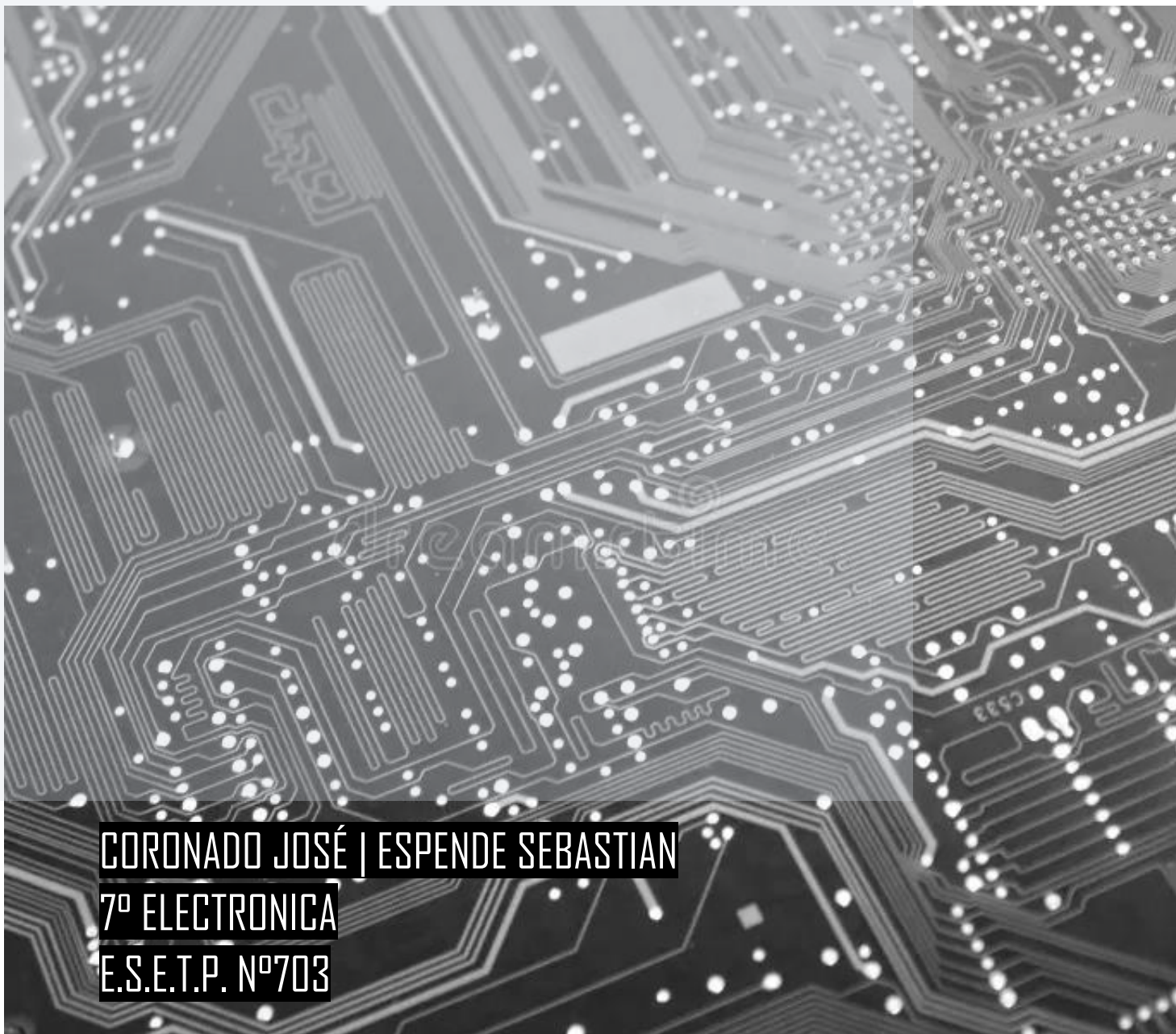


+

PROYECTO FINAL

CNC PERFORADORA DE PCB



CORONADO JOSÉ | ESPENDE SEBASTIAN
7º ELECTRONICA
E.S.E.T.P. N°703

ÍNDICE

CONTENIDO

índice.....	0
INTRODUCCIÓN.....	1
MARCO TEORICO	3
DISEÑO Y DESARROLLO	6
SOFTWARE ESP-32.....	17
CONCLUSIÓN.....	24
BIBLIOGRAFIA	25

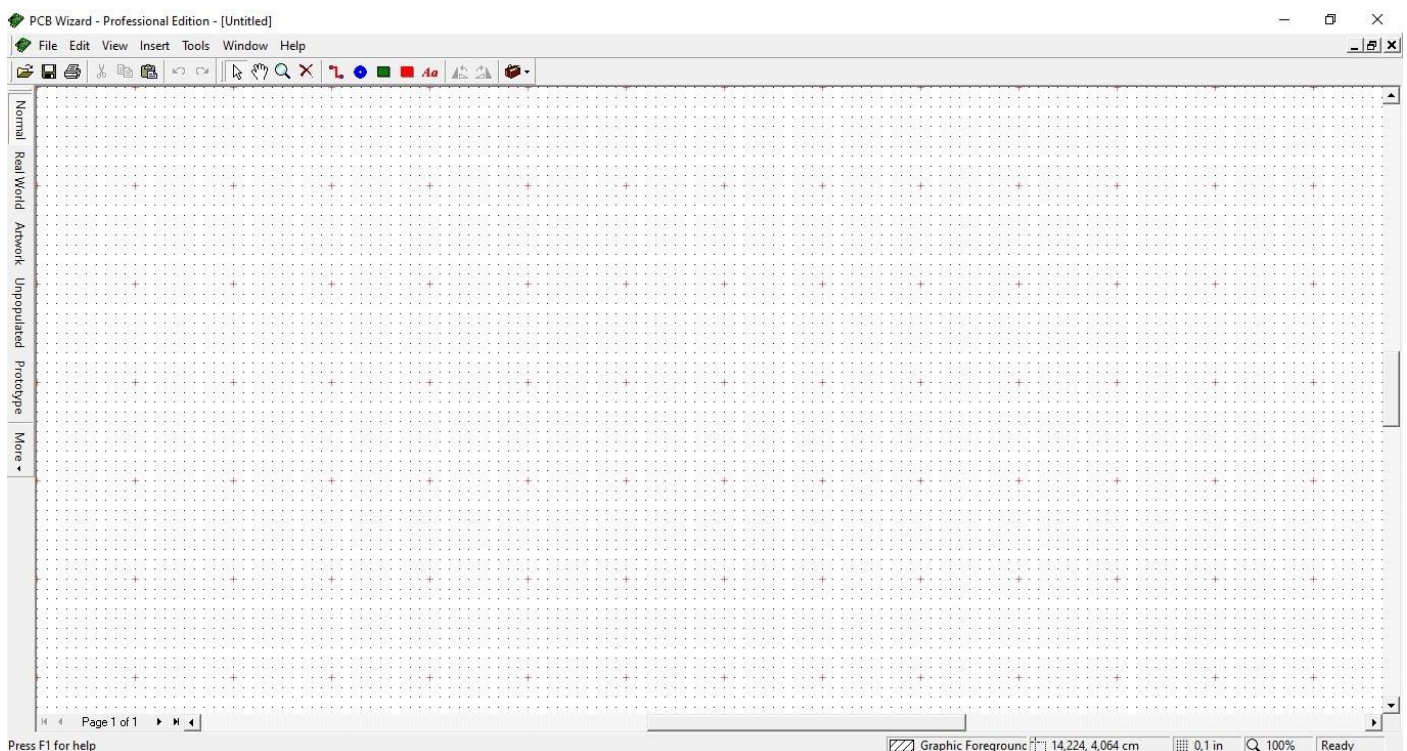
INTRODUCCIÓN

Este proyecto fue pensado y realizado con el fin de diseñar y construir un enrutador CNC de bajo costo, su principal función es agujerear placas PCB de producción caseras por estudiantes. Con este dispositivo se busca facilitar el proceso de la creación de proyectos que requieran placas electrónicas, permitiendo al usuario una mayor precisión y eficiencia en la perforación de placas, en comparación con los métodos manuales. Este proyecto también tiene como objetivo fomentar el aprendizaje práctico de tecnologías CNC, de la electrónica y programación, con el fin de que sea una herramienta útil para los proyectos de electrónica y robótica.

Las placas PCB (**Printed Circuit Board** o circuito impreso) en nuestro ámbito estudiantil son muy importantes, ya que podemos definirlas como la culminación de un proyecto que hayamos realizado primeramente en una placa de pruebas o bien un diseño de un circuito que queramos armar. Las placas están conformadas por una o más capas, en el caso de los estudiantes acostumbramos a usar el tipo de placa que está conformado por un material aislante que puede ser una resina y una capa fina de cobre.

Cuando se realizan placas electrónicas de producción “caseras” se pueden emplear distintos métodos para la realización de pistas, que serán la conexión entre los componentes electrónicos, una de ellas es el uso de marcadores indelebles para realizar el trazo y recorrido de las pistas. Otro método es el uso de tóner, el cual primeramente debemos diseñar el circuito en algún programa como lo puede ser PCB Wizard o KiCad.

PCB WIZARD:

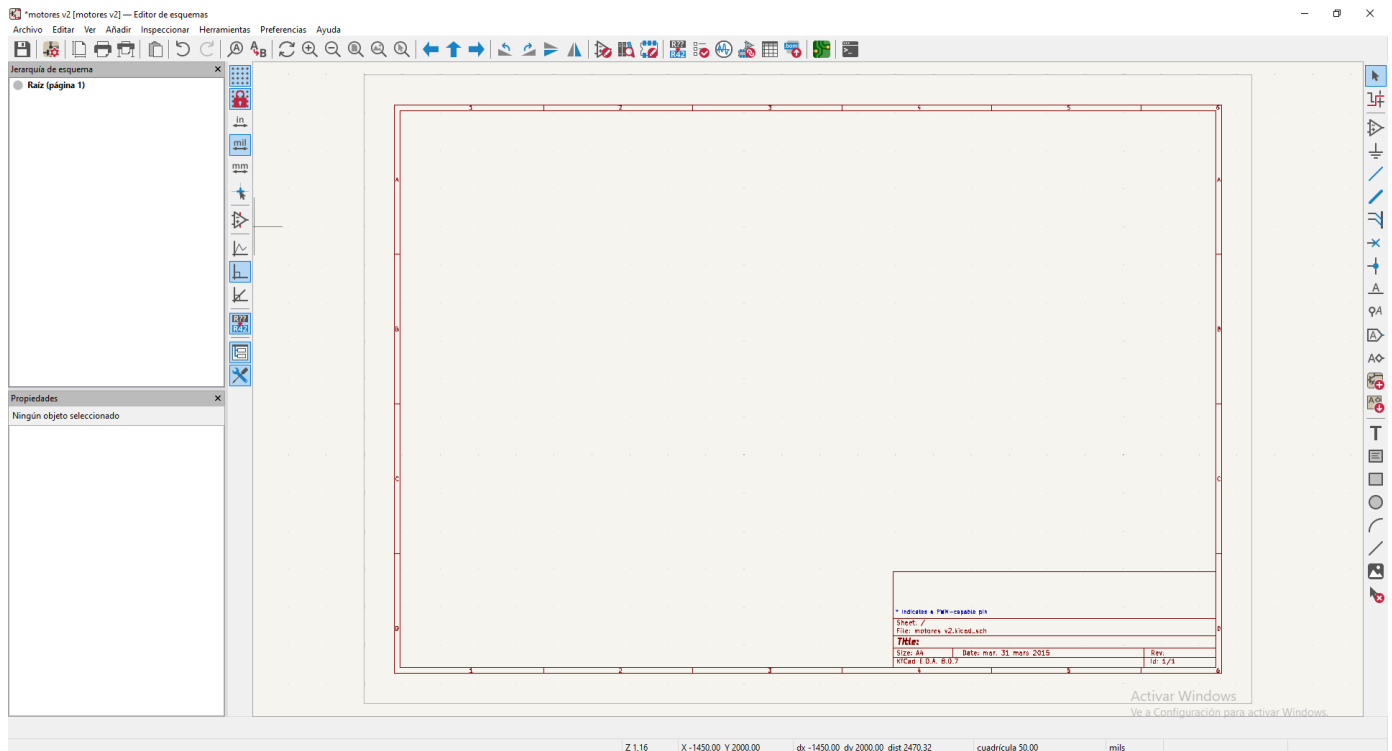


PCB Wizard es un software de diseño de circuitos impresos que destaca por su facilidad de uso y sus capacidades versátiles. Este programa tiene una interfaz sencilla y ofrece herramientas para diseñar un PCB estándar. Incluye funcionalidades como el dibujo esquemático, la conversión a circuito impreso y el posicionamiento de componentes electrónicos. Además, cuenta con una opción de autoruteado que simplifica el proceso de diseño.

Entre las características adicionales, PCB Wizard permite generar reportes de materiales, lo cual es crucial para llevar un control detallado de los componentes necesarios para el ensamblaje del circuito. También ofrece la posibilidad de realizar simulaciones

para verificar el funcionamiento del diseño, con el programa Livewire, antes de la fabricación. La integración con otras herramientas de software y la capacidad de exportar archivos en distintos formatos hacen de PCB Wizard una opción atractiva tanto para estudiantes como para profesionales que buscan un software de diseño de PCBs eficiente y accesible.

KiCad:



Este programa tiene una interfaz sencilla y ofrece herramientas avanzadas para el diseño de PCBs. Incluye dibujo esquemático, conversión a circuito impreso, posicionamiento de componentes, autoruteado y gestión de bibliotecas. Además, permite la visualización 3D del diseño y la generación de archivos Gerber para la fabricación de las placas. La integración con otras herramientas de software y la capacidad de exportar archivos en distintos formatos hacen de KiCad una opción atractiva tanto para estudiantes como para profesionales que buscan un software de diseño de PCBs eficiente y accesible.

Luego del proceso de fabricación de PCBs se requiere perforar la placa para poder realizar el montaje de los componentes electrónicos, tales como: componentes pasivos o componentes activos.

Para la realización de las perforaciones se puede usar taladros de mano o de banco, lo cual puede aumentar el error humano por las siguientes razones:

1. Precisión:

- Al perforar manualmente puede haber error en la alineación de la placa, debido a que no se utilizan guías o plantillas para mantener la estabilización de la placa. Esto puede representar un error de un 5% a 10% en la posición de los orificios.

2. Daños en la PCB:

- El error humano puede provocar daños en las pistas de cobre o en las islas de la placa, y este aumenta si se usa un taladro de mano, ya que se perfora en un ángulo incorrecto o se aplica demasiada fuerza. Esto puede representarse con un 3% a 5% de error en los agujeros.

3. Profundidad de perforación:

- La profundidad de los agujeros puede llegar a ser diferentes entre cada perforación, esto afecta al momento de colocar los componentes, ya que en algunos casos puede que estos no logren ingresar fácilmente por el orificio. Esto representa un 2% a 4% de error humano.

4. Tiempo y eficiencia:

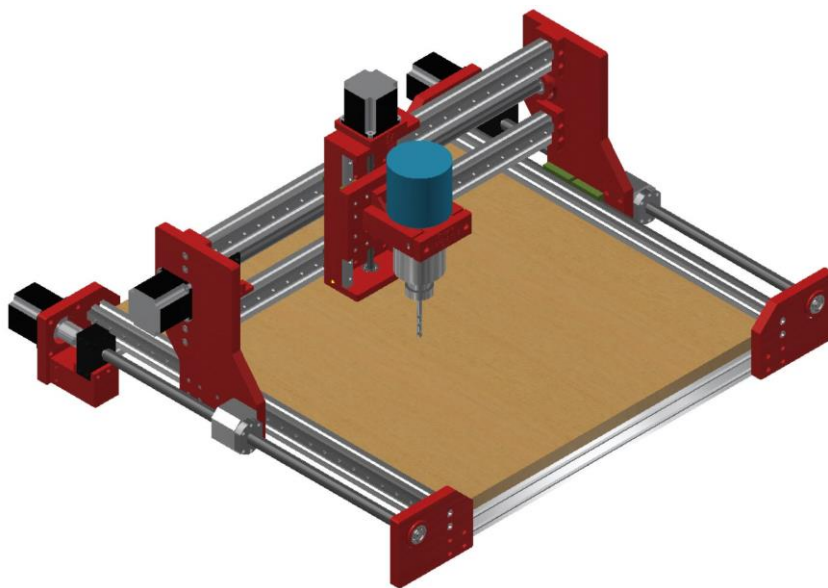
- El tiempo requerido para perforar manualmente una PCB es mayor al que con una CNC, y el cansancio puede aumentar la probabilidad de errores.

El proyecto de la CNC busca reducir este margen de error humano, ya que el mismo posee un soporte/guía para mantener la placa estable, las perforaciones siempre se realizan a un ángulo recto de 90°, la fuerza aplicada es siempre la misma, la profundidad de igual manera siempre es la misma y el tiempo que emplea es mucho menor.

MARCO TEORICO

Una CNC (Control Numérico Computarizado) es un dispositivo que se controla por computadora con el fin de cortar, tallar, grabar o perforar materiales con alta precisión. En el caso de este proyecto, el enrutador CNC está diseñado específicamente para la realización de perforaciones en placas PCB.

Una máquina CNC puede tener diferentes diseños y estar hecha de diversos materiales, pero hay componentes que son indispensables en todas ellas. Estos incluyen motores paso a paso, guías lineales y herramientas como perforadores o cortadores, dependiendo del uso que se les quiera dar.



Esta CNC está compuesta por varios elementos esenciales que trabajan en conjunto para garantizar su funcionamiento. Entre ellos, se encuentran los motores paso a paso, que permiten un movimiento preciso en cada eje, las guías lineales, que facilitan el desplazamiento suave y estable de la herramienta, y el motor perforador, encargado de realizar los orificios en la placa PCB.

Además, la estructura y la placa de trabajo proporcionan la base necesaria para soportar todos estos componentes y asegurar la rigidez del sistema.

Los materiales utilizados en su fabricación pueden variar según el tipo de máquina y su propósito. En proyectos caseros o educativos, es común encontrar estructuras de aluminio, plásticos o madera, mientras que en la industria suelen emplearse materiales más robustos como el acero para garantizar mayor resistencia y durabilidad.

Cada uno de estos materiales y componentes cumple un rol fundamental en el rendimiento de la CNC. Una estructura firme reduce vibraciones y mejora la precisión, mientras que unas guías bien alineadas permiten movimientos más suaves. De esta manera, todos los elementos trabajan en conjunto para lograr perforaciones exactas y eficientes en las placas PCB.

ESTRUCTURA

La estructura de una máquina CNC es una parte clave, ya que brinda soporte y estabilidad a todo el sistema. Está formada por un marco o chasis, guías lineales, husillos o tuercas, y una plataforma de trabajo. Todos estos elementos trabajan en conjunto para asegurar que la herramienta de perforación se desplace con precisión sobre la placa PCB.

MOTORES PASO A PASO:



Los motores paso a paso son un componente esencial en una máquina CNC, ya que permiten un control preciso del movimiento en los ejes X, Y e Z. A diferencia de otros motores, los motores paso a paso se desplazan en incrementos exactos (pasos), lo que facilita un posicionamiento preciso de la herramienta de perforación sobre la placa PCB.

Estos motores funcionan mediante pulsos eléctricos que activan sus bobinas en una secuencia específica, generando movimientos controlados sin necesidad de sensores de posición adicionales. Existen dos tipos principales:

- Motores de pasos completos (full-step): Se mueven en pasos fijos, adecuados para aplicaciones donde la precisión no es extrema.
- Motores de micro pasos (microstepping): Dividen cada paso en fracciones más pequeñas, proporcionando mayor suavidad y precisión en los movimientos.

La elección del motor adecuado es muy importante en una CNC para PCB. Un motor con bajo torque podría perder pasos, causando errores en la perforación, mientras que un motor muy grande puede ser innecesario y consumir más energía. Generalmente, se utilizan motores NEMA 17 o NEMA 23, que ofrecen un buen equilibrio entre tamaño, potencia y precisión.

El uso de motores paso a paso en una CNC educativa permite a los estudiantes comprender principios de control de movimiento y automatización, habilidades fundamentales en la ingeniería y la manufactura.

GUÍAS Y VARILLAS ENROSCADAS:

Las guías son un componente esencial en la estructura de la CNC, ya que permiten tanto la movilidad como la estabilidad del sistema. Las guías lineales, en particular, facilitan un movimiento lineal suave y preciso a lo largo de los ejes X, Y e Z. Estas guías están diseñadas para minimizar la fricción y el juego mecánico, lo que garantiza que los movimientos sean consistentes y exactos, incluso durante operaciones repetitivas. Esto es especialmente importante en aplicaciones como la perforación de placas PCB, donde la precisión en el posicionamiento es crítica para asegurar que los agujeros estén correctamente alineados y dimensionados.



Las varillas roscadas desempeñan un papel crucial en la conversión del movimiento rotativo del eje del motor en movimiento lineal. Cuando el motor paso a paso gira, la rotación se transmite a la varilla roscada, que a su vez mueve una tuerca o un acoplador a lo largo de su longitud. Este mecanismo permite un desplazamiento lineal preciso y controlado, esencial para el posicionamiento exacto de la herramienta de corte en los ejes X, Y y Z de la CNC. La precisión de este sistema depende en gran medida del paso de la rosca de la varilla, que determina la distancia que se desplaza la tuerca por cada vuelta completa del motor. Este diseño es fundamental para garantizar movimientos suaves y repetibles, lo que es especialmente importante en aplicaciones como la perforación de placas PCB, donde la exactitud es crítica.



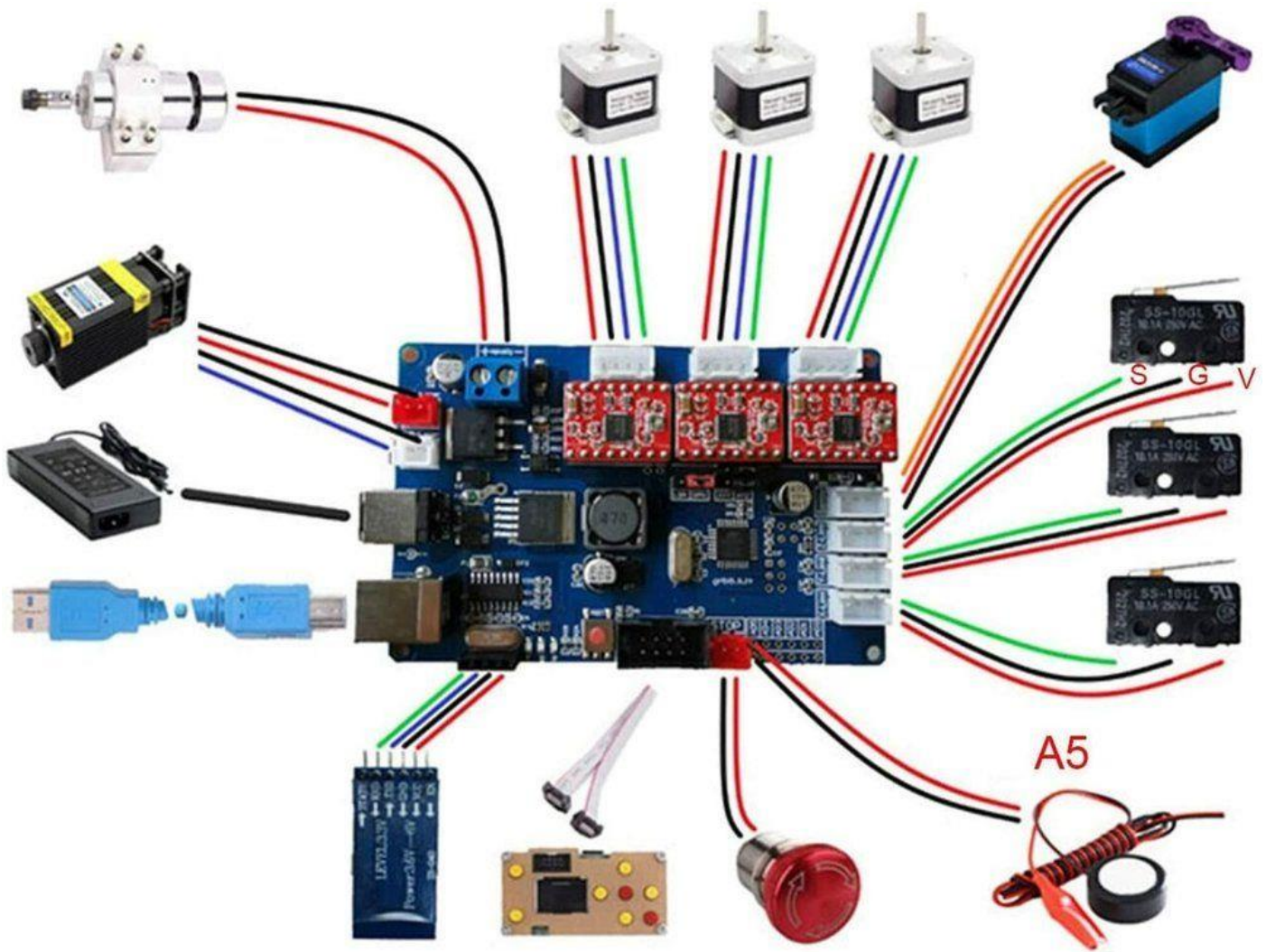
SISTEMA DE CONTROL

El sistema de control es el "cerebro" de la máquina CNC, ya que se encarga de interpretar las órdenes y convertirlas en movimientos precisos en los ejes de la máquina. En este caso, el sistema está basado en una placa programable **Arduino** junto con el firmware de código abierto **GRBL**, una combinación que permite analizar y ejecutar instrucciones escritas en código G.

El código G es el lenguaje estándar utilizado en máquinas CNC para indicar qué movimientos debe realizar la herramienta de trabajo. Cuando se carga un archivo con código G en la CNC, el **Arduino con GRBL** lo interpreta y envía las señales necesarias a los controladores de los motores paso a paso, quienes a su vez mueven la herramienta en los ejes X, Y e Z con precisión.

Uno de los principales beneficios de este sistema es su **bajo costo** en comparación con controladores CNC comerciales, que pueden ser bastante costosos. Además, al ser un sistema de código abierto, **GRBL es accesible y ampliamente utilizado**, lo que significa que cuenta con una gran comunidad de usuarios y desarrolladores que ofrecen soporte, mejoras y soluciones a posibles problemas.

Otra ventaja es su **facilidad de uso y configuración**. No se necesita un software complejo para programarlo, y se puede controlar desde una computadora mediante programas como **Universal Gcode Sender** o **Candle**, que permiten cargar y enviar los archivos de código G de manera sencilla.



En esta imagen podemos apreciar los materiales básicos para una CNC, en donde tenemos los 3 motores paso a paso, 3 fines de carreras para limitar la posición de los ejes, un elemento perforador o cortador, una fuente de poder, un lector de memorias SD, una pantalla con pulsadores, los drivers de motores, etc....

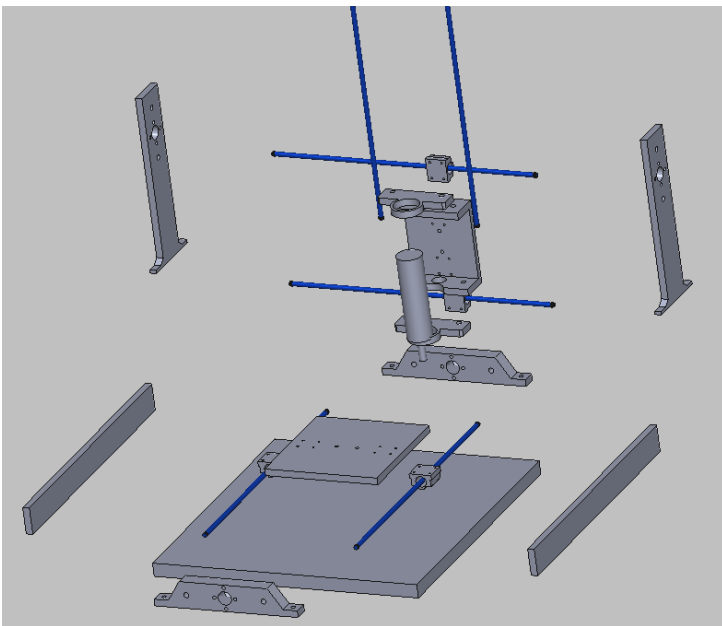
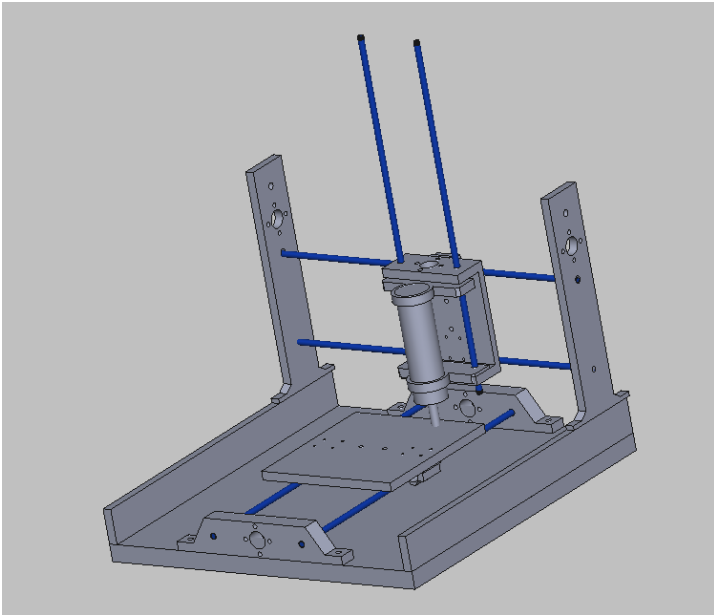
DISEÑO Y DESARROLLO

ETAPAS DEL DISEÑO CNC:

El primer paso en el desarrollo de la CNC fue la creación de un prototipo digital utilizando **SolidWorks**, un software especializado en diseño 3D. Este programa permitió modelar cada uno de los componentes de la máquina, como la estructura, los motores, las guías lineales y el sistema de perforación, asegurando que todas las piezas encajaran correctamente antes de la fabricación.

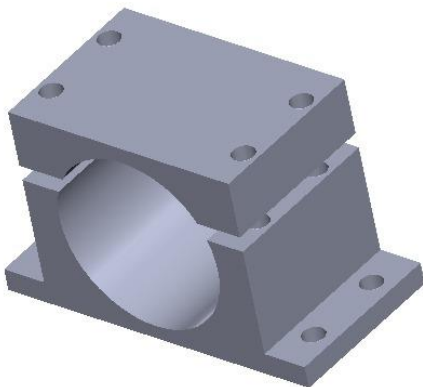
Gracias a SolidWorks, fue posible visualizar el diseño en un entorno virtual, realizar simulaciones de movimiento y detectar posibles fallos estructurales o de ensamblaje. Además, nos permitió optimizar el uso de materiales y prever los ajustes necesarios para garantizar un montaje preciso y funcional.

Una vez que el diseño digital estuvo finalizado y validado, se procedió a la fabricación del prototipo físico, utilizando los materiales adecuados y ensamblando cada componente según lo planificado en el modelo 3D. Este proceso facilitó la transición del diseño teórico a una máquina real, minimizando errores y asegurando un mejor resultado en la construcción de la CNC.



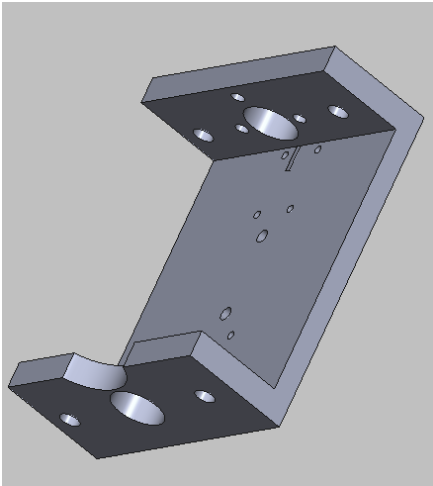
En base a este diseño se decidieron las modificaciones y planteamiento del diseño actual de la máquina, en cuanto a los diseños de algunos elementos fueron impresos en 3D, los cuales fueron:

Soporte de motor de taladro:



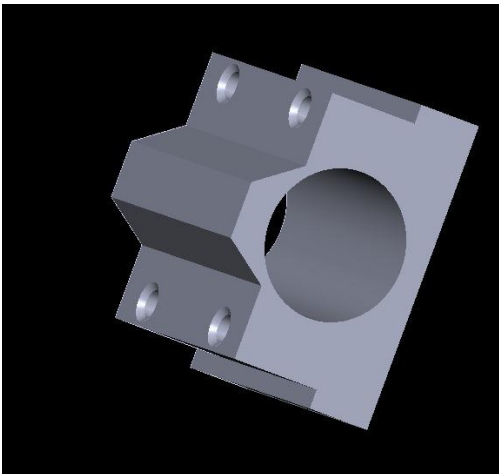
Soporte eje Y:

El mismo consta con dos orificios para la colocación de rodamientos lineales que permitirán un movimiento más suave por los ejes.



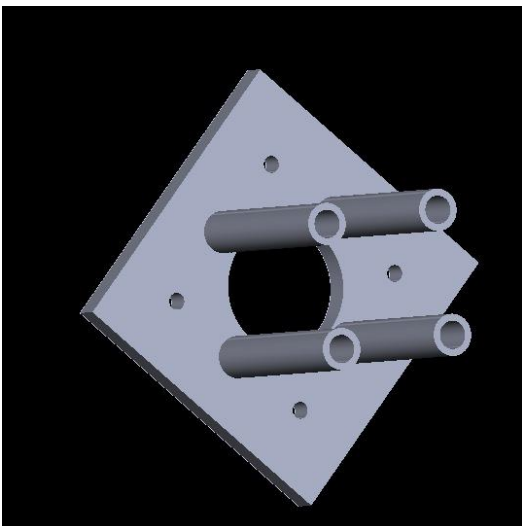
Soporte lineales:

Estos soportes son utilizados en conjunto con un rodamiento lineal, se utilizan en los ejes X e Z.



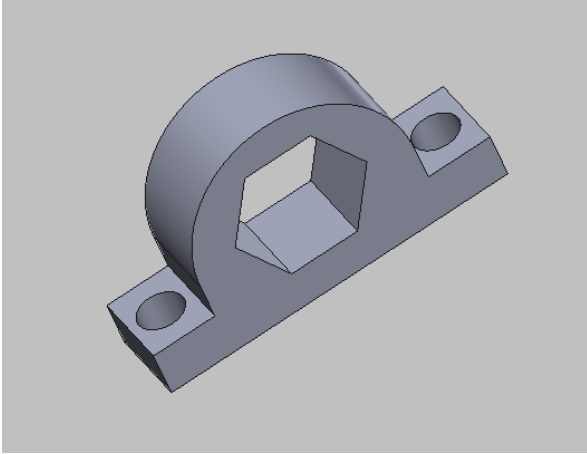
Soporte para motores Nema 17:

Se utiliza para soportar los motores, los orificios chicos son para el acople del motor y los separadores son para acoplar a la base del CNC.



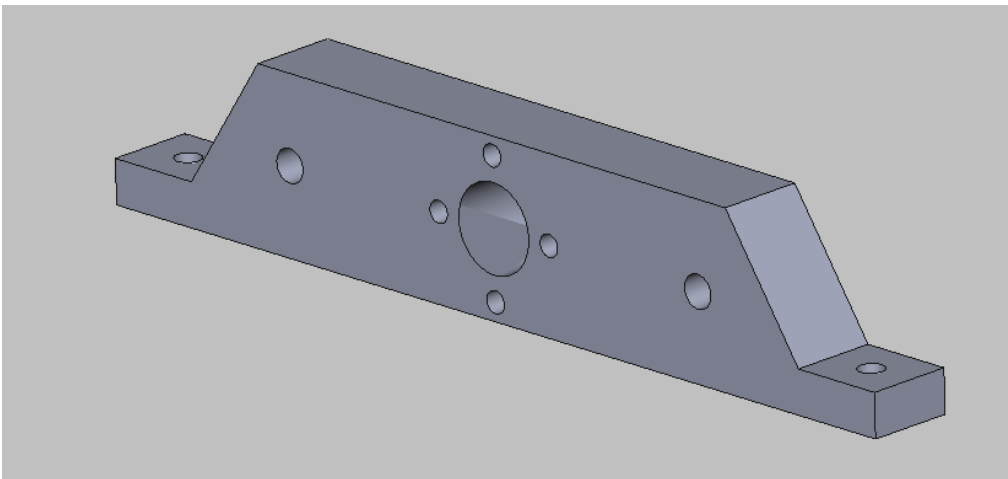
Soporte de tuerca M8:

El soporte en conjunto con una tuerca, se encargan del desplazamiento en los ejes.



Soporte del eje X:

Este soporte acoplado a la base soporta dos varillas lineales y el motor en el centro.



Base:

Una tabla de madera con una medida de 400mm x 400mm



Se deben realizar cuatro perforaciones en las piezas con medidas precisas, ya que estas deben acoplar los soportes de las guías y varillas del eje X.

Soportes laterales:

Dos maderas de 100mm x 400mm



A cada una de las maderas se le deben realizar tres perforaciones. Dos de ellas con un diámetro de 8 mm que están destinadas a acoplar las varillas guías que conectan las piezas. La perforación central, por su parte, tiene un diámetro de 22 mm. En una de las maderas, se debe instalar un rodamiento con medidas de 8 mm en su diámetro interior y 22 mm en el exterior.

Estas maderas cumplen la función de soportes laterales de la estructura. Junto con las varillas guías, permiten el desplazamiento suave y preciso del taladro a lo largo del eje Z de la CNC, facilitando el movimiento vertical necesario para realizar las perforaciones en las placas PCB.

PROCESO DE ARMADO:

En esta etapa además de los materiales previamente mencionados, se necesitan de materiales menores, tales como: tornillos; tuercas; arandelas; varillas roscadas; etc.

Estos se pueden conseguir fácilmente en ferreterías.



Se debe acoplar los soportes del eje X a la tabla de 400mm x 400mm, se debe comprobar que ambas estén en las mismas posiciones para que a futuro no haya problemas de alineación para que los motores no tengan que esforzarse demás por la fricción.

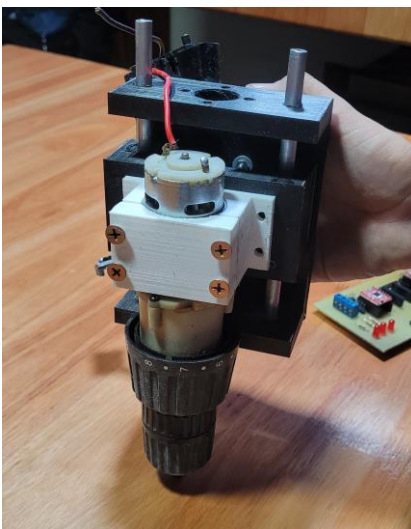


Se debe colocar los soportes lineales a la base y colocar en el centro una tuerca M8 para que se realice el movimiento lineal según el giro del motor. La sujeción de los soportes y la tuerca se puede realizar con tornillos u pegamentos.



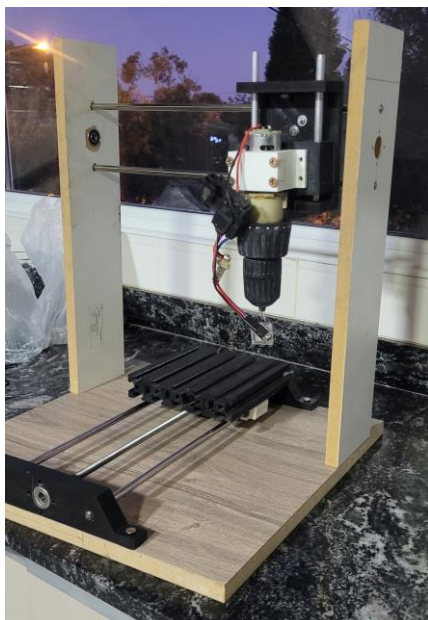
Una vez armada la base con sus soportes, se deben colocar las varillas y comprobar que la base se desplace según el giro de la varilla.

Terminando esta etapa del proyecto se puede realizar el montaje de los otros ejes, debido a que se necesita saber la posición de la base cuando se encuentra en uno de los extremos.



Previamente se debe realizar el montaje del taladro, el mismo consta de 3 piezas impresas 3D, en donde uno es el soporte del motor, otro es el soporte que realizara el movimiento vertical y por último el soporte que realizara el movimiento horizontal.

Para el taladro se reciclo el motor y el mandril de un taladro de mano.

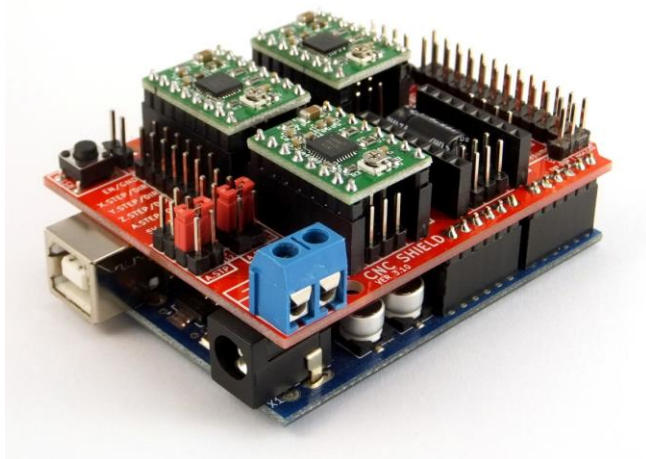


Teniendo todas las piezas armadas y en posición, se puede realizar el montaje de todas las maderas a la base, hay que comprobar las posiciones antes de realizar las perforaciones.

PARTE ELECTRONICA:

Para la parte electrónica del proyecto se requieren de varios materiales, que bien fácilmente se pueden comprar o armar por nosotros mismos:

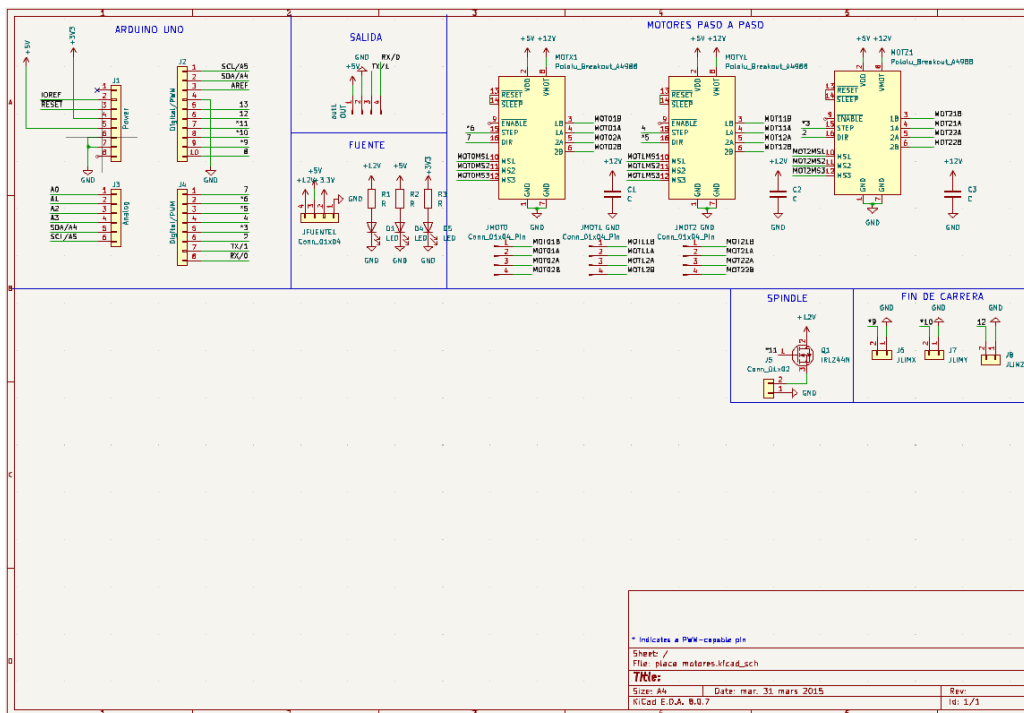
- Shield Arduino:



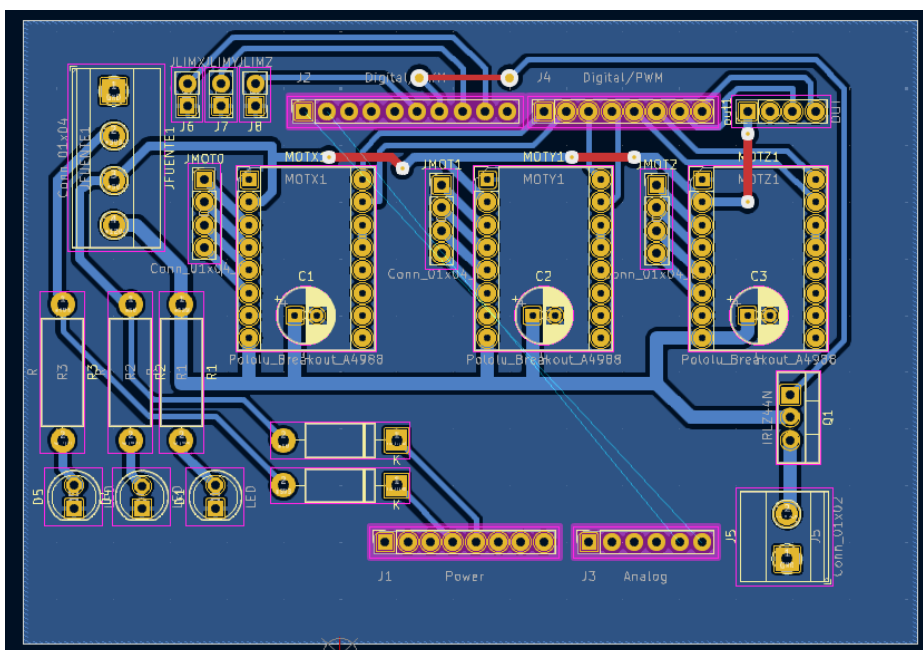
Este shield de Arduino Uno es una opción buena para este proyecto, pero en el caso de este proyecto se fabricó un prototipo de este shield.

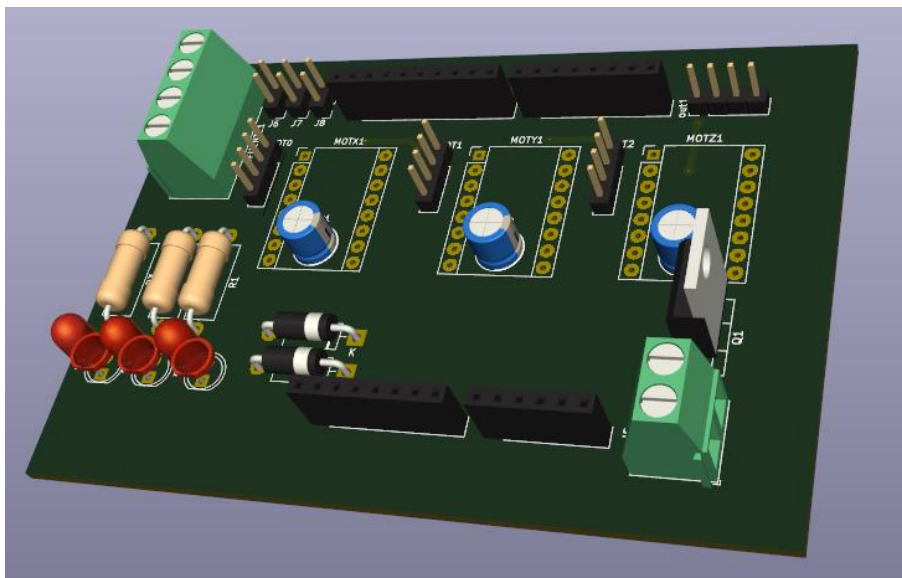
Circuito esquemático:

Este shield cuenta con las conexiones para los motores paso a paso, la salida de 12V para el motor del taladro, los fines de carrera y una conexión RX e TX para conectar el Arduino con el ESP32.



Circuito PCB:



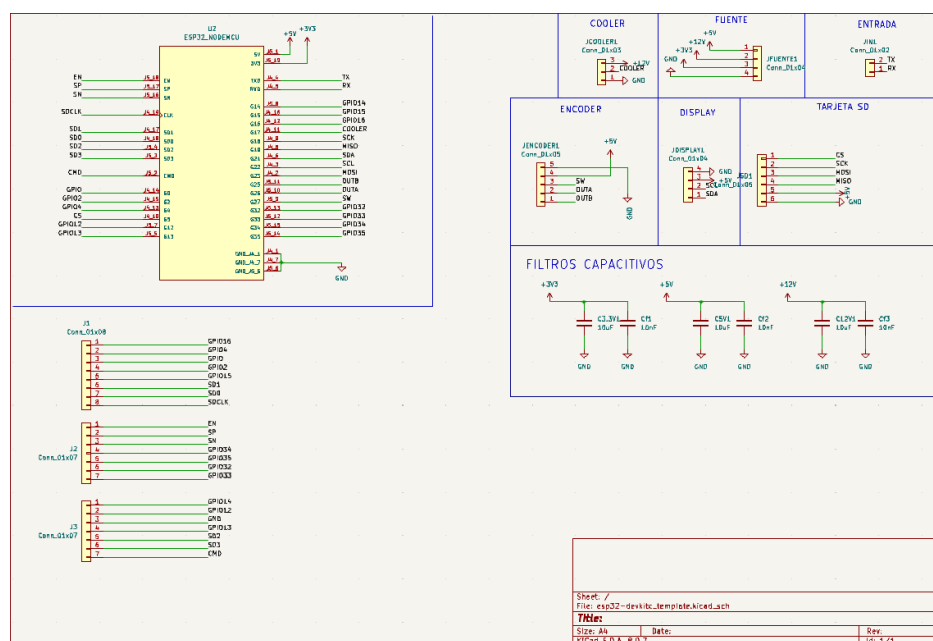


Materiales:

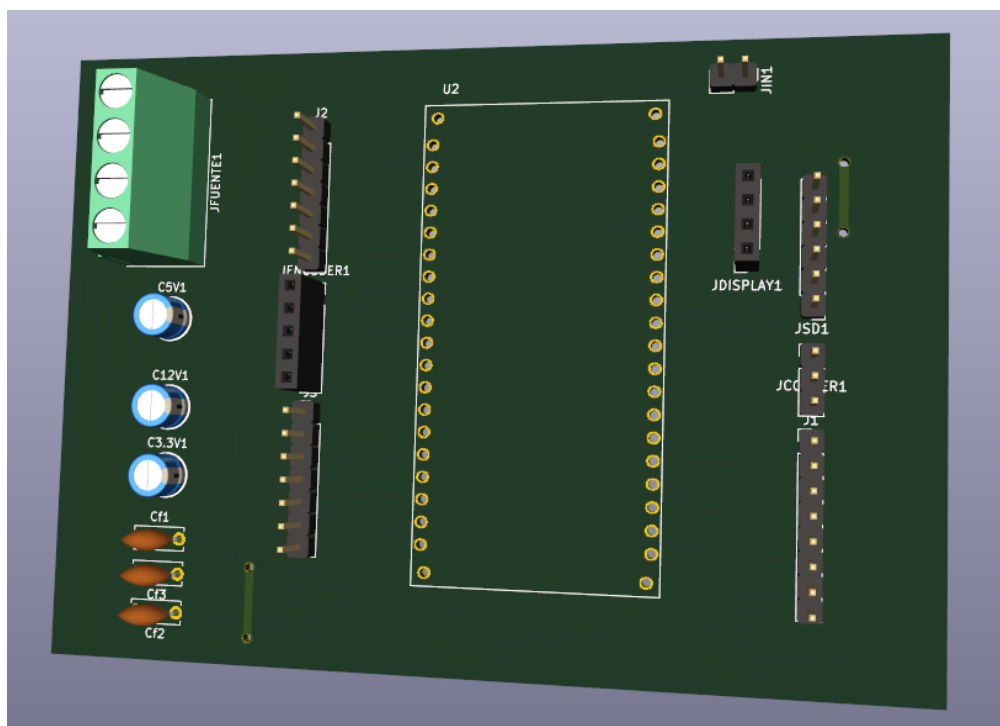
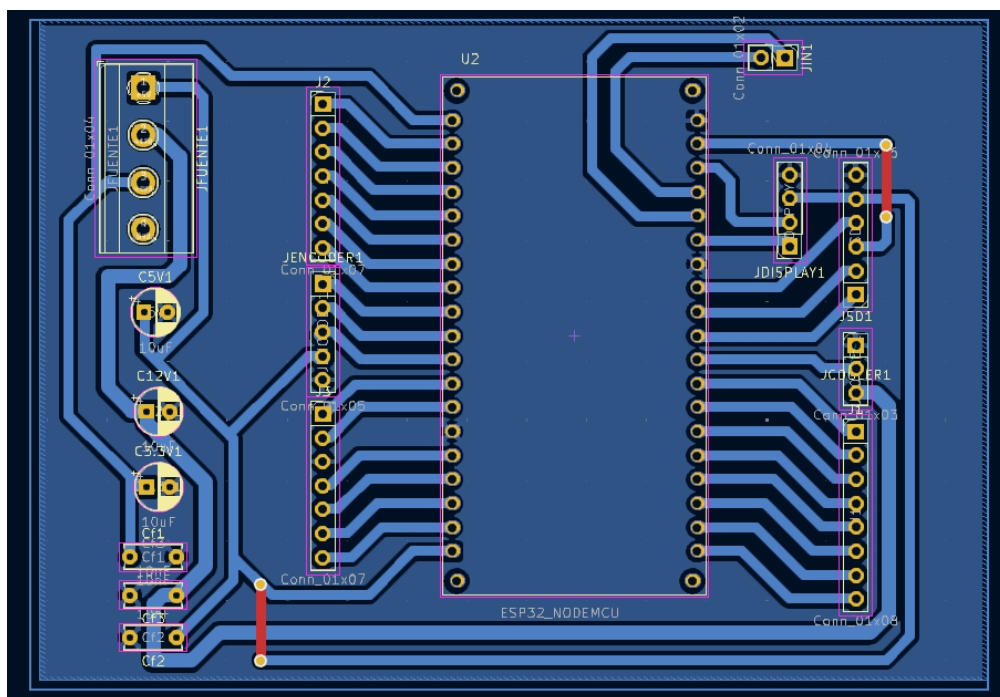
- Resistencias 1k Ohm x 3UN
 - Leds indicadores de estado x 3UN
 - Borneras doble x 3UN
 - MOSFET IRLZ44N x 1UN
 - Pines hembra 9 pines por lado x 6 UN
 - Pines macho x 60 UN
 - Diodos Zener (opcionales) x 3 UN (3,3V, 5V y 12V)
 - Drivers de motores A4988 u DRV8825
 - Placa 10cm x 10cm
- Shield ESP32

Este shield para el ESP32 nos permite utilizar todos sus pines con mayor facilidad, posee conexiones específicas como: para un cooler que se planteo utilizar para mantener temperaturas estables en los drivers de motores; conexión para una pantalla; una conexión para un lector de tarjetas SD y la conexión para el encoder; además posee pines disponibles para lo que se requiera a futuro.

Circuito esquemático:



Circuito PCB:



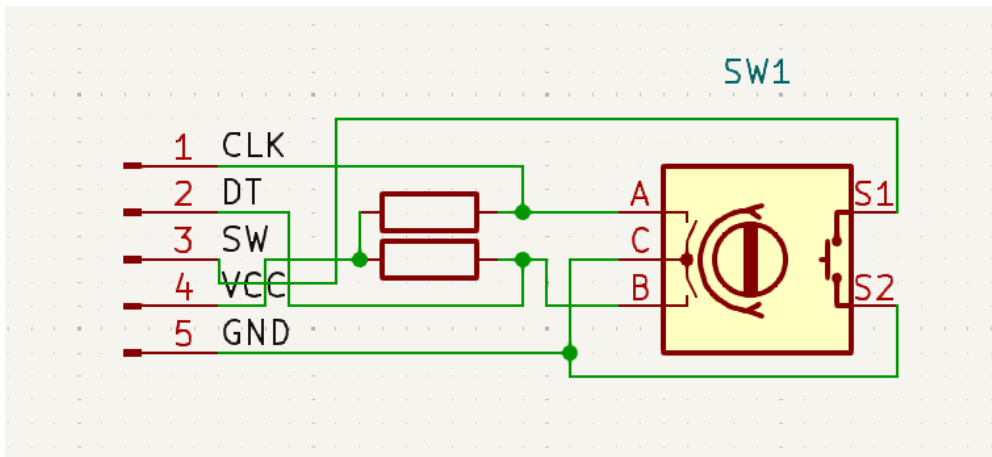
Materiales:

- Capacitor 16uF 25V x 3 UN.
- Capacitor 10nF x 3 UN
- Bornera doble x 2 UN
- Pines hembras
- Pines machos.
- Placa 10cm x 10 cm.

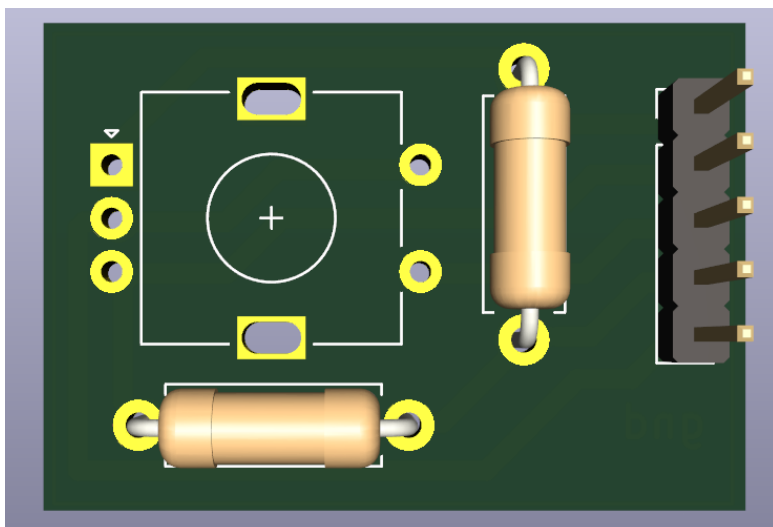
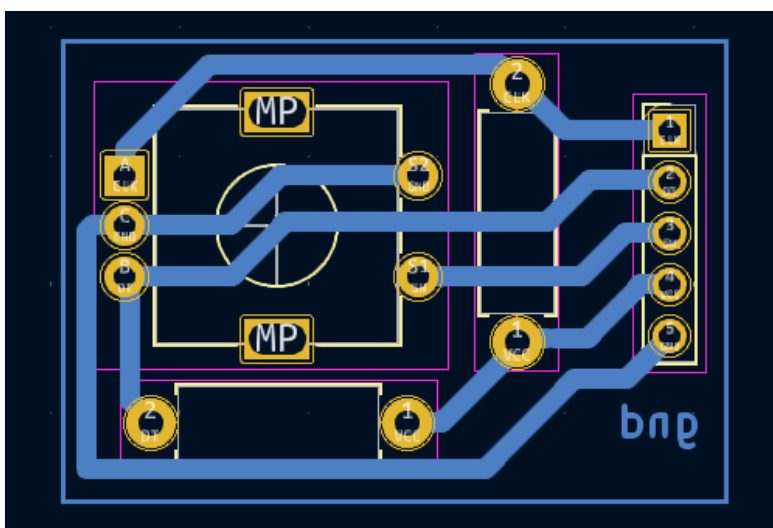
- Placa encoder:

Placa sencilla para poder realizar la conexión del encoder al ESP32.

Circuito esquemático:



Circuito PCB:



Materiales:

- Placa 3cm x 3cm
- Resistencia 1k x 2UN
- Encoder
- Pines hembra

- Alimentación:

Para la alimentar toda la maquina se puede utilizar una fuente de poder reciclada, en donde necesitaremos realizar unas modificaciones debido a que únicamente necesitaremos 4-5 cables de esta.

GRBL

GRBL es un firmware de código abierto y alto rendimiento para controlar máquinas CNC (Control Numérico Computarizado), como fresadoras, grabadoras láser y cortadoras de plasma.

GRBL recibe comandos en G-code, un lenguaje de programación estándar utilizado en la industria CNC. El G-code especifica las coordenadas, velocidades y otras instrucciones para el movimiento de la herramienta. GRBL actúa como un intérprete, traduciendo estos comandos en señales eléctricas que controlan los motores de la máquina.

Este está diseñado para controlar motores paso a paso, que son motores eléctricos que giran en incrementos precisos. El firmware genera pulsos eléctricos que controlan el movimiento de los motores, permitiendo un posicionamiento preciso de la herramienta. GRBL optimiza la aceleración y desaceleración de los motores para lograr movimientos suaves y precisos.

SOFTWARE ESP-32

COMUNICACION CON GRBL


Los creadores de GRBL, recomiendan varios protocolos de comunicación para interconectar con GRBL, en nuestro caso utilizaremos un protocolo de transmisión envío-respuesta

El protocolo de transmisión de envío-respuesta se presenta como el método más robusto y sencillo para la transmisión de programas de código G a GRBL. Este protocolo opera mediante el envío secuencial de líneas de código G desde el transmisor a GRBL, seguido de una espera de confirmación, ya sea una respuesta 'ok' o 'error', antes de proceder con el envío de la siguiente línea.

Esta metodología garantiza la integridad de la transmisión al sincronizar la ejecución entre el transmisor y GRBL. La confirmación de recepción permite que el transmisor verifique que cada línea de código G ha sido procesada correctamente por GRBL, independientemente de las posibles demoras en el procesamiento o la ejecución de las instrucciones por parte de GRBL.

La implementación de este protocolo asegura que no se pierdan instrucciones y que la secuencia de ejecución se mantenga precisa, lo que resulta crucial para la operación correcta de sistemas de control numérico.

Comenzamos definiendo dos protocolos UART en el ESP32



```

1 void COM_SERIAL_SETUP() {
2   Serial.begin(115200);
3   arduinoSerial.begin(115200, SERIAL_8N1, ARDUINO_RX_PIN, ARDUINO_TX_PIN);
4 }

```

Serial se utilizará para monitorear el comportamiento del ESP32 y para debuggear cualquier error.

Arduino Serial se utilizará para comunicarse del GRBL

Esta distinción se debe a que, si queremos utilizar la misma interfaz UART toda información enviada del esp32 será recibida por el Arduino, incluido cualquier mensaje de debug, lo que puede traer conflicto con GRBL.

Luego de esto, continuamente revisaremos si hay algún mensaje que ha sido enviado por GRBL:




```

1 void COM_SERIAL_LOOP() {
2   if (arduinoSerial.available() > 0) {
3     String grblResponse = arduinoSerial.readStringUntil('\n');
4     grblResponse.trim();
5     if (grblResponse.length() > 0) {
6       Serial.print("Respuesta de Grbl: ");
7       Serial.println(grblResponse);
8     }
9   }
10 }

```

En un archivo de código G, cada línea es uno o varios movimientos de distintos tipos, desde precisos hasta lineales rápidos. Necesitamos enviar línea por línea del archivo al GRBL, por lo que, necesitamos una función que procese un archivo seleccionado y envíe la información a GRBL



```

1 void handleFileSelection(fs::FS& fs, const String filename)

```

HandleFileSelection hace exactamente eso, empieza intentando abrir el archivo y devolver un error en caso de que sea necesario:

```

1  Serial.print("\r\nAbriendo archivo: ");
2  Serial.println(filename);
3
4  String temppath = String("/") + String(filename);
5  const char* path = temppath.c_str();
6  File arch = fs.open(path);
7
8  if (!arch) {
9      Serial.println("No se pudo abrir el archivo");
10     return;
11 }

```

Una vez que pasemos este proceso, sabemos que el archivo se ha podido abrir correctamente por lo que prosigue procesar las líneas:

```

1  // Procesa cada línea del archivo y la envía a GRBL
2  while (arch.available()) {
3      String line = readLine(arch);
4      if(arduinoSerial.available() > 0) {
5          String grblResponse = arduinoSerial.readStringUntil('\n');
6          Serial.println("Respuesta de GRBL: ");
7          Serial.println(grblResponse);
8      }
9      if (line != "") {
10         Serial.println("Mandando a GRBL: ");
11         Serial.println(line);
12         arduinoSerial.println(line);
13         delay(100);
14     }
15 }
16
17 arch.close();
18 Serial.println("\r\nLectura de archivo completada");

```

GRBL distingue una línea de otra ya sea por una nueva línea “\n” o un “retorno de carruaje” “\r”

Nótese la distinción entre enviar la línea al Serial y al Serial de comunicación con GRBL

Además, no queremos enviar una línea vacía, por lo que debemos asegurarnos de que ese no sea el caso.

Mas adelante se explicará cómo se selecciona el archivo.

TARJETA SD

Hay tres cosas importantes que se deben hacer con la tarjeta SD, poder listar todos los archivos que contiene, accederlos y leer cada línea.

Para la primera `listDir` hace exactamente eso.

```
1 void listDir(fs::FS &fs, const char *dirname, uint8_t levels)
```

Su funcionamiento es muy sencillo, simplemente utilizaremos la librería incluida con el ESP para acceder a la ruta y a cada uno de sus archivos

```
1 File root = fs.open(dirname);
```

```
1  
2 File file = root.openNextFile();
```

También debemos asegurarnos de incluir cualquier carpeta dentro del directorio:

```
1 if (levels) {  
2     listDir(fs, file.path(), levels - 1);  
3 }  
4 }
```

Leer cada línea es aún más simple, una vez comenzamos a leer el archivo simplemente nos detendremos en cada “\n”
o “\r”

```
1 String readLine(File& file) {  
2     String line = "";  
3     while (file.available()) {  
4         char c = file.read();  
5         if (c == '\n' || c == '\r') break;  
6         line += c;  
7     }  
8     return line;  
9 }
```

WIFI

El WIFI se utilizará para conectarse al servidor y poder recibir archivos a través de la red y no tener que utilizar la tarjeta SD.

Actualmente solo podemos conectarnos a internet, pero no podemos hacer nada con eso.

```
1 void wifisetup()  
2 {  
3     WiFi.mode(WIFI_STA);  
4     pinMode(btnGPIO, INPUT);  
5     wifistart(ssid, password);  
6     Serial.println("Configuración completada");  
7     BuildWifiScreen();  
8 }
```

Comenzamos poniendo la antena en modo estación, en la que solo podemos enviar y recibir, luego conectándonos a una red a través de wifistart, que enviara los valores por defecto de SSID y contraseña, que en caso de que no sea modificado estarán vacíos.

BuildWifiScreen será explicado en el apartado de LCD.
Debemos hacer dos cosas en este apartado, escanear las redes y conectarnos.


```

1 void wifiscan()
2 {
3   Serial.println("Iniciando escaneo");
4
5   // WiFi.scanNetworks devuelve el número de redes encontradas
6   int n = WiFi.scanNetworks();
7   Serial.println("Escaneo finalizado");
8
9   if (n == 0)
10  {
11    Serial.println("No se encontraron redes");
12  }
13  else
14  {
15    Serial.print(n);
16    Serial.println(" redes encontradas");
17    for (int i = 0; i < n; ++i)
18    {
19      WIFI[i] = WiFi.SSID(i).c_str();
20      Serial.printf("%-32.32s", WiFi.SSID(i).c_str());
21      Serial.println();
22      delay(10);
23    }
24  }
25
26  Serial.println("");
27
28  // Eliminar el resultado del escaneo para liberar memoria
29  WiFi.scanDelete();
30
31  // Esperar un poco antes de escanear de nuevo
32  delay(5000);
33 }

```

Wifiscan, hace exactamente lo que dice su nombre, buscamos las redes disponibles y las guardamos en un Array para su uso posterior.

```

1 void wifistart(char *ssid, char *password)

```

Wifistart nos conectara dado un SSID y una contraseña, y logeara su resultado.

LCD

El LCD es lo que nos permite controlar la CNC, por tanto, debemos poder:

- Seleccionar un archivo ya sea desde la SD o el Servidor.
- Conectarnos a WIFI
- Obtener información sobre el estado actual de la maquina

Archivos

Con lo explicado antes en el apartado de SD, ya tenemos una manera de enlistar todos los archivos dentro de un directorio, por lo que simplemente deberemos crear un nuevo ítem para cada archivo, con una función como callback para poder hacer algo sobre él.

```

1 void buildSDlocScreen() {
2   Serial.println("Building Local Files Screen...");
3   if (SDlocScreen != nullptr) {
4     delete SDlocScreen;
5     SDlocScreen = nullptr;
6   }
7
8   for (int i = 0; i < MAX_FILES + 2; i++) {
9     SDlocItems[i] = nullptr;
10  }
11
12  int menuIndex = 0;
13
14  SDlocItems[menuIndex++] = new MenuItem("Archivos Locales");
15
16  for (int i = 0; i < counter && i < 50; i++) {
17    if (Files[i].length() > 0) {
18      String* filename = new String(Files[i]);
19      String fn = *filename;
20      char* menuItemText = strdup(fn.c_str());
21
22      SDlocItems[menuIndex++] = new FileMenuItem(menuItemText, *filename, SD, handleFileSelection);
23    }
24  }
25
26  SDlocItems[menuIndex++] = new ItemBack();
27  SDlocItems[menuIndex] = nullptr;
28
29  SDlocScreen = new MenuScreen(SDlocItems);
30 }

```

Info

GRBL nos proporciona una manera de saber el estado actual de la máquina, enviando un “?” a través del Serial.

Luego debemos analizar la información para obtener lo que nos interesa:

```

1 void DatosPosicion() {
2   delay(100);
3   arduinoSerial.println("?");
4   if (arduinoSerial.available() > 0) {
5     grblline = arduinoSerial.readStringUntil('\n');
6     grblline.trim();
7     int inicioPosW = grblline.indexOf("WPos:");
8     if (inicioPosW > 0) {
9       int posFin = grblline.indexOf(',', posInicio);
10      if (posFin < 0) posFin = grblline.indexOf('>', posInicio);
11      if (posFin < 0) return; // Error en formato
12
13      String datosPosicion = grblline.substring(posInicio, posFin);
14
15      // Dividir por comas
16      int primeraComa = datosPosicion.indexOf(',');
17      int segundaComa = datosPosicion.indexOf(',', primeraComa + 1);
18
19      if (primeraComa > 0 && segundaComa > 0) {
20        XC = datosPosicion.substring(0, primeraComa).toFloat();
21        YC = datosPosicion.substring(primeraComa + 1, segundaComa).toFloat();
22        ZC = datosPosicion.substring(segundaComa + 1).toFloat();
23      }
24    }
25
26  }
27 }

```

CONCLUSIÓN

Este proyecto fue interesante de abordar. Al principio nos confiamos con los tiempos y pensamos que iba a ser algo rápido de armar, pero fue todo lo contrario, con las cursadas de las materias, las pasantías y temas de universidad no pudimos entregarlo en tiempo y forma.

Luego al diseñar lo que sería la estructura nos vimos con varias dificultades, como: “¿Cómo hacemos la base?”, “¿Y si probamos agregando esto?” “¿Se le puede poner esto?” Y así con muchas preguntas, fuimos resolviendo poco a poco y conseguimos el diseño final.

Un verdadero problema fue el diseño del shield del Arduino, no solo el diseño, sino que también el desarrollo, teniéndolo que rehacer como 4-5 veces, es por esto por lo que es importante repasar varias veces los circuitos antes de pasarlos a una placa. Ya que no es solo una pérdida de dinero, sino que también de tiempo, nosotros precisábamos la placa antes de las vacaciones, por lo que la hicimos de manera apurada durante el ciclo lectivo, pero al momento de empezar a armar el proyecto nos dimos cuenta que estaba mal, y decíamos “¿Y ahora que hacemos”, debido a que la escuela estaba cerrada y no teníamos como conseguir los materiales de forma local, así que tuvimos que comprar desde Buenos Aires, y con las demoras que hubieron, pasaron los días lo cual fue tiempo perdido, ya que sin la placa no podíamos hacer mucho más. En cuanto llegaron los materiales, lo armamos lo más rápido que pudimos y empezamos a probar los motores.

A pesar de las complicaciones que tuvimos, pudimos resolverlas y concretamos el proyecto.

BIBLIOGRAFIA

[Qué Es Una Máquina CNC Y Su Funcionamiento Explicado](#) ✓

[CÓMO CONSTRUI MI CNC PARTEI - Electrónica Práctica Aplicada](#)

[GRBL with Arduino CNC Shield – Complete Guide](#)

[ESP32 UART Communication Pins Explained with Example](#)

[ESP32 UART - Serial Communication, Send and Receive Data \(Arduino IDE\) | Random Nerd Tutorials](#)

[Connect ESP32 to WiFi - Step-By-Step Tutorial](#)

[ESP32: Guide for MicroSD Card Module Arduino | Random Nerd Tutorials](#)

[GitHub - grbl/grbl: An open source, embedded, high performance g-code-parser and CNC milling controller written in optimized C that will run on a straight Arduino](#)