes:

- Interfaz sencilla e intuitiva.
- Experiencia de usuario satisfactoria.
- Títulos sencillos y claros.
- Estructura en columnas por cada eje.
- Botones de gran tamaño para facilitar su visualización.
- Procesado y visualización de ficheros gerber.
- Modo oscuro.
- Colores suaves con contraste.
- Fuente: Microsoft Sans Serif.
- Estilo de la fuente: Negrita
- Tamaño de la fuente: 10.

4.3.2. Diseño de la interfaz

Durante la fase de diseño de la interfaz, ha sido necesario determinar el número de pantallas necesarias y sus funcionalidades:

1. **Pantalla de control**. Desde esta pantalla, el objetivo es poder controlar los motores de una forma sencilla, para su comprobación y recalibración. En la Ilustración 47 se pueden observar los siguientes botones:

    - **Move +**. Al pulsar los botones de avance positivo de cada uno de los ejes, avanzará en sentido positivo y según el número seleccionado (Unidades en pasos).

    - **Move -**. Al pulsar los botones de avance positivo de cada uno de los ejes, avanzará en sentido positivo y según el número seleccionado (Unidades en pasos).

    - **Auto Home**. Al pulsar este botón, los 3 ejes se moverán hasta su posición de inicio.

Ilustración 47. Pantalla de Control

2. **Pantalla de monitoreo**. Desde esta pantalla, el objetivo es poder subir ficheros al sistema y ver una previsualización del trabajo que va a realizar la CNC. En la Ilustración 48 se pueden observar los siguientes botones:

   - **Upload gerber file**. Desde este botón, será posible adjuntar una carpeta .zip que contenga los ficheros gerber de la placa que se quiere fabricar. El programa internamente separará los ficheros y obtendrá los datos, para después mostrar una previsualización tanto de los pads como de los agujeros en la placa.

   - **Start plot**. Al pulsar este botón, la máquina comenzará a dibujar con el rotulador.
   - **Start Drill**. Al pulsar este botón la máquina comenzará a taladrar los agujeros.

Ilustración 48. Pantalla de monitoreo

4.3.3. Desarrollo de la interfaz

Como se ha destacado previamente en apartados anteriores de este mismo documento, la interfaz del proyecto se ha desarrollado en Visual Studio en el lenguaje de programación C. Una vez concebido y validado el diseño, se procede al desarrollo de las pantallas, a lo largo de esta sección se explicará en detalle la programación de dicha interfaz. Este desarrollo ha requerido la utilización diversas herramientas proporcionadas por Visual Studio, las cuales de detallan a continuación:

- **TabControl**. Es un componente que permite organizar y gestionar múltiples pestañas en una interfaz de usuario. Se utiliza comúnmente para crear interfaces con pestañas, como las ventanas de un navegador web, lo que facilita la navegación entre diferentes contenidos o vistas dentro de una aplicación.

- **TabPage**. Es un componente que representa cada una de las pestañas individuales de un TabControl y puede contener contenido específico como elementos visuales, botones o funciones particulares de la aplicación.

- **Button**. Son elementos que permiten a los usuarios interactuar con una aplicación tras hacer clic en ellos. Pueden ejecutar acciones, desencadenar eventos o cambiar el estado de la aplicación. Los botones son una parte fundamental en la creación de interfaces de usuario interactivas y se pueden personalizar en términos de estilo y comportamiento para adaptarse a las necesidades de la aplicación. En las Ilustración 49 podemos ver un ejemplo del botón que mueve el eje X. En este caso, al pulsar este botón se ejecuta el código dentro de la función **'X_axis_1_positive_click()',** la cual a su vez invoca a la función **'sendDataToArduino()'**, que será detallada más adelante.

Ilustración 49. Clic en botón de la interfaz

```
//Buttons
//X AXIS
private void X_axis_1_positive_click(object sender, EventArgs e)
{
    sendDataToArduino("x_1_p");
}
```

- **Label**. Es un componente de la interfaz que únicamente se utiliza para mostrar texto o etiquetas descriptivas, proporcionando información. Son personalizables en cuanto a formato y estilo, facilitando la organización y la claridad en la aplicación.

- **PictureBox.** Es un elemento que permite mostrar imágenes en una interfaz de usuario. Puede cargar imágenes de archivos locales o recursos, y es comúnmente utilizada para representar gráficos, fotografías u otros elementos visuales. En la Ilustración 50 se puede apreciar un ejemplo del funcionamiento de dicho elemento.

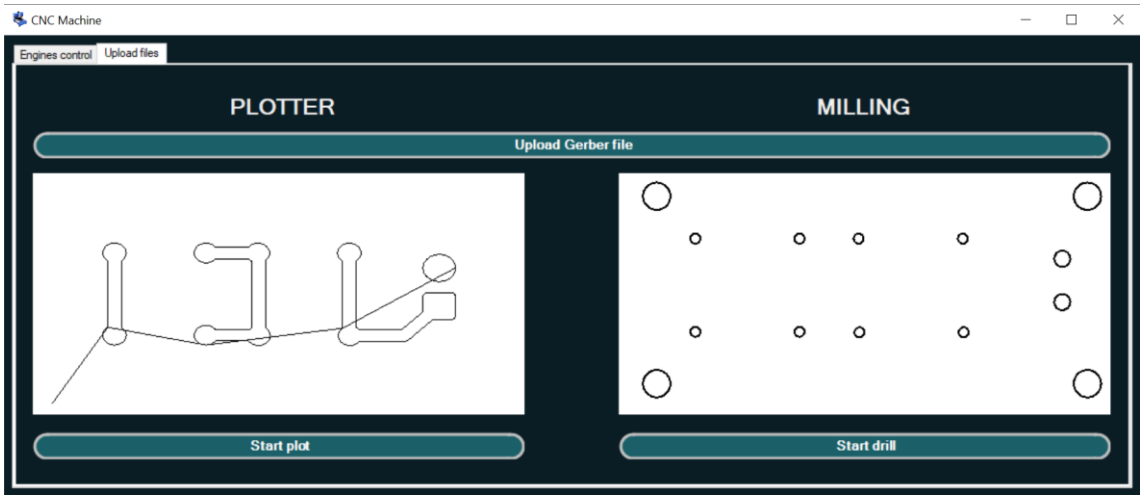


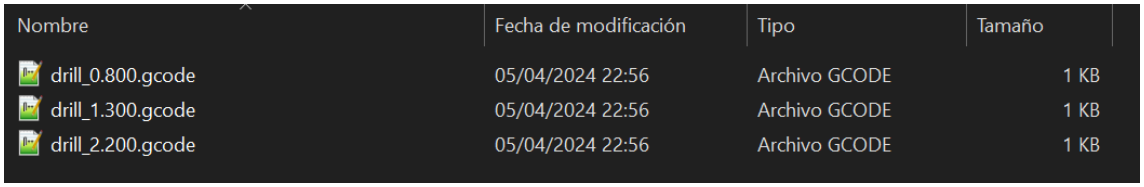
Ilustración 50. PictureBox en Visual Studio

El funcionamiento de la interfaz es sencillo, se podrá seleccionar un fichero de tipo .zip que contendrá los ficheros de taladro y de dibujo generados por KiCad y FlatCAM respectivamente, la interfaz comprobará la integridad de ambos ficheros y los procesará. En el caso del fichero de taladro, la interfaz reconocerá el número de agujeros, sus coordenadas y sus tamaños, con el objetivo de separar el fichero en varios dependiendo del tamaño. Por ejemplo, el contenido de un fichero de taladrado generado por KiCad se puede apreciar en la Ilustración 51.

```
M48
; DRILL file {KiCad 8.0.0} date 2024-03-03T19:58:16+0100
; FORMAT={-:-/ absolute / metric / decimal}
; #@! TF.CreationDate,2024-03-03T19:58:16+01:00
; #@! TF.GenerationSoftware,Kicad,Pcbnew,8.0.0
; #@! TF.FileFunction,Plated,1,2,PTH
FMAT,2
METRIC
; #@! TA.AperFunction,Plated,PTH,ComponentDrill
T1C0.800
; #@! TA.AperFunction,Plated,PTH,ComponentDrill
T2C1.300
; #@! TA.AperFunction,Plated,PTH,ComponentDrill
T3C2.200
%
G90
G05
T1
X6.92Y20.0
X6.92Y9.06
X17.08Y20.0
X17.08Y9.06
X22.84Y20.0
X22.92Y9.0
X33.0Y20.0
X33.08Y9.0
T2
X43.0Y18.0
X43.0Y12.92
T3
X4.0Y26.0
X4.0Y4.0
X46.0Y26.0
X46.0Y4.0
M30
```

Ilustración 51. Ejemplo de fichero de tipo .drl

La interfaz al reconocer que este fichero cuenta con 3 tamaños de agujeros distintos, lo procesará y generará 3 ficheros de tipo gcode, los cuales los almacenará en la ruta "output/drill/". En la Ilustración 52 se pueden apreciar los archivos generados tras el procesamiento de la Ilustración anterior por parte de la interfaz de control.



Nombre	Fecha de modificación	Tipo	Tamaño
drill_0.800.gcode	05/04/2024 22:56	Archivo GCODE	1 KB
drill_1.300.gcode	05/04/2024 22:56	Archivo GCODE	1 KB
drill_2.200.gcode	05/04/2024 22:56	Archivo GCODE	1 KB

Ilustración 52. Ficheros de taladrado gcode por tamaño

Estos archivos ya son legibles por la CNC y se le pueden enviar mediante puerto serie para que realice los taladros correspondientes, en la Ilustración 53 se puede apreciar el contenido de uno de ellos.

```
M300 S50
G1 X0 Y0
G1 X6,92 Y20
M300 S30
M300 S50
G1 X6,92 Y9,06
M300 S30
M300 S50
G1 X17,08 Y20
M300 S30
M300 S50
G1 X17,08 Y9,06
M300 S30
M300 S50
G1 X22,84 Y20
M300 S30
M300 S50
G1 X22,92 Y9
M300 S30
M300 S50
G1 X33 Y20
M300 S30
M300 S50
G1 X33,08 Y9
M300 S30
M300 S50
G1 X0 Y0

```

Ilustración 53. Contenido del fichero drill_0.800.gcode

En el caso del fichero de dibujo es más sencillo, puesto que al haber sido procesado previamente por el software de FlatCAM, ya se encuentra en formato legible por la CNC y únicamente será necesario modificar los comandos GCODE (Por ejemplo, modificar "G01" por "G1"), acto seguido será almacenado en la ruta "output/plot/".

4.3.4. Integración con la CNC

Una vez finalizado el proceso de desarrollo de la interfaz, junto con la creación del código correspondiente para Arduino, es imperativo efectuar la integración de ambos códigos. Esta acción posibilita el establecimiento de una comunicación efectiva entre la interfaz y el Arduino, permitiendo el intercambio de órdenes y datos. Para lograr esta sincronización, se utiliza el puerto serie como medio de transmisión de información, empleando un conjunto de comandos definidos con tal propósito.

En esta combinación, la interfaz asume la responsabilidad de emitir comandos específicos, los cuales son recibidos y ejecutados por el código de Arduino. El proceso se detalla a lo largo de este apartado y se sugiere consultar el Anexo 3 de este documento para acceder a información más exhaustiva del código empleado.

Dentro del contexto de la interfaz, se realiza previamente una configuración interna del puerto, en este caso se ha empleado el 'COM14' y se establece una velocidad de transmisión de 9600 baudios, cabe destacar que para que dicha configuración funcione correctamente, debe ser idéntica a la configurada en el código propio de Arduino.

Al interactuar con la interfaz y requerir el envío de comandos al Arduino, se invoca a la función '**SendDataToArduino()**'. Esta función, a su vez, inicia la apertura de la conexión del puerto serie mediante la función '**OpenConnection()**', posteriormente procede a transmitir el comando por medio de la función '**SendData()**', y finalmente cierra la conexión a través de la función '**CloseConnection()**'. Este procedimiento garantiza una comunicación fluida y segura entre la interfaz y el Arduino. En la Ilustración 54 se puede observar el contenido de la función '**SendDataToArduino()**'.

```
private void sendDataToArduino(String data)
{
    arduino.OpenConnection();
    dataToSend = data;
    arduino.SendData(dataToSend);
    arduino.CloseConnection();
}
```

Ilustración 54. Función 'SendDataToArduino()'

De igual manera, desde el Arduino se recibe esta información mediante el empleo del bucle infinito **loop()**, el cual se encarga de recibir líneas por el puerto serie, procesarlas y ejecutar órdenes en consecuencia.

4.4. Control de versiones

Durante el transcurso del proyecto y con el objetivo de mejorar la organización y aplicar conceptos adquiridos a lo largo de los estudios de grado, se ha decidido emplear un control de versiones basado en la herramienta Git.

Concretamente, se ha utilizado la plataforma integral de desarrollo GitLab, la cual proporciona un conjunto completo de herramientas para gestionar el ciclo de vida de las aplicaciones. Debido a la variedad existente en el presente proyecto, se han generado 5 ramas del repositorio, las cuales serán:

- **ssanzlop/arduino_code**. Esta rama contiene el código asociado al microprocesador Arduino
- **ssanzlop/interface_code**. Esta rama del repositorio contiene el código asociado a la interfaz en Visual Studio.
- **ssanzlop/3d_code**. Esta rama contiene los ficheros de diseño 3D generados tras el proceso de diseño de las piezas 3d utilizadas en el presente proyecto.
- **ssanzlop/memory**. Esta rama contiene el fichero Word de la memoria del presente proyecto.
- **ssanzlop/kicad**. Esta rama contiene los esquemáticos de los circuitos desarrollados durante las pruebas, además de sus diseños de PCB necesarios para la generación de los correspondientes gerber.

En la Ilustración 55 se puede apreciar el repositorio que contiene las ramas mencionadas.

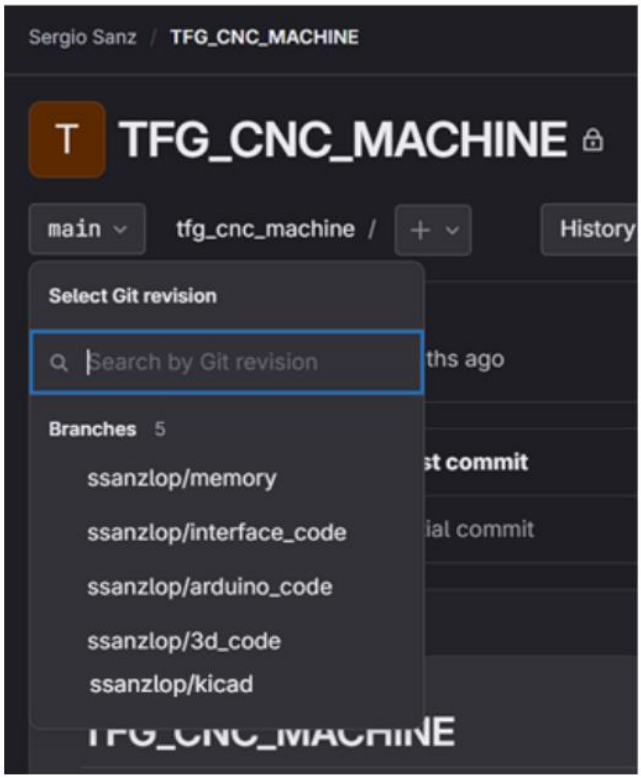


Ilustración 55. Control de versiones en GitLab

Todas y cada una de las ramas mencionadas cuentan con un sistema de mensajes descriptivo a la hora de realizar nuevas subidas de código. Durante el transcurso del proyecto, se han empleado dos etiquetas que se han considerado imprescindibles a lo largo del desarrollo. Dichas etiquetas son las siguientes:

- **feat:**. Etiqueta empleada para informar de que el código actualizado contiene nuevas funcionalidades o características. En la Ilustración 56 se puede apreciar un ejemplo de uso de dicha etiqueta.

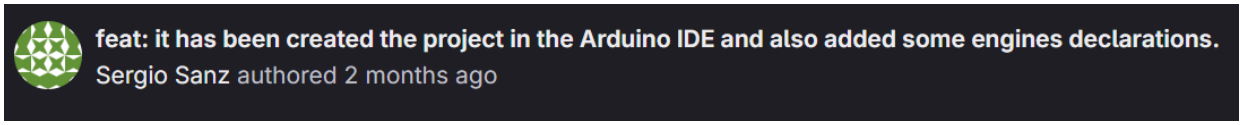


Ilustración 56. Etiqueta 'feat'

- **fix:**. Etiqueta empleada para informar de que el código actualizado contiene corrección de errores. En la Ilustración 57 se puede apreciar un ejemplo de uso de dicha etiqueta.

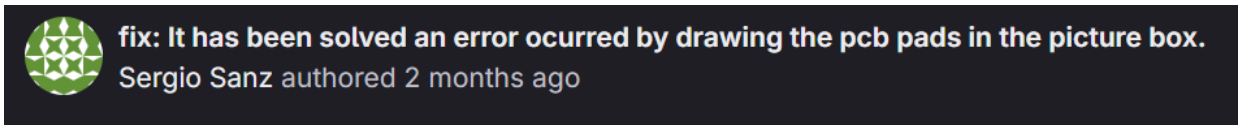


Ilustración 57. Etiqueta 'fix'

5. Resultados

Durante el transcurso de este apartado, se realizarán diversas pruebas con el objetivo de comprobar el funcionamiento de la máquina con circuitos reales.

CURRENT REGULATOR

En primer lugar, se ha empleado un circuito regulador de corriente [B5] para poder corroborar la precisión de la máquina, el diseño ha sido realizado mediante la herramienta de *KiCad*. En la *Ilustración 58* se puede observar el esquemático del circuito.

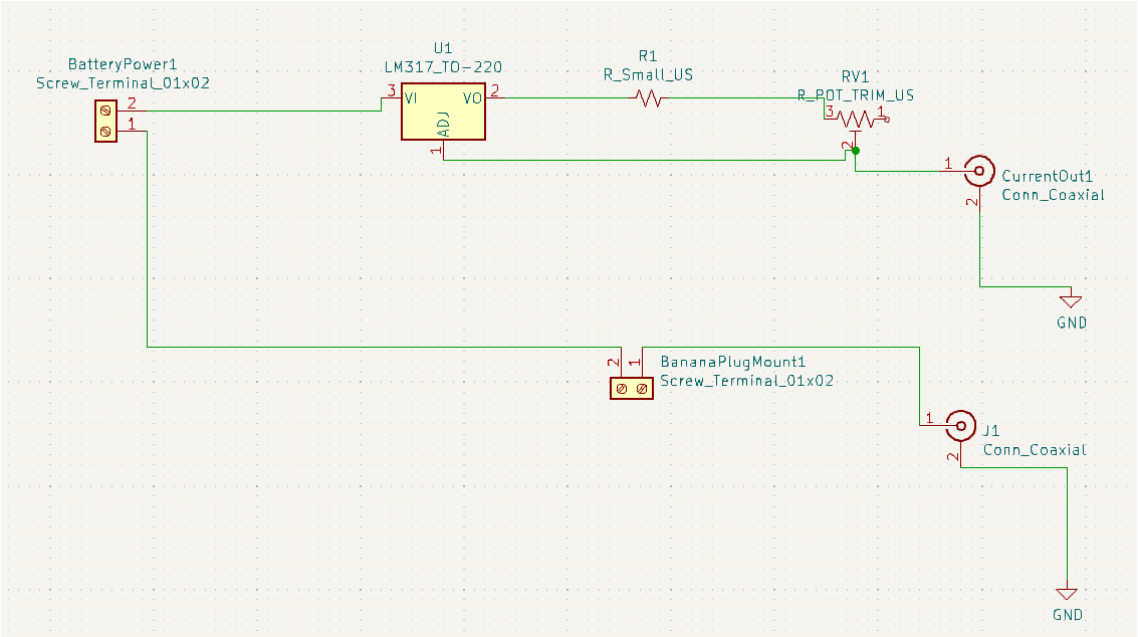


Ilustración 58. Esquemático 1

El diseño de la PCB de este primer circuito se puede observar en la siguiente Ilustración 59.

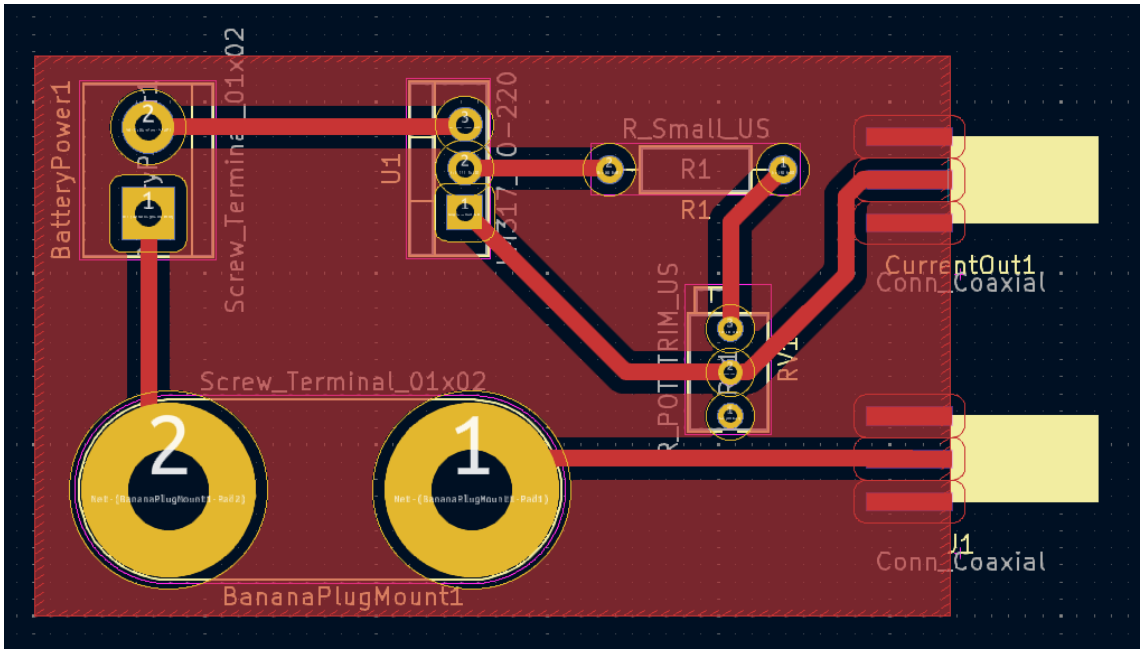


Ilustración 59. Diseño PCB de prueba 1

Una vez obtenidos los ficheros de fabricación gerber, el archivo que contiene los agujeros (extensión de tipo .drl) no necesita ser procesado, en cambio el fichero que contiene la información de la capa superior debe ser procesado por el software de *FlatCAM*, con el objetivo de obtener el contorno de las pistas en formato *GCODE*. En las Ilustraciones 60, 61 y 62 se puede apreciar el proceso de generación de los ficheros Gcode que serán procesados y enviados desde la interfaz.

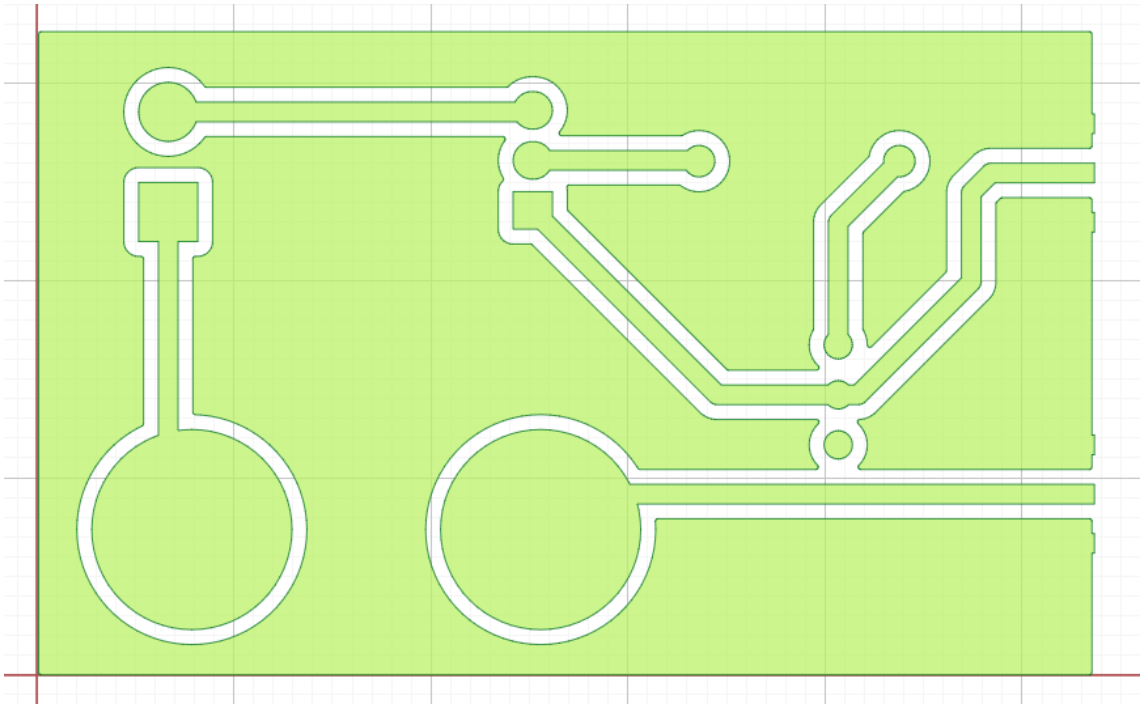


Ilustración 60. Circuito sin isolación

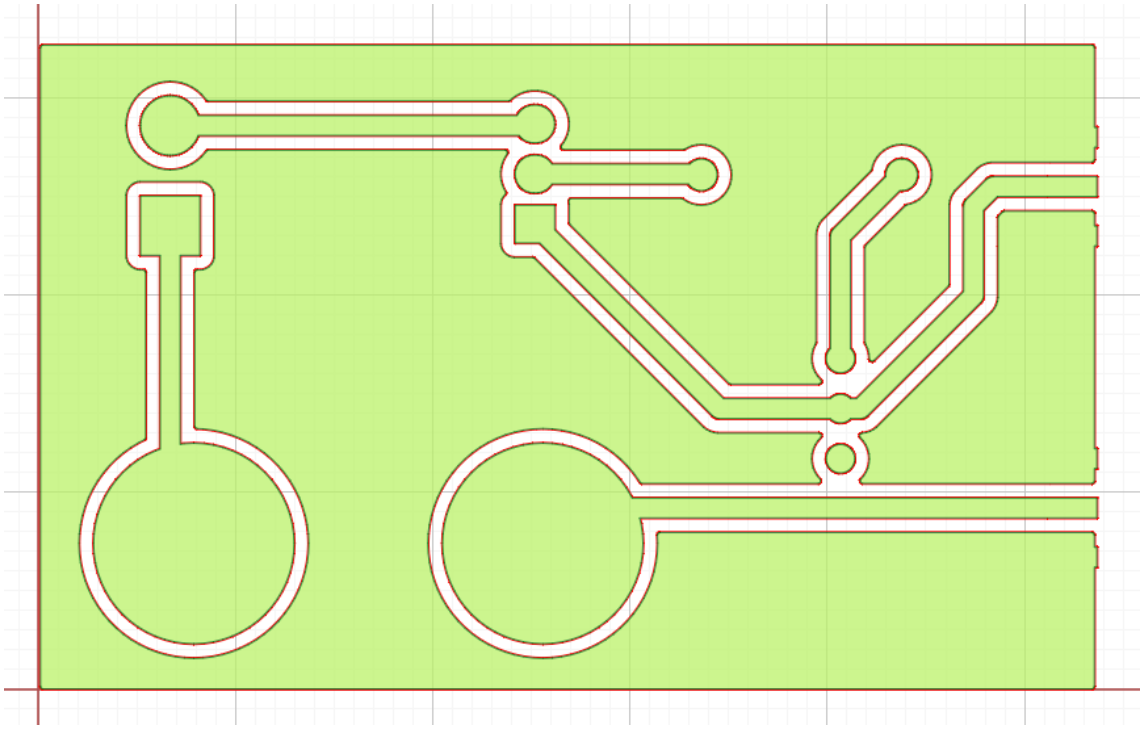


Ilustración 61. Circuito con isolación

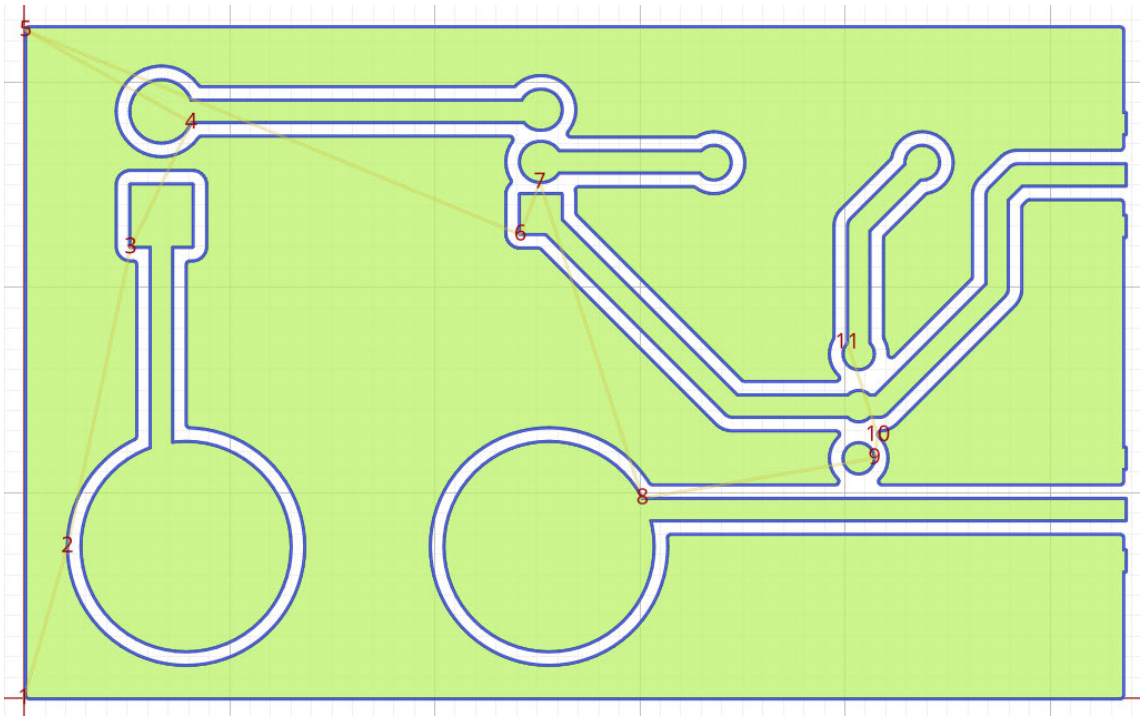


Ilustración 62. Circuito con trabajo CNC generado

Una vez enviado este circuito a la CNC, tras el procesamiento del mismo desde la interfaz se ha obtenido el siguiente resultado, el cual se observa en la Ilustración 63.

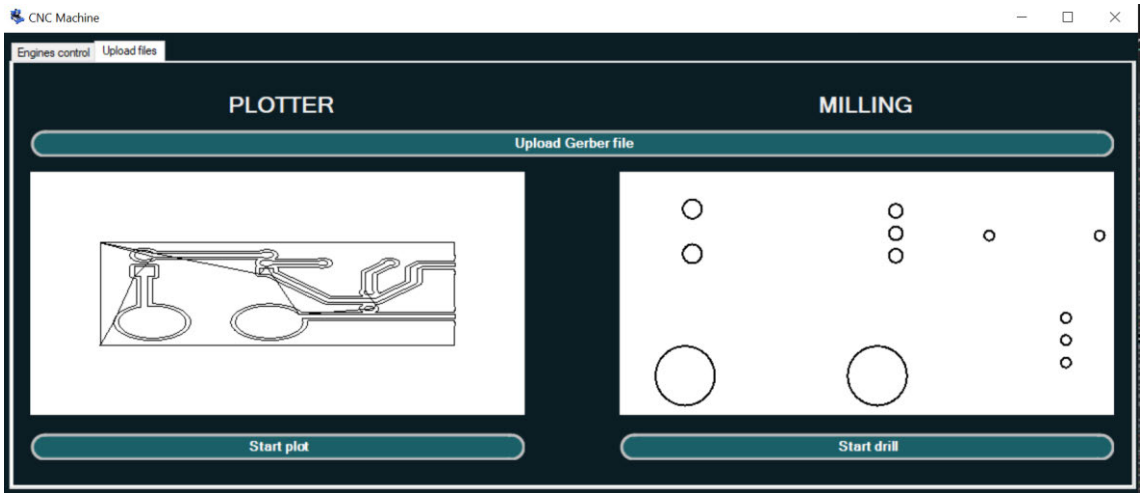


Ilustración 63. Interfaz primer circuito

Por último, tras pulsar el botón “Start plot” el resultado ha sido el siguiente, apreciable en la Ilustración 64.

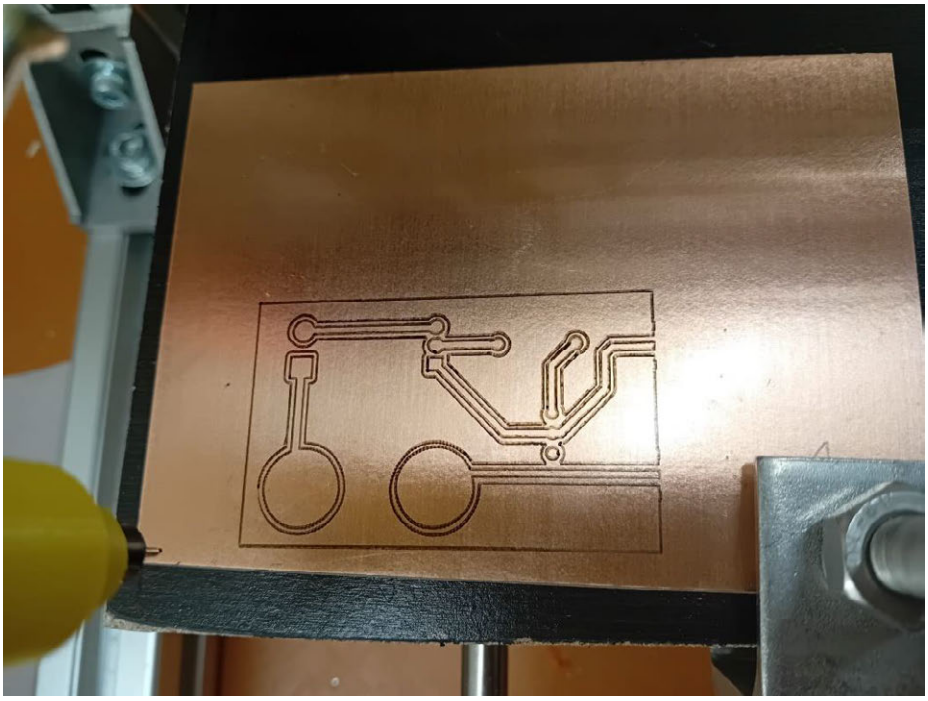


Ilustración 64. Resultados primer circuito

DRIVEN AMPLIFIER

En segundo lugar, se ha empleado el siguiente circuito [B5] para poder corroborar de nuevo la precisión de la máquina, el diseño ha sido realizado mediante la herramienta de *KiCad*. En la *Ilustración 65* se puede observar el esquemático del circuito.

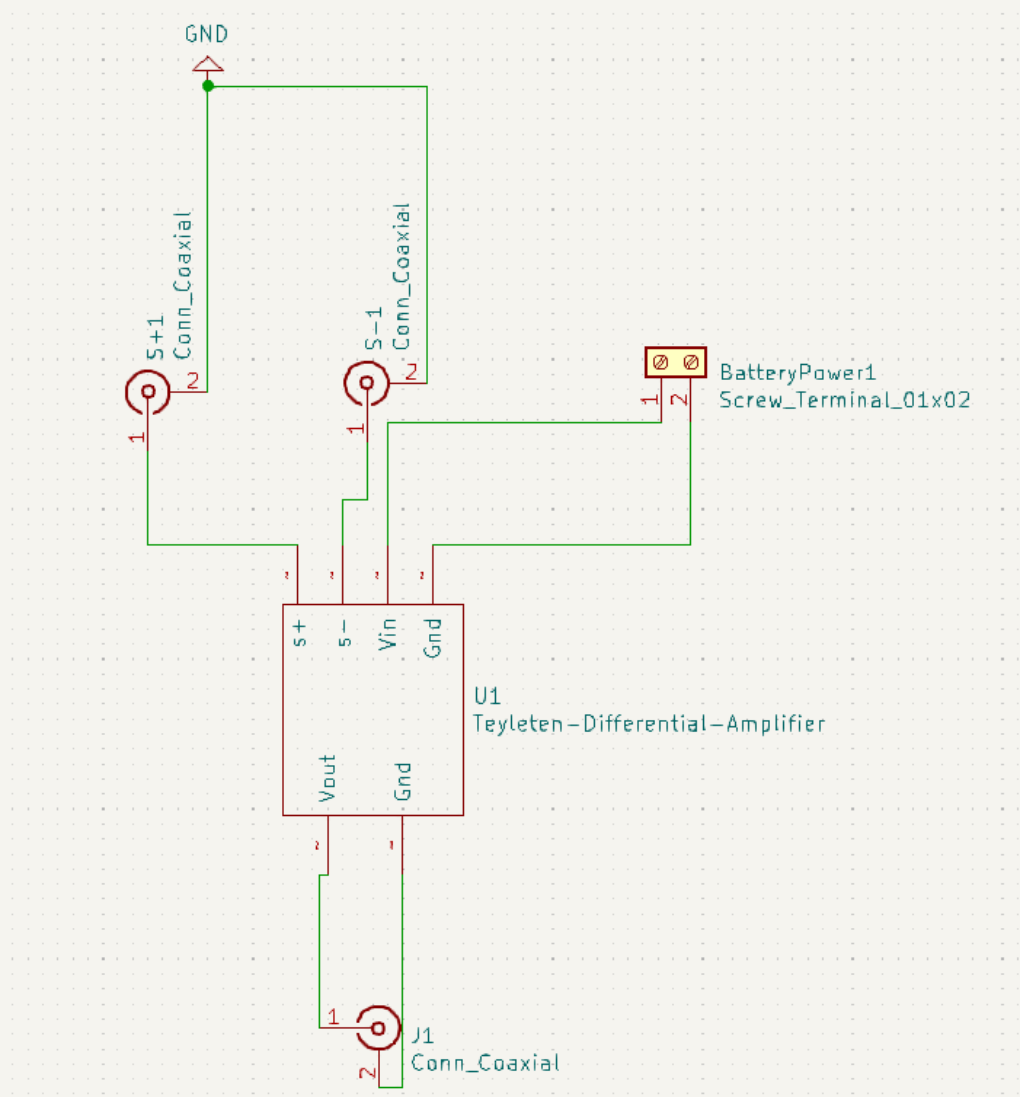


Ilustración 65. Esquemático 2

El diseño de la PCB de este segundo circuito se puede observar en la siguiente Ilustración 66.

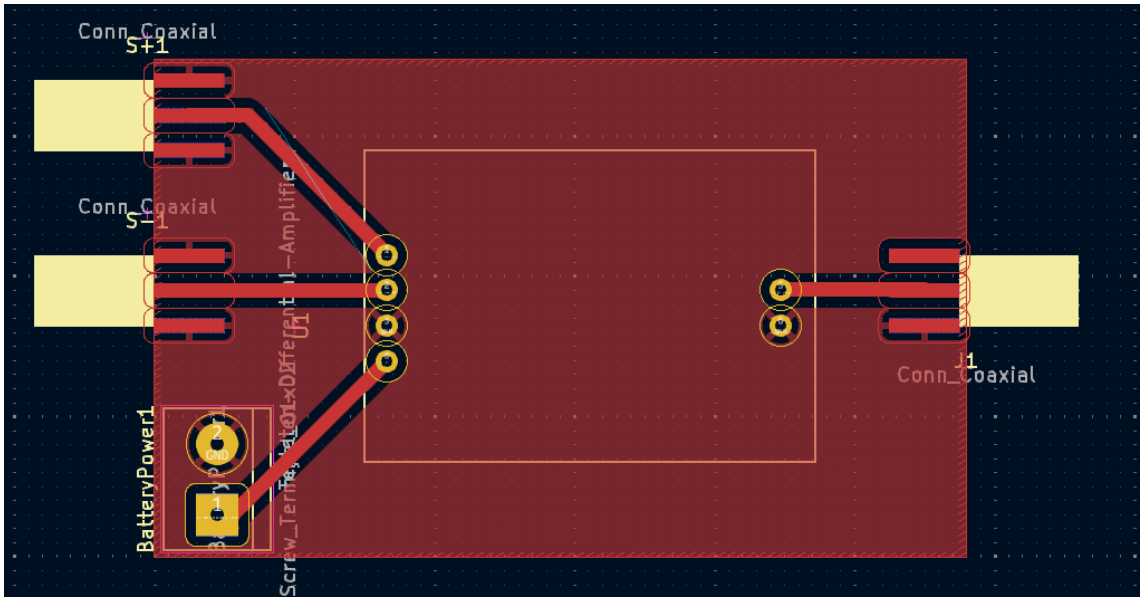


Ilustración 66. Diseño PCB de prueba 2

De nuevo, el fichero gerber debe de ser procesado mediante el software de "FlatCAM". En las ilustraciones 67, 68 y 69 se puede apreciar el proceso de generación de los ficheros Gcode que serán procesados y enviados desde la interfaz.

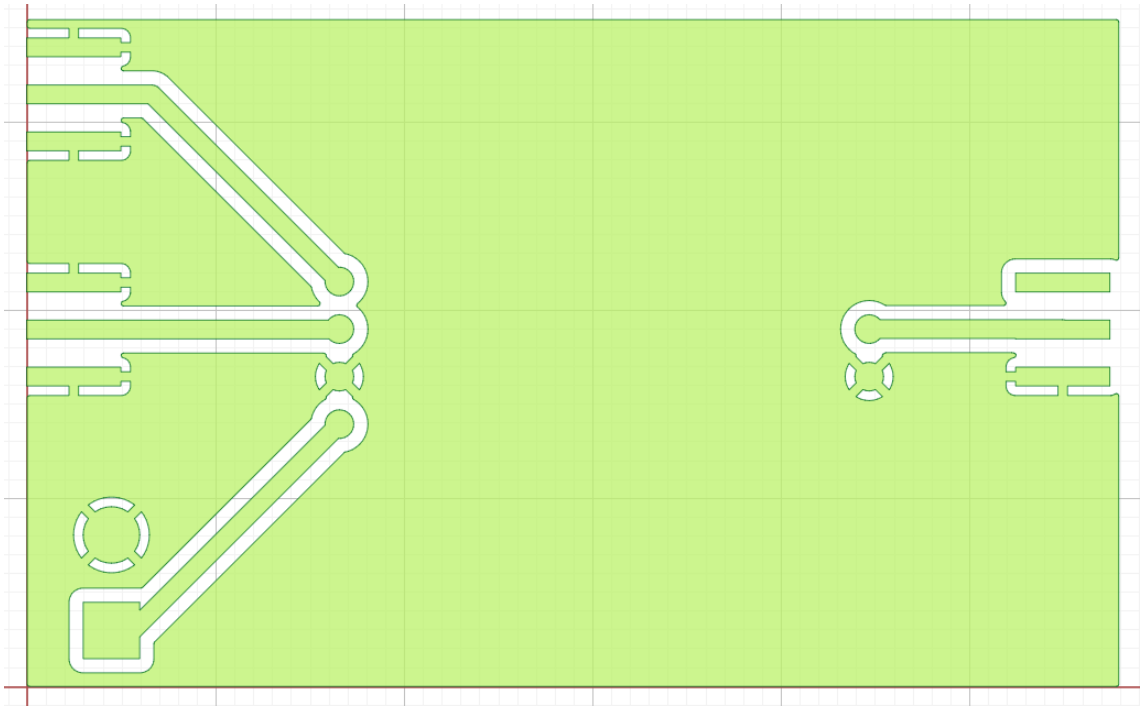


Ilustración 67. Circuito 2 sin isolación

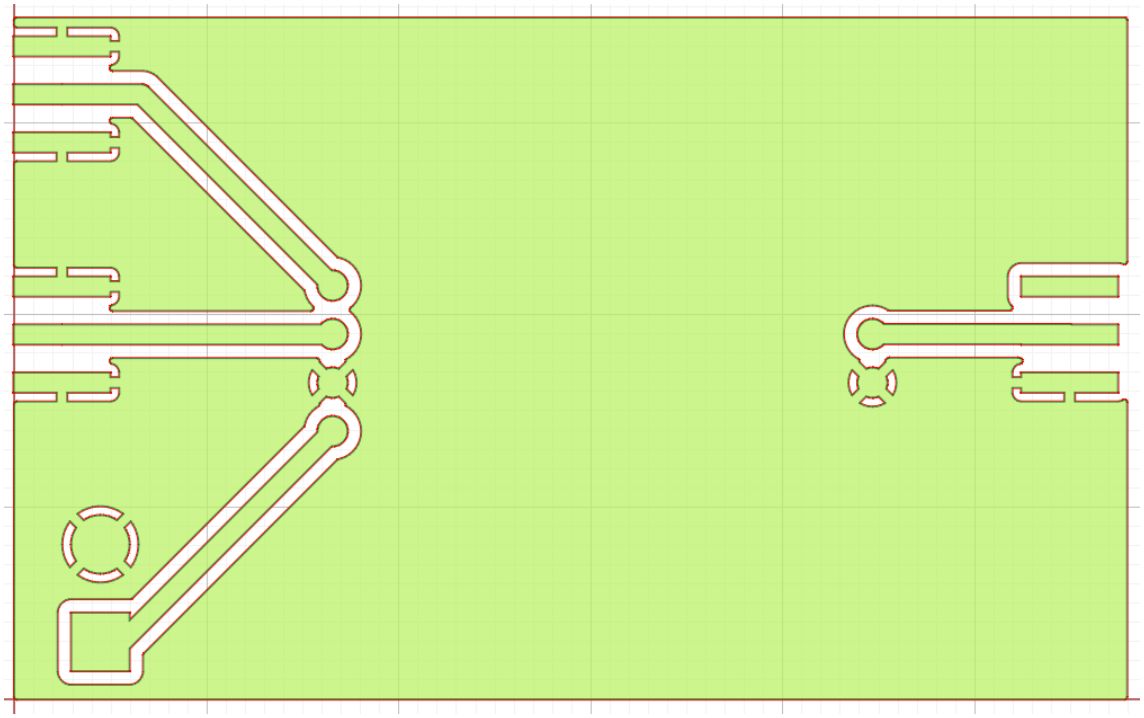


Ilustración 68. Circuito 2 con isolación

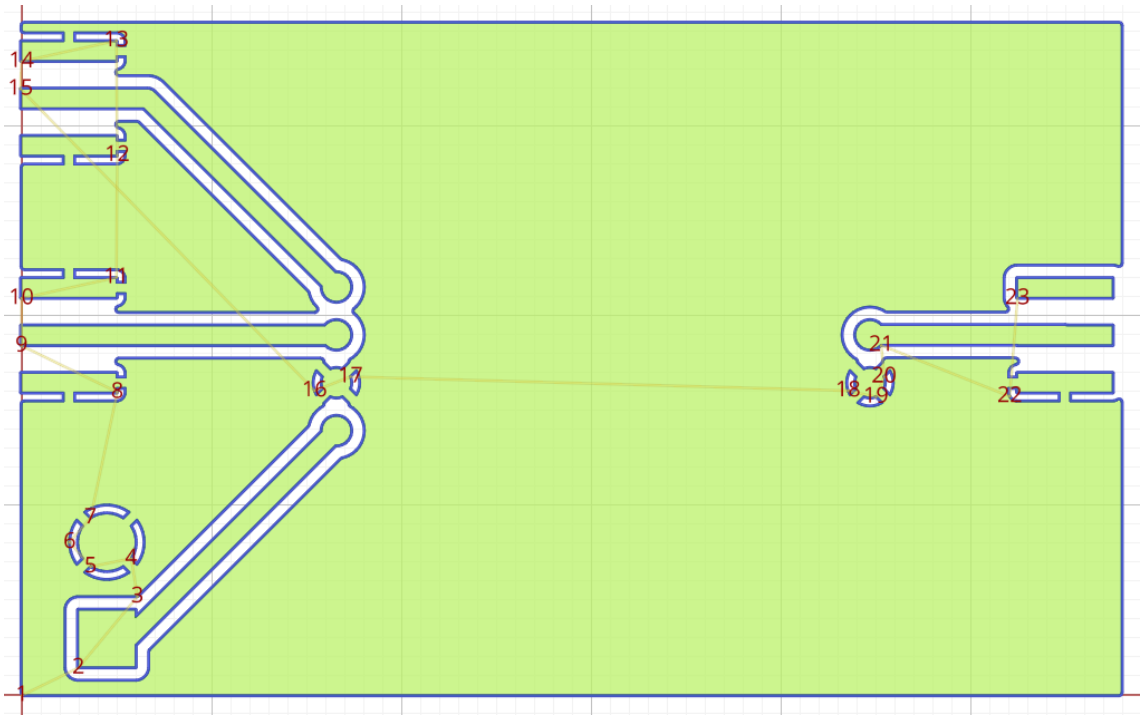


Ilustración 69. Circuito 2 con trabajo CNC generado

Una vez enviado este circuito a la CNC, tras el procesamiento del mismo desde la interfaz se ha obtenido el siguiente resultado, el cual se observa en la ilustración 70.

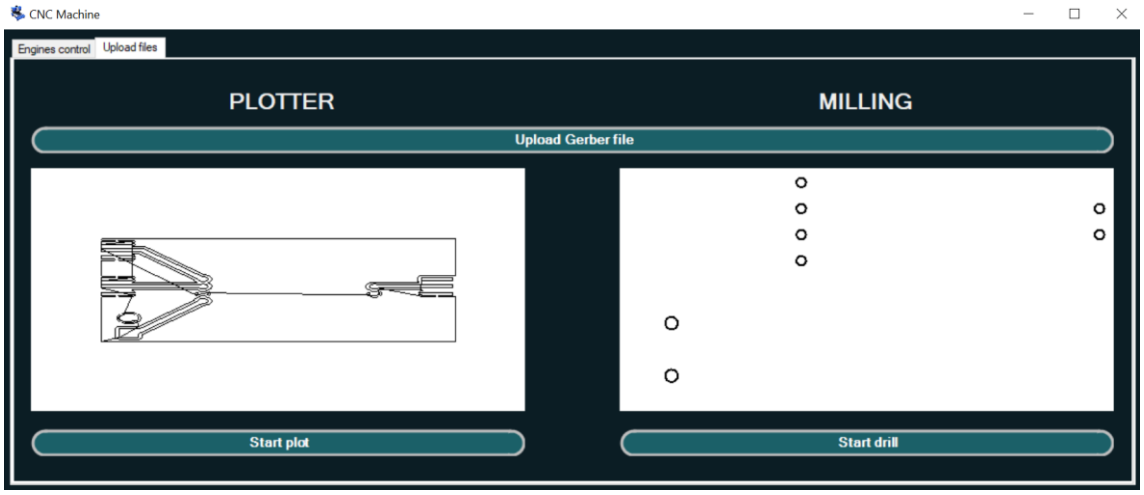


Ilustración 70. Interfaz segundo circuito

Por último, tras pulsar el botón "Start plot" el resultado ha sido el siguiente, apreciable en la Ilustración 71.

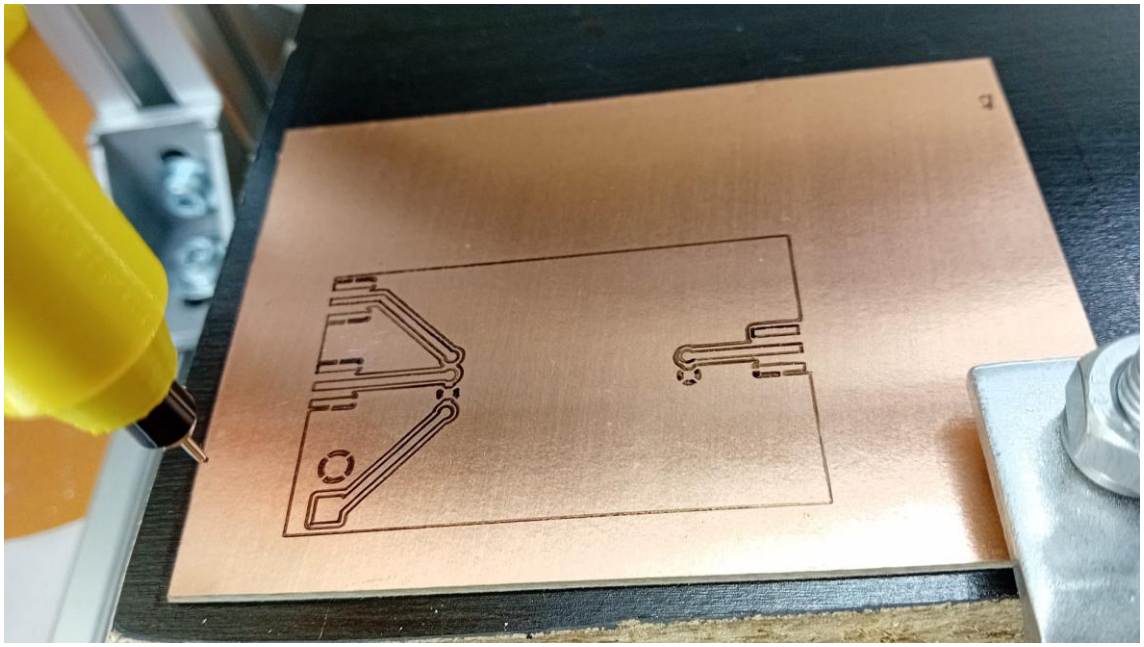


Ilustración 71. Resultados segundo circuito

KELVIN PROBE

En tercer lugar, se ha empleado el siguiente circuito [B5] para poder corroborar de nuevo la precisión de la máquina, el diseño ha sido realizado mediante la herramienta de *KiCad*. En la *Ilustración 72* se puede observar el esquemático del circuito.

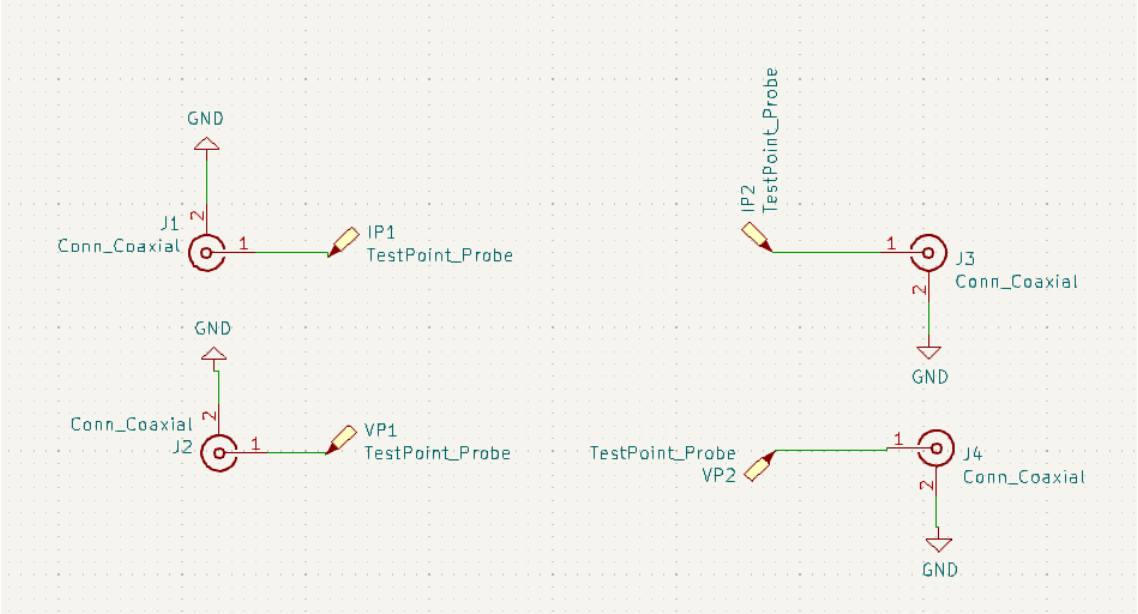


Ilustración 72. Esquemático 3

El diseño de la PCB de este segundo circuito se puede observar en la siguiente Ilustración 73.

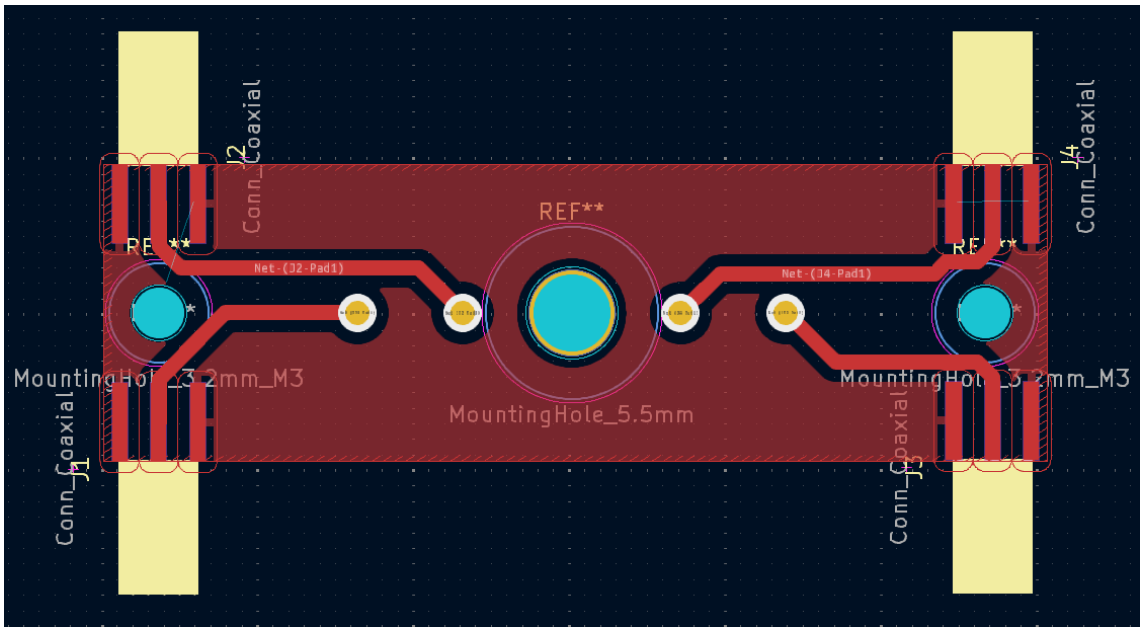


Ilustración 73. Diseño PCB de prueba 3

De nuevo, el fichero gerber debe de ser procesado mediante el software de "FlatCAM". En las ilustraciones 74, 75 y 76 se puede apreciar el proceso de generación de los ficheros Gcode que serán procesados y enviados desde la interfaz.

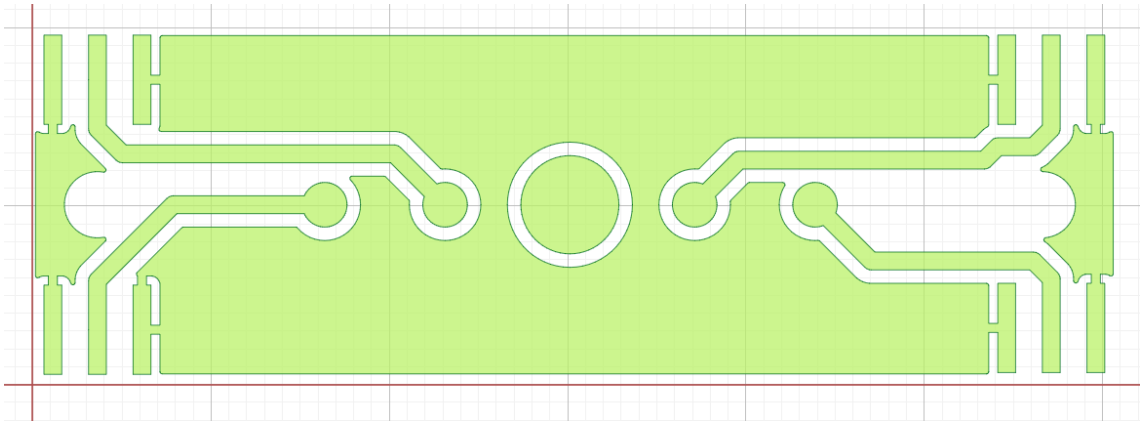


Ilustración 74. Circuito 3 sin isolación

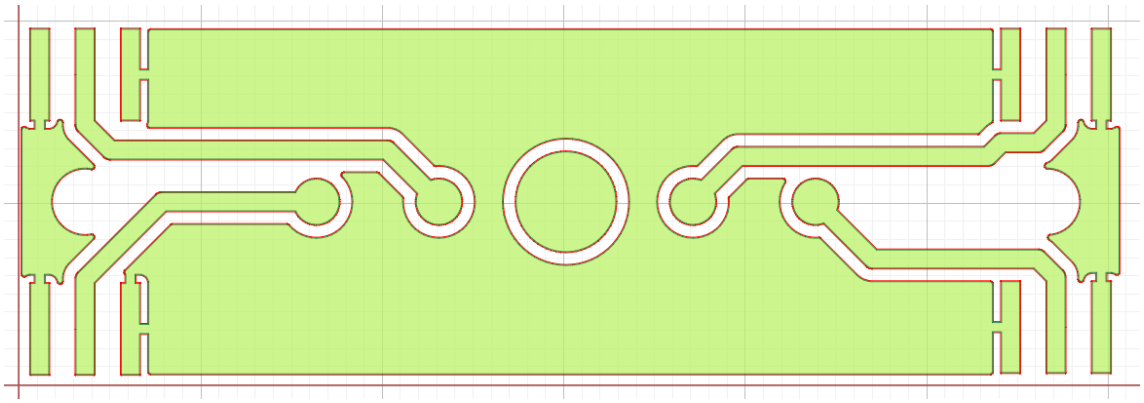


Ilustración 75. Circuito 3 con isolación

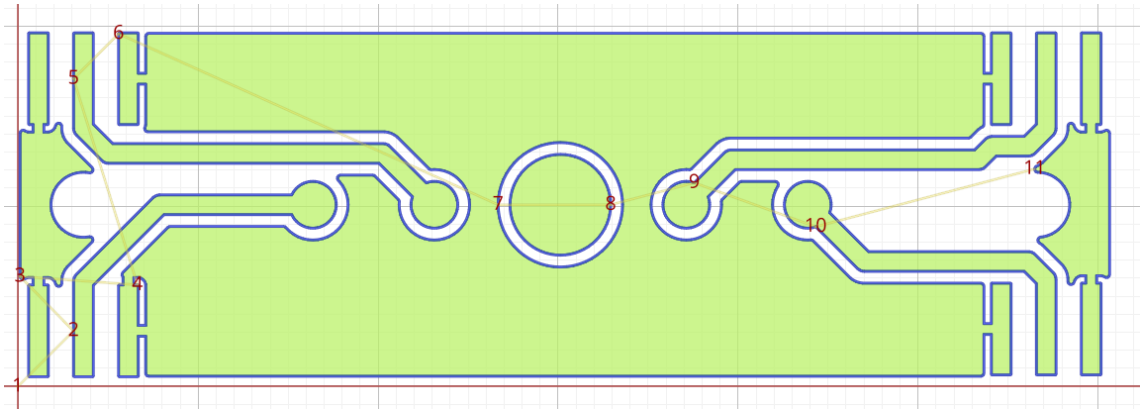


Ilustración 76. Circuito 3 con trabajo CNC generado

Una vez enviado este circuito a la CNC, tras el procesamiento del mismo desde la interfaz se ha obtenido el siguiente resultado, el cual se observa en la ilustración 77.

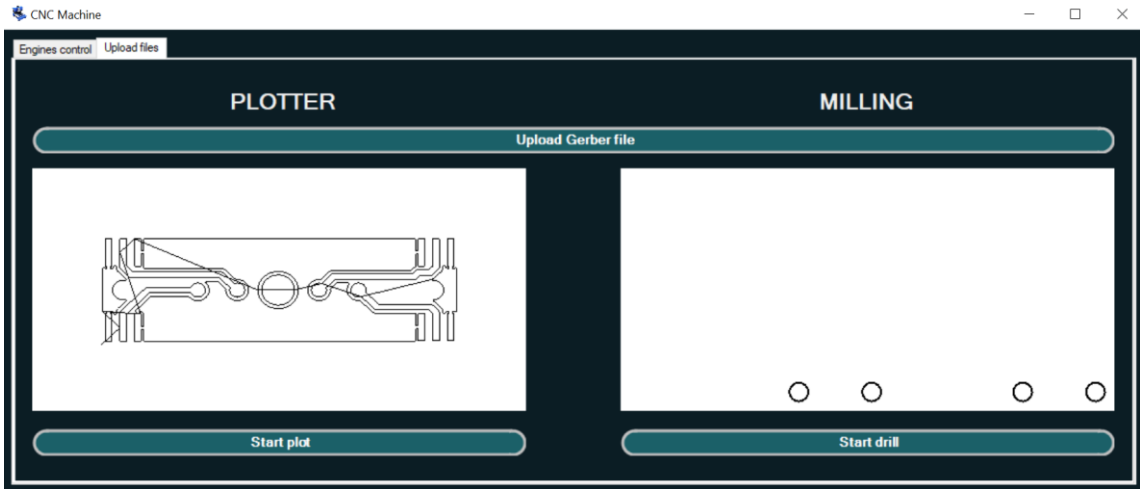


Ilustración 77. Interfaz tercer circuito

Por último, tras pulsar el botón “Start plot” el resultado ha sido el siguiente, apreciable en la Ilustración 78.

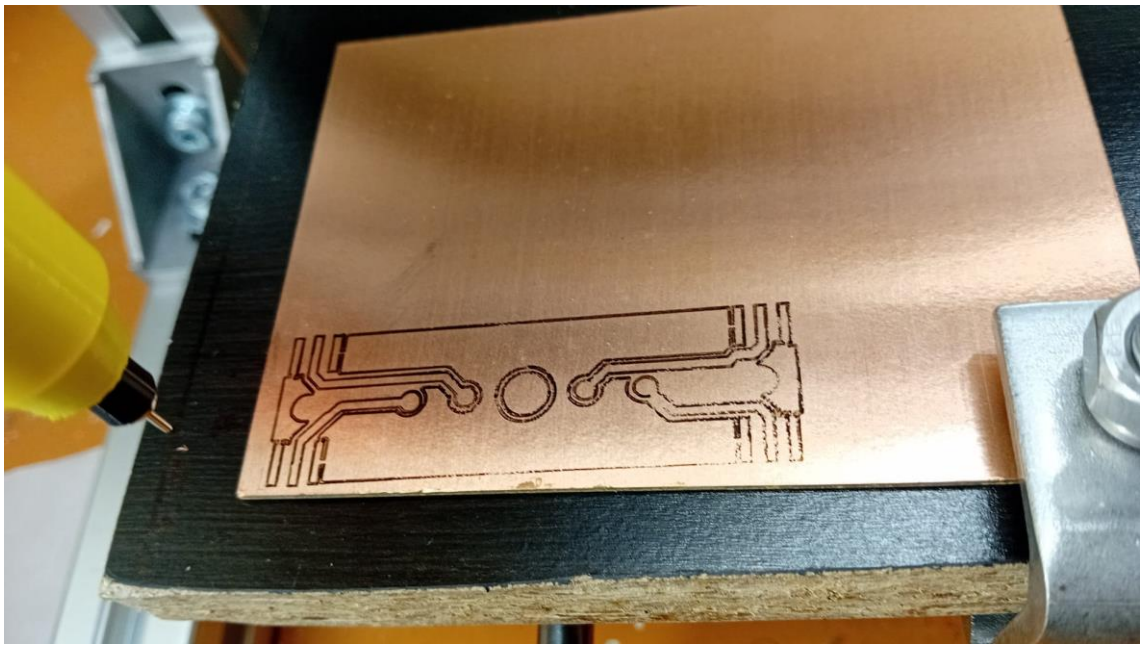


Ilustración 78. Resultados tercer circuito

Por último, se ha realizado una prueba de taladrado de uno de los agujeros de los circuitos, coincidiendo con el tamaño de broca disponible en el taladro, el resultado se aprecia en la Ilustración 79.

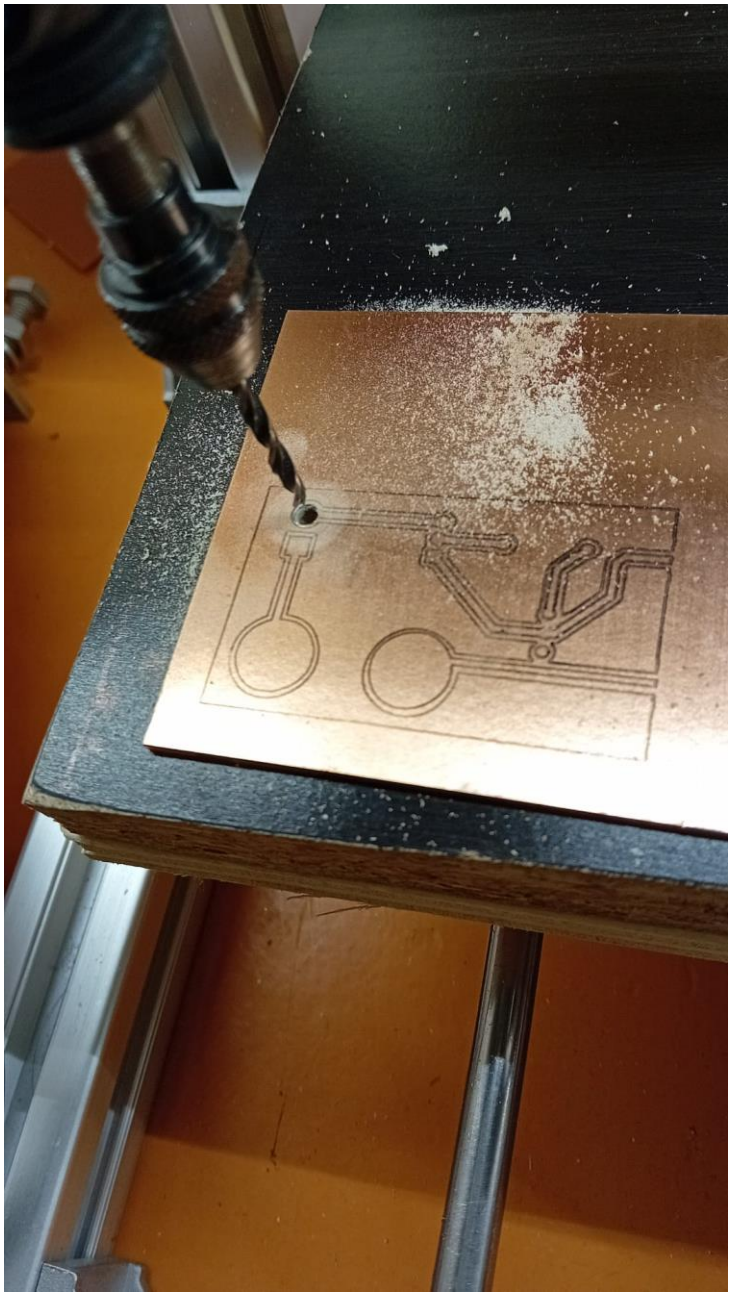


Ilustración 79. Prueba de taladrado

5.1. Pruebas de la CNC

Durante el transcurso del siguiente apartado, se adjuntarán en formato tabla a modo resumen el conjunto de pruebas realizadas a la CNC. La Tabla 2 agrupa los resultados:

Descripción de la prueba	Resultado
Alimentación	Satisfactoria
Conexión con la interfaz.	Satisfactoria
Movimiento de los motores	Satisfactoria
Calibración de movimiento correcta (Desplazamientos lineales satisfactorios).	Satisfactoria
Funcionamiento de los finales de carrera (Parada inmediata del motor correspondiente para evitar daños en el equipo).	Satisfactoria
Vibraciones durante el movimiento mínimas.	Satisfactoria
Precisión y fiabilidad.	Satisfactoria
Robustez de la máquina.	Satisfactoria

Tabla 2. Pruebas de la CNC

5.2. Pruebas de la interfaz de control

Durante el transcurso del siguiente apartado, se adjuntarán en formato tabla a modo resumen el conjunto de pruebas realizadas a la interfaz de control. La Tabla 3 agrupa los resultados:

Descripción de la prueba	Resultado
Prueba de funcionamiento de cada uno de los botones que integran la interfaz.	Satisfactoria
`Prueba de funcionamiento de los PictureBox (representan el trabajo que realizará la CNC).	Satisfactoria
Prueba de lectura de fichero comprimido .zip y extracción de los ficheros con extensión .drl y .gcode.	Satisfactoria
Traducción de ficheros .drl y .gcode a .gcode con comandos personalizados de la CNC.	Satisfactoria
Prueba de estrés (Pulsación consecutiva del mismo botón).	Satisfactoria
Apertura y cierre del puerto serie.	Satisfactoria
Envío de comandos de acción por puerto serie (Por ejemplo, “auto_home”).	Satisfactoria
Envío de comandos GCODE por puerto serie.	Satisfactoria
Espera de recepción de ACKs.	Satisfactoria

Tabla 3. Pruebas interfaz de control.

6. Presupuesto

Este apartado mostrará en detalle el presupuesto final del proyecto, teniendo en cuenta todos los materiales y componentes empleados. Se ha dado el caso de que el alumno disponía ya de ciertos materiales previos al inicio del proyecto, los cuales han sido incluidos en el presupuesto de igual manera para concretar la totalidad del gasto necesario para el montaje de la solución. En la Tabla 4 se puede observar el desglose del coste de todos los materiales empleados.

Descripción	Cantidad	Precio (€)	Precio total (€)
Kit CNC (3x Motor NEMA 17, 3x Final de carrera, 3x Kit cable y soporte NEMA 17, 3x Drivers DRV8825, 1x Arduino UNO, 1x CNC Shield v3)	1	59.99	59.99
Kit de montaje motor Eje Z	1	54.20	54.20
Interruptor universal (1 pin ON-OFF)	1	3.50	3.50
Fuente de alimentación 12V 10A	1	15.99	15.99
Acoplamiento eje 5mm a 8mm	2	1.80	3.60
Soporte KP08 de 8mm	2	2.49	4.98
Soporte SK08 de 8mm	4	1.49	5.96
Kit fresadora eléctrica	1	39.99	39.99
Husillo de longitud 400mm + tuerca	2	5.99	11.98
Tornillo de plomo T8 antiretroceso	1	3.48	3.48
Varilla lisa de 8mm con longitud de 400mm	2	3.15	6.30
Perfil de aluminio 3030 de 400mm	2	4.12	8.24
Perfil de aluminio 3030 de 340mm	5	3.50	17.50
Perfil de aluminio 3030 de 270mm	2	2.78	5.56
Escuadra para perfiles de aluminio 3030	18	0.90	16.20
Tornillo M5 para escuadra de aluminio 3030	54	0.22	11.88
Arandela M5	60	0.01	0.60
Tuerca deslizante tipo T para perfil de aluminio 3030	54	0.19	10.26
Rodamiento lineal SC8UU	4	2.10	8.40
Cable conector de motor NEMA17	1	1.95	1.95
Placa de montaje motor NEMA 17	2	2.07	4.14
Varilla lisa 10mm con longitud de 400mm	2	4.90	9.80
Soporte SK10 de 10mm	4	1.75	7

Gastos de ferretería (Tornillería y herramienta básica)	1	10	10
Gasto de cableado y componentes varios	1	5	5
Gasto de filamento 3D PLA 1.75mm	1	8	8
Madera contrachapada 1m x 1m	1	3	3
Kit brocas varios tamaños	1	7	7
Total			344.5

Tabla 4. Presupuesto de materiales

A este presupuesto, se le debe añadir el costo de la mano obra estimada por tiempo de montaje necesario para el montaje final de la máquina.

En la Tabla 5 se puede observar el presupuesto de la mano de obra.

Tarea	**Nº horas**	**€/hora**	**Costo (€)**
Montaje de la estructura	5	2	10
Instalación de finales de carrera en soportes personalizados	1	2	2
Instalación Arduino + CNC Shield	0.5	2	1
Instalación de motores	1	2	2
Instalación de base	2	2	4
Instalación eje Z	0.5	2	1
Total			20

Tabla 5. Presupuesto de mano de obra

En la Tabla 6 se puede apreciar el conjunto del presupuesto total, teniendo en cuenta tanto los materiales como la mano de obra empleada.

Presupuesto	**Costo (€)**
Materiales	344.5
Mano de obra	20 €

Tabla 6. Presupuesto total

Con estos datos, se puede afirmar que el presupuesto final para todo el desarrollo y fabricación de este proyecto ha sido de **364.5 €.**

7. Impacto del proyecto

Durante el transcurso de este apartado, se detallará el impacto de este proyecto dependiendo del tipo de implicación presente:

Implicaciones Sociales

Las implicaciones sociales relativas a este proyecto son las siguientes:

- **Acceso a la tecnología**. El desarrollo de esta CNC podría democratizar el acceso a la fabricación digital, permitiendo tanto a usuarios como a comunidades con recursos limitados crear prototipos y productor personalizados.

- **Empoderamiento de la comunidad**. El hecho de compartir conocimientos sobre la fabricación y desarrollo de una CNC podría empoderar a la comunidad de usuarios hacia el desarrollo de habilidades técnicas con el objetivo de emprender proyectos de fabricación locales.

- **Educación y formación**. La implementación de este tipo de tecnología en entornos educativos podría mejorar la enseñanza de temas relacionados con la ingeniería, la programación y la fabricación, preparando de esta manera a los estudiantes ante futuras oportunidades laborales, además de lograr un mayor grado de motivación hacia las tecnologías.

Implicaciones de Salud y Seguridad

Las implicaciones de salud y seguridad relativas a este proyecto son las siguientes:

- **Prevención de accidentes**. Al emplear una tecnología como la de la CNC con el objetivo de realizar taladros, se minimiza la interacción humana, evitando así posibles accidentes durante el manejo de herramientas eléctricas y/o de cualquier otro tipo.

- **Ergonomía**. La configuración adecuada de la máquina y la estación de trabajo (Interfaz de control) podría contribuir a la prevención de lesiones por esfuerzo repetitivo, además de mejorar las condiciones de trabajo de los operarios.

Implicaciones ambientales

Las implicaciones ambientales relativas a este proyecto son las siguientes:

- **Fabricación sostenible**. La máquina desarrollada para este proyecto podría utilizarse para fabricar productor personalizados con materiales reciclados o ecológicos, promoviendo de esta manera prácticas de fabricación más sostenibles, reduciendo el desperdicio de materiales.

- **Eficiencia energética**. Tras la optimización del diseño y de los procesos de la CNC, se podría reducir el consumo de energía.

- Reducción de emisiones. El hecho de poder realizar fabricaciones de PCB localmente, evitaría el transporte de productor a larga distancia, posibilitando la reducción de emisiones de gases generadas por los vehículos de transportes.

Además, este proyecto cumple con aportaciones para los Objetivos de Desarrollo Sostenibles (ODS) las cuales son las siguientes:

- **Industria, innovación e infraestructura (ODS 9)**. La implementación de tecnologías como la CNC podría fomentar la innovación en el sector manufacturero y mejorar la infraestructura tecnológica en comunidades locales.

- **Trabajo decente y crecimiento económico (ODS 8)**. Al proporcionar acceso a habilidades técnicas y oportunidades de fabricación, sería posible contribuir al crecimiento económico y a la creación de empleo en áreas donde escasean las oportunidades laborales.

- **Producción y consumo responsables (ODS 12)**. Al promover p´racticas de fabricación sostenible y producción local, sería posible contribuir a reducir el consumo excesivo y fomentar un enfoque más responsable hacia los recursos naturales.

8. Conclusiones

Durante el transcurso de este apartado, se propondrá un vistazo general a las conclusiones del proyecto y sus líneas futuras de desarrollo.

8.1. Conclusiones

Para concluir, se ha logrado construir una máquina robusta y eficiente capaz de realizar el dibujo de las pistas sobre la PCB, además de realizar los taladros pertinentes con el tamaño definido durante el diseño de la propia placa.

Con este trabajo se suple la necesidad de una máquina capaz de realizar el trabajo con mayor precisión y eficacia con la que lo haría un humano, además de lograr este hito en menor tiempo.

El diseño de la máquina ha necesitado el empleo de un software de diseño llamado Sketchup, el cual ha facilitado las labores de diseño gracias a el empleo de componentes ya diseñados por la comunidad. En relación con dicho diseño, los componentes y materiales de la máquina han sido adquiridos teniendo en cuenta el coste y la calidad, permitiendo así el ensamblaje de un producto final de gran calidad con un precio asequible.

Para el control de la máquina, se ha desarrollado una interfaz de usuario en Visual Studio 2022 amigable y de uso sencillo adecuada para su correcto manejo por los usuarios.

La integración de la interfaz con la máquina ha sido desarrollada mediante el empleo de puerto serie, con una comunicación basada en el sistema de Acks, permitiendo así un correcto flujo de información en ambos sentidos sin bloquear el sistema.

Con respecto a las pruebas y resultados finales, se puede concluir que el desarrollo final es satisfactorio, teniendo en cuenta los resultados teóricos esperados y los resultados finales obtenidos por la máquina.

8.2. Líneas futuras

Como líneas futuras del proyecto, se debe tener en cuenta que durante la fase de diseño de este se optó por emplear un eje Z con cabezal intercambiable, permitiendo así el futuro empleo de la máquina para diversas tareas más allá del mecanizado. Entre las posibles tareas que la máquina podría realizar en el futuro destacan las siguientes:

- **Realización de dibujo mediante el empleo de rotulador o cualquier otro tipo de método de dibujo.** Esta tarea la realiza la máquina en este proyecto.

- **Taladrado de agujeros mediante el empleo de taladro / broca.** Esta tarea la realiza la máquina en este proyecto.

- **Fabricación de piezas 3D mediante el empleo de extrusor de plástico convencional.** Únicamente equipando la máquina con un extrusor de plástico,

con un pequeño motor para el filamento y con mínimas modificaciones en el código, esta máquina sería capaz de extruir filamento y realizar fabricación de piezas 3D. (Con las únicas limitaciones de la superficie de la base y la altura de movimiento del eje Z).

- **Grabado láser en madera**. Equipando la máquina con un láser, ajustando la potencia y realizando pequeñas modificaciones en el código, la máquina sería capaz de realizar grabado láser en madera. (Con la única limitación de la superficie de la base).

9. Referencias

[1] Arduino-Home, “Página oficial de Arduino”, online: https://www.arduino.cc/ [Visitado en febrero de 2023]

[2] Steve Jennings, MOTORES PASO A PASO, online: http://revistas.sena.edu.co/index.php/inf_tec/article/view/899 [Visitado en noviembre de 2022].

[3] ¿Qué es G-Code?. Disponible en: https://polaridad.es/que-es-g-code/, [Visitado en noviembre de 2022]

[4] Algoritmo de Bresenham. Disponible en: https://www.glosarioit.com/Algoritmo_de_Bresenham ,[Visitado en mayo de 2023]

[5] AccelStepper, «AccelStepper-Library,» [En línea].:

https://github.com/adafruit/AccelStepper/blob/master/AccelStepper.h. [Visitado en abril de 2023].

[6] gearoid-murphy/circuits. Repo for Kicad / FlatCam / CNC workflows. https://github.com/search?q=kicad+flatcam&type=repositories [Visitado en Marzo de 2024]

Anexo: Integración interfaz/CNC

Código asociado al envío de datos desde la interfaz

La clase '**ArduinoCommunication**' contiene el constructor y las funciones empleadas por la interfaz para el envío de datos al Arduino.

```
public class ArduinoCommunication
{
    private SerialPort serialPort;

    public ArduinoCommunication()

    public void OpenConnection()

    public void SendData(string data)

    public void CloseConnection()
}
```

El constructor '**ArduinoCommunication()**' es el encargado de configurar el puerto serie a utilizar y la velocidad de transmisión del mismo.

```
public ArduinoCommunication()
{
    serialPort = new SerialPort("COM13", 9600);
}
```

La función '**OpenConnection()**' abre el puerto serie configurado para poder empezar a transmitir información.

```
public void OpenConnection()
{
    try
    {
        serialPort.Open();
        Console.WriteLine("Puerto serie abierto.");
    }
    catch (Exception ex)
    {
        Console.WriteLine("Error al abrir el puerto serie: " + ex.Message);
    }
}
```

La función '**SendData()**' envía por el puerto serie configurado la información otorgada como argumento.

```
public void SendData(string data)
{
    if (serialPort.IsOpen)
    {
        try
        {
            serialPort.WriteLine(data);
            Console.WriteLine("Datos enviados al Arduino: " + data);
        }
        catch (Exception ex)
        {
            Console.WriteLine("Error al enviar datos: " + ex.Message);
        }
    }
    else
    {
        Console.WriteLine("El puerto serie no está abierto.");
    }
}
```

La función '**CloseConnection()**' abre el puerto serie configurado para poder empezar a transmitir información.

```
public void CloseConnection()
{
    if (serialPort.IsOpen)
    {
        try
        {
            serialPort.Close();
            Console.WriteLine("Puerto serie cerrado.");
        }
        catch (Exception ex)
        {
            Console.WriteLine("Error al cerrar el puerto serie: " +
ex.Message);
        }
    }
    else
    {
        Console.WriteLine("El puerto serie ya está cerrado.");
    }
}
```

Código asociado a la recepción de datos desde el Arduino

Desde el Arduino, será necesario gestionar la recepción de datos, dichos datos llegan en forma tanto de comandos predefinidos como de líneas Gcode. Gracias a la variable booleana **'running'** el Arduino es capaz de detectar si la máquina envía comandos predefinidos o líneas del tipo Gcode. El siguiente código agrupa la recepción de datos desde el Arduino:

```
while (1) {
    if (Serial.available() > 0 && !running) {  //Comprueba que no se está
ejecutando un trabajo (Plotter/Milling)
      String receivedData = Serial.readStringUntil('\n');
      executeCommand(receivedData);
    } else if (Serial.available() > 0 && running) {
        if (drawingMode) {
          c = Serial.read();
          if ((c == '\n') || (c == '\r')) {  //Final de línea
            if (lineIndex > 0) {
              line[lineIndex] = '\0';        // Ejecutar comando
              processIncomingLine(line, lineIndex);
              lineIndex = 0;
            } else {
            }
            lineIsComment = false;
            lineSemiColon = false;
          } else {
            if ((lineIsComment) || (lineSemiColon)) {  //Control de
comentarios o ;
              if (c == ')') lineIsComment = false;
            } else {
              if (c <= ' ') {
              } else if (c == '/') {
              } else if (c == '(') {
                lineIsComment = true;
              } else if (c == ';') {
                lineSemiColon = true;
              } else if (lineIndex >= LINE_BUFFER_LENGTH - 1) {
                lineIsComment = false;
                lineSemiColon = false;
              } else if (c >= 'a' && c <= 'z') {
                line[lineIndex++] = c - 'a' + 'A';
              } else {
                line[lineIndex++] = c;
              }
            }
          }
        }
    }
```

```
  }
```

Como se puede apreciar, una vez que la máquina está funcionando con líneas Gcode, se llama a la función **'ProcessIncomingLine(char* line, int charBN)',** este método será el encargado de procesar y ejecutar las líneas Gcode recibidas:

```
void processIncomingLine(char* line, int charNB) {
  struct point newPos;
  int currentIndex = 0;
  char buffer[64];

  newPos.x = 0.0;
  newPos.y = 0.0;
  float i = 0.0;
  float j = 0.0;

  //  Comandos
  //  G1 Movimiento
  //  G1 X60 Y30 Ejemplo
  //  G2 X60 Y30 I10 J-4 Movimiento circular en sentido horario, con
centro relativo en (X+I, Y+J)
  //  M300 S30 (Z AXIS DOWN)
  //  M300 S50 (Z AXIS UP)

  while (currentIndex < charNB) {
    switch (line[currentIndex++]) {
      case 'U':
        autohomeZ();
        break;
      case 'D':
        autohomeZ();
        turnNegativeZ(80);
        break;
      case 'G':
        buffer[0] = line[currentIndex++];
        buffer[1] = '\0';

        switch (atoi(buffer)) {
          case 1:
            {
              char* indexX = strchr(line + currentIndex, 'X');
              char* indexY = strchr(line + currentIndex, 'Y');

              if (indexX != NULL) {
                newPos.x = atof(indexX + 1);
              } else {
                newPos.x = actuatorPos.x;
              }
```

```
            if (indexY != NULL) {
              newPos.y = atof(indexY + 1);
            } else {
              newPos.y = actuatorPos.y;
            }

            drawLine(newPos.x, newPos.y, false);  //Pinta una linea
hacia la nueva dirección
            actuatorPos.x = newPos.x;
            actuatorPos.y = newPos.y;
            Serial.println("ACK1");
          }
          break;
        case 2:
          {
            char* indexX = strchr(line + currentIndex, 'X');
            char* indexY = strchr(line + currentIndex, 'Y');
            char* indexI = strchr(line + currentIndex, 'I');
            char* indexJ = strchr(line + currentIndex, 'J');

            if (indexX != NULL) {
              newPos.x = atof(indexX + 1);
            } else {
              newPos.x = actuatorPos.x;
            }

            if (indexY != NULL) {
              newPos.y = atof(indexY + 1);
            } else {
              newPos.y = actuatorPos.y;
            }

            if (indexI != NULL) {
              i = atof(indexI + 1);
            } else {
              i = 0.0;
            }

            if (indexJ != NULL) {
              j = atof(indexJ + 1);
            } else {
              j = 0.0;
            }

            float centerX = actuatorPos.x + i;
            float centerY = actuatorPos.y + j;
```

```
              float radius = sqrt((actuatorPos.x - centerX) *
(actuatorPos.x - centerX) + (actuatorPos.y - centerY) * (actuatorPos.y -
centerY));

              // Dibuja el círculo en sentido horario (true)
              drawArc(actuatorPos.x, actuatorPos.y, newPos.x, newPos.y,
centerX, centerY, radius, true);
              actuatorPos.x = newPos.x;
              actuatorPos.y = newPos.y;
              Serial.println("ACK2");
            }
            break;
          case 3:
            {
              char* indexX = strchr(line + currentIndex, 'X');
              char* indexY = strchr(line + currentIndex, 'Y');
              char* indexI = strchr(line + currentIndex, 'I');
              char* indexJ = strchr(line + currentIndex, 'J');

              if (indexX != NULL) {
                newPos.x = atof(indexX + 1);
              } else {
                newPos.x = actuatorPos.x;
              }

              if (indexY != NULL) {
                newPos.y = atof(indexY + 1);
              } else {
                newPos.y = actuatorPos.y;
              }

              if (indexI != NULL) {
                i = atof(indexI + 1);
              } else {
                i = 0.0;
              }

              if (indexJ != NULL) {
                j = atof(indexJ + 1);
              } else {
                j = 0.0;
              }

              float centerX = actuatorPos.x + i;
              float centerY = actuatorPos.y + j;
```

```
              float radius = sqrt((actuatorPos.x - centerX) *
(actuatorPos.x - centerX) + (actuatorPos.y - centerY) * (actuatorPos.y -
centerY));

              // Dibuja el círculo en sentido antihorario (false)
              drawArc(Xpos, Ypos, newPos.x, newPos.y, centerX, centerY,
radius, false);
              actuatorPos.x = newPos.x;
              actuatorPos.y = newPos.y;
              Serial.println("ACK3");
            }
          break;
          case 4:
          {
            running = false;
            drawingMode = false;
          }
        }
      break;
      case 'M':  //Z AXIS
        buffer[0] = line[currentIndex++];
        buffer[1] = line[currentIndex++];
        buffer[2] = line[currentIndex++];
        buffer[3] = '\0';
        switch (atoi(buffer)) {
          case 300:
            {  //Interpreta los M300 S50 (Z AXIS UP) y M300 S30 (Z AXIS
DOWN)
              char* indexS = strchr(line + currentIndex, 'S');
              float Spos = atof(indexS + 1);
              if (Spos == 30) {
                turnNegativeZ(5250);
                Serial.println("ACK4");
              }
              if (Spos == 50) {
                turnPositiveZ(5250);
                Serial.println("ACK5");
              }
            }
            break;
          default:
            break;
        }
        break;
      default:
        break;
    }
  }
```

```
}
```

Manual de usuario

Durante el transcurso de este apartado, se explicará el detalle el Manual de usuario de la CNC, detallando el flujo de la aplicación de escritorio desarrollada para su control y monitorización. Los pasos a seguir para lograr el funcionamiento de la solución son los siguientes:

1. Ejecutar la aplicación, haciendo doble clic sobre el icono de la misma, o en su defecto, un único clic si la aplicación está anclada a la barra de herramientas de Windows. En la Ilustración 80 se puede observar el aspecto del icono de la aplicación, el cual ha sido generado mediante un generar de iconos basado en IA.

Ilustración 80. Icono de la aplicación de escritorio

2. Una vez abierta la aplicación, el aspecto será el siguiente, visible en la Ilustración 81. En esta primera pantalla, se podrán observar numerosos botones de control de movimiento de cada uno de los motores.

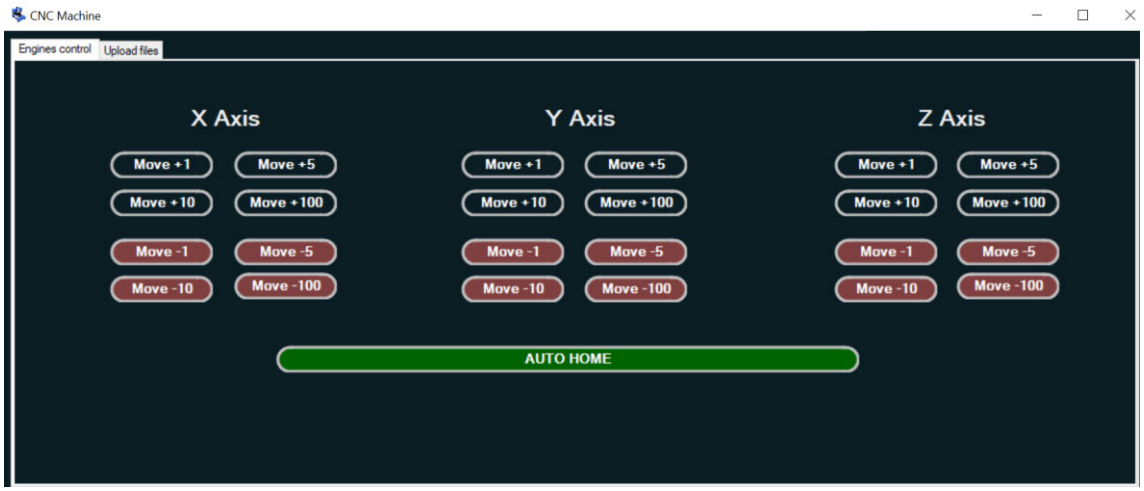


Ilustración 81. Pantalla principal de la aplicación

3. Con el objetivo de comprobar el correcto funcionamiento de la CNC además de la conexión entre la interfaz y la máquina, será altamente recomendable pulsar el botón “AUTO HOME”, esto hará que la máquina realice un auto calibrado inicial, comprobando así el movimiento de los motores. En la Ilustración 82 se puede observar el botón que debe de ser pulsado por el usuario para tal fin.

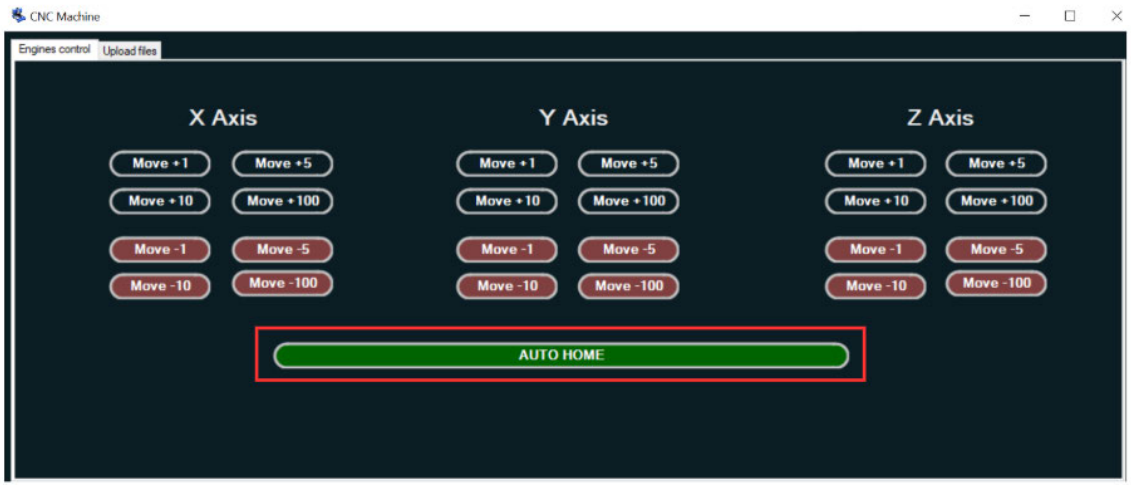


Ilustración 82. Botón "auto home"

4. Una vez completado el auto calibrado, el siguiente paso será acceder a la pantalla de procesamiento y envío de datos, para ello, será necesario pulsar el botón “Upload Files”, presente en la parte superior de la interfaz, como se puede ver en la Ilustración 83.

Ilustración 83. Botón "Upload Files"

5. Tras la pulsación de dicho botón, se accederá a la siguiente pantalla, la cual se puede observar en la Ilustración 84.

Ilustración 84. Pantalla de procesamiento de datos

6. El primer paso en esta pantalla será seleccionar los ficheros que serán procesados, para ello se pulsará el botón "Upload Gerber File", el cual abrirá el Gestor de ficheros de Windows para su búsqueda y selección. En la Ilustración 85 se puede apreciar el botón que deberá ser pulsado para tal fin.

Ilustración 85. Botón "Upload Gerber Files"

7. Tras la pulsación de dicho botón y de la apertura del gestor de ficheros, se deberá seleccionar el archivo en formato .zip[1] el cuál contendrá los ficheros de dibujo y de taladro previamente generados mediante KiCad y FlatCAM. En la Ilustración 86 se puede observar un ejemplo de selección de fichero.

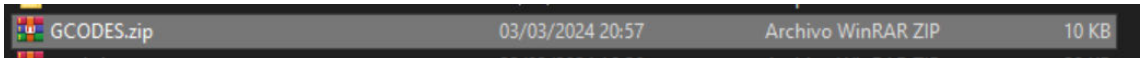


Ilustración 86. Ejemplo de fichero

8. Al seleccionar el fichero deseado, si contiene un archivo de taladros correcto (Extensión .drl), aparecerá el siguiente mensaje, el cual se puede observar en la Ilustración 87. Tras pulsar en este momento en el botón de aceptar, se procesará la información que contiene dicho fichero y la interfaz mostrará una aproximación del layout.

[1] Será totalmente necesario el empleo de ficheros en formato .zip.

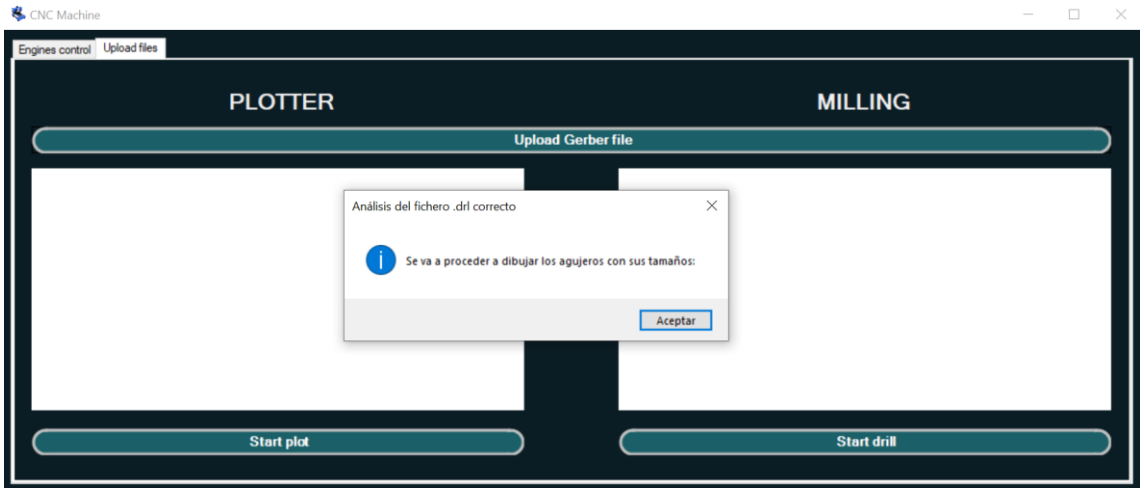


Ilustración 87. Mensaje de análisis correcto

9. Además de la aproximación en dibujo de los agujeros, realizará una segunda aproximación del recorrido que realizará la CNC durante el trabajo de dibujo, en la Ilustración 88 se puede observar el resultado.

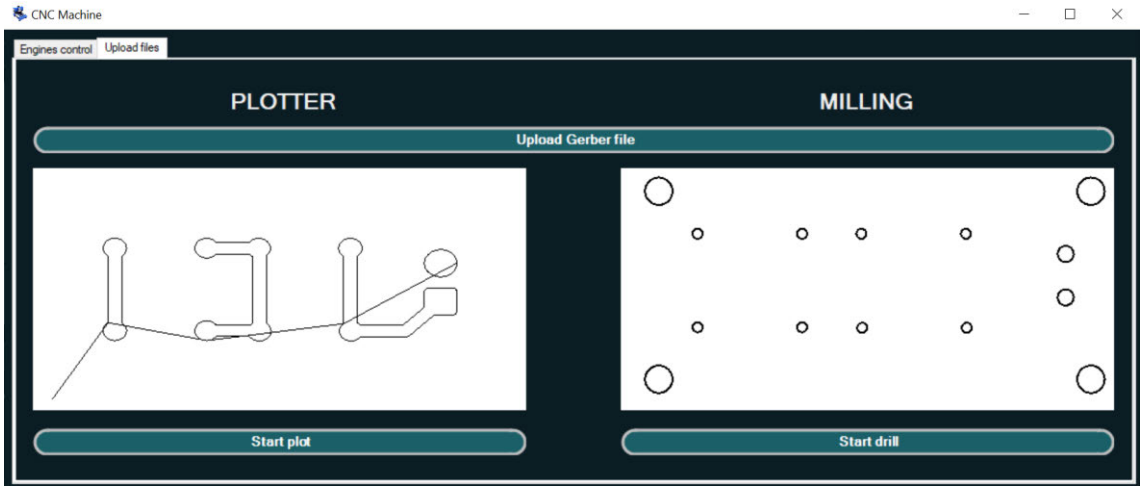


Ilustración 88. Resultados de dibujo de la interfaz de control

10. Llegados a este punto, el siguiente paso será pulsar los botones de "Start plot" y de "Start Drill", los cuales comenzarán los procesos de dibujo[2] y taladrado[3] respectivamente. En la Ilustración 89 se puede observar el aspecto de uno de estos mensajes.

[2] Será necesario tener colocado el rotulador previamente a la pulsación de este botón.
[3] La interfaz indicará al usuario qué tamaño de broca debe de instalar antes de realizar cualquier tipo de taladro.

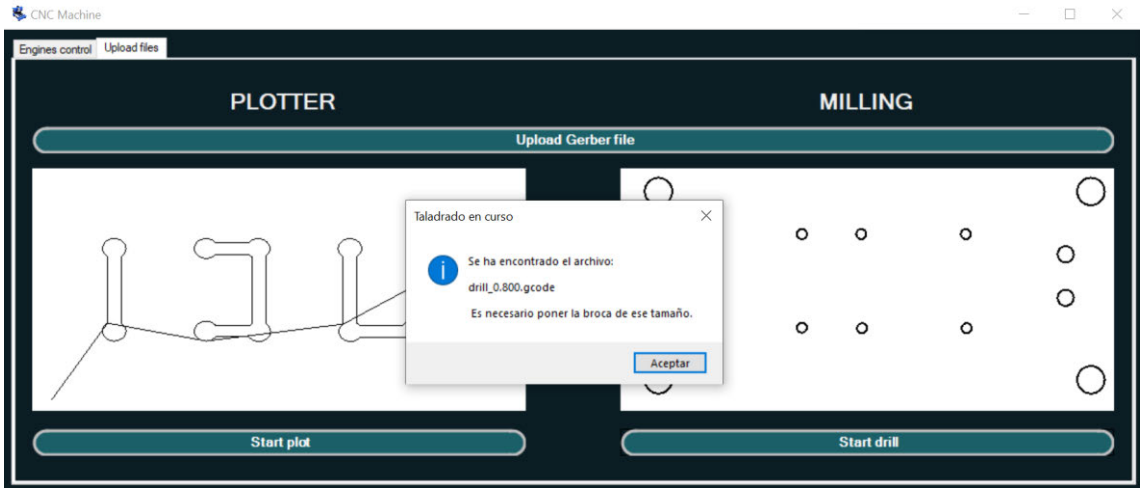


Ilustración 89. Mensaje de tipo taladro a emplear