
**LUIS GUSTAVO GUIDELLI REVOLTI**

# PLOTTER DE BAIXO CUSTO PARA PROTOTIPAÇÃO DE PLACAS DE CIRCUITO IMPRESSO

**LAVRAS – MG**

**2019**

**LUIS GUSTAVO GUIDELLI REVOLTI**

# PLOTTER DE BAIXO CUSTO PARA PROTOTIPAÇÃO DE PLACAS DE CIRCUITO IMPRESSO

Relatório Técnico apresentado à Universidade Federal de Lavras, como parte das exigências do curso de Engenharia de Controle e Automação, para a obtenção do título de Bacharel.

Prof. Wilian Soares Lacerda
Orientador

**LAVRAS – MG**
**2019**

# RESUMO

O presente trabalho tem como objetivo a construção e implementação de uma *Plotter* de baixo custo utilizando microcontrolador Arduino e drivers CD provenientes de computadores já inutilizados na Universidade. Desta forma, além de reutilizar equipamentos já descartados, torna o produto acessível à toda a comunidade acadêmica. O trabalho demonstra o desenvolvimento de toda a estrutura mecânica, que permite a movimentação da ferramenta nos eixos X, Y e Z, além do projeto elétrico e o desenvolvimento do software. O equipamento tem por objetivo imprimir trilhos em placas de circuito eletrônico para posterior imersão em percloreto de ferro. Para isto, foi utilizado o *software* KiCad que foi responsável pelo projeto da placa e desenho do trilho impresso. Ele é capaz de interpretar o desenho criado e gerar um arquivo *SVG*, que posteriormente foi lido pelo *software* Inkscape. Este permite editar o projeto importado e setar determinadas configurações de impressão, além de exportar o arquivo em formato *G Code*. O *software* Processing é o responsável pela comunicação entre o computador e o controlador, além de interpretar o arquivo *G Code* e enviar as instruções de impressão ao controlador. O *software* Arduino IDE, por fim, foi responsável pela programação do controlador e controle dos motores.

O projeto resultou em um protótipo capaz de realizar impressão em placas de circuito eletrônico de forma eficaz, apresentando nível considerável de precisão e possuindo uma velocidade relativamente alta. Resultou, ainda, em um equipamento compacto, de baixo custo e que reutilizou equipamentos já inutilizados pela Universidade, gerando um produto acessível aos alunos da Universidade.

A *Plotter*, porém, não apresentou bons resultados na impressão de formas circulares ou de placas muito complexas em que a precisão precisa ser superior. Para projetos de placas menos complexas e sem formas circulares, no entanto, o equipamento apresentou ótimos resultados.



**Palavras-chave:** Plotter. Arduino. L293D. Prototipação. CNC. Motor de passo.

# ABSTRACT

The present work aims to construct and implement a low cost Plotter using Arduino microcontroller and CD drivers from computers already unused at the University. Thus, in addition to reusing discarded equipment, it makes the product accessible to the entire academic community. The work demonstrates the development of the entire mechanical structure, which allows tool movement on the X, Y and Z axes, as well as electrical design and software development. The equipment aims to print rails on electronic circuit boards for later immersion in iron perchloride. For this, the KiCad software was used, which was responsible for the plate design and printed rail design. It is able to interpret the created drawing and generate an SVG file, which was later read by Inkscape software. It allows you to edit the imported project and set print settings, as well as export the file in G Code format. The Processing software is responsible for communication between the computer and the controller, as well as interpreting the G Code file and sending the printing instructions to the controller. Finally, the Arduino IDE software was responsible for controller programming and motor control.

The project resulted in a prototype capable of printing electronic circuit boards effectively with a considerable level of accuracy and relatively high speed. It also resulted in a compact, low cost equipment that reused equipment already unused by the University, generating a product accessible to the students of the University.

Plotter, however, did not perform well in printing very complex circular shapes or plates where the accuracy needs to be higher. For less complex and non-circular plate designs, however, the equipment had excellent results.



**Keywords:** Plotter. Arduino. L293D. Prototype. CNC. Stepper motor.

## LISTA DE FIGURAS

## LISTA DE TABELAS

**SUMÁRIO**

# 1 INTRODUÇÃO

A tecnologia de impressão 3D se mostra cada dia mais popular e demostra suas variadas aplicabilidades em praticamente todos os segmentos da industria. Apesar de parecer novidade, estudos relacionados à prototipagem rápida se iniciou há quase 40 anos por Hideo Kodama, do Instituto de Pesquisa Industrial Municipal de Nagoya, no Japão. Ele utilizou laser UV para modificar a estrutura de um fotopolímero, que altera facilmente sua estrutura química quando expostos a pequenas cargas de energia luminosa. Criando várias pequenas camadas deste material, foi possível realizar a impressão de um modelo sólido. A partir daí, novas tecnologias vem sendo desenvolvidas, e hoje é possível encontrar próteses, alimentos, sapatos e até órgãos humanos impressos tridimensionalmente (DIGITALTRENDS, 2019).

Com a eletrônica se fazendo cada vez mais presente nos mais diversos equipamentos, instrumentos de prototipagem de circuitos impressos são cada vez mais comuns e necessários, principalmente em ambientes universitários onde cursos relacionados à tecnologia são oferecidos. Apesar da demanda, equipamentos que realizam tal tarefa se encontram a preços elevados, inviabilizando, muitas vezes, sua utilização.

Com base na elevada demanda encontrada dentro da Universidade e na dificuldade em suprí-la, foi pensada a construção de um equipamento de prototipagem de circuito impresso que pudesse ser acessível a todos e que pudesse ser construída pelos próprios universitários a partir de materiais inutilizados pela Universidade, reduzindo os custos e reutilizando equipamentos já obsoletos.

A impressora foi construída reutilizando *drivers* de CD de computadores antigos e foi capaz de imprimir circuitos com área de até 50x50mm, de acordo com o comprimento dos eixos dos motores. Os *softwares KiCad* e *InkScape* foram responsáveis pelo projeto da placa, edição e geração do arquivo *G Code* que foi lido pela impressora. O *software Processing* foi responsável pela comunicação entre o computador e o controlador através da porta USB, e o controlador responsável pela interpretação dos códigos e pelo controle de todos os motores foi o Arduino Uno.

A técnica utilizada para construção da placa foi realizar a impressão de um *layout* pela *Plotter* sobre uma placa de fenolite cobreada. A ferramenta utilizada para gravar o desenho foi uma caneta de tinta permanente. Ela foi responsável por criar um isolante sobre os trilhos que fariam as conexões elétricas da placa. As áreas restantes de cobre devem ser removidas. Para isso, a placa foi submergida sobre uma solução de percloreto de ferro, que foi responsável por

reagir com o cobre e corroê-lo, mantendo na placa apenas o trilho isolado pela tinta da caneta. Posteriormente, foi utilizado álcool isopropílico para retirar a tinta da caneta.

## 1.1 Objetivo

O presente trabalho tem como objetivo a construção de uma *Plotter* para prototipação de placas de circuito impresso, assim como o desenvolvimento do *software* responsável por receber, editar e interpretar os comandos do arquivo *G Code*. Os componentes principais do equipamento devem ser reutilizados de *drivers* de CD de computadores antigos, e o custo final de todo o projeto deve ser baixo.

O objetivo geral é dividido nos seguintes objetivos específicos:

- Utilizar *softwares* para realizar o projeto da placa e gerar o arquivo *G Code*.
- Construção da estrutura física do projeto reutilizando materiais e componentes eletrônicos de *drivers* de CD de computadores antigos, visando baixo custo.
- Desenvolver um *software* para receber os comandos *G Code*, editá-los, interpretá-los e realizar o controle dos motores de acordo com o comando recebido.
- Realizar a impressão de uma placa de teste e validar o projeto.

## 1.2 Motivação

A construção de placas de circuito eletrônico, principalmente nos cursos relacionados à eletrônica, se faz cada vez mais necessária na Universidade Federal de Lavras. Apesar da alta demanda, existe a carência de equipamentos que realizem tal trabalho, visto o alto custo encontrado principalmente na indústria nacional.

Utilizando *drivers* de CD de computadores antigos, é possível construir uma mini *Plotter* que realiza impressão de trilhos de placas não muito complexas com elevada qualidade. Considerando que, na Universidade Federal de Lavras, existem dezenas de computadores armazenados já inutilizados e que podem ser desmontados para a obtenção de peças, a obtenção dos *drivers* pode ser feita sem dificuldades e sem qualquer custo.

O microcontrolador Arduino Uno, utilizado neste projeto, está cada vez mais presente na vida acadêmica dos alunos, sendo fácil de se encontrar e por um preço bastante acessível.

Demais itens utilizados no projeto também podem ser encontrados por preços bastante acessíveis ou até mesmo fabricados utilizando materiais encontrados em casa, resultando em um projeto barato, útil e contribuindo para a reciclagem de material.

## 1.3 Organização do texto

No capítulo 1, é apresentado uma introdução sobre o tema e sobre o projeto em específico. Apresenta também os objetivos gerais e específicos deste trabalho, assim como a motivação para realizá-lo e torná-lo real.

No capítulo 2, o relatório apresenta uma revisão bibliográfica, apresentando assuntos pertinentes para a elaboração do projeto, como o controlador utilizado para a interpretação dos comandos da *Plotter*, a linguagem de programação utilizada para gerar tais comandos e demais assuntos essenciais para a realização do trabalho.

No capítulo 3, foi apresentada a montagem estrutural de todo o equipamento passo a passo. Além disso, foi discutido a função de cada um dos *softwares* utilizados e a função e execução de cada um deles.

No capítulo 4 encontra-se os resultados, que apresentam informações de como a *Plotter* se comportou, os resultados obtidos, as modificações realizadas para melhorar seu funcionamento e o resultado final, apresentando a placa já impressa.

No capítulo 5 é apresentado a conclusão, que conclui o projeto e apresenta suas considerações finais com base nos resultados obtidos. Discute, ainda, se os objetivos foram alcançados, as dificuldades encontradas e possíveis melhorias para projetos futuros.

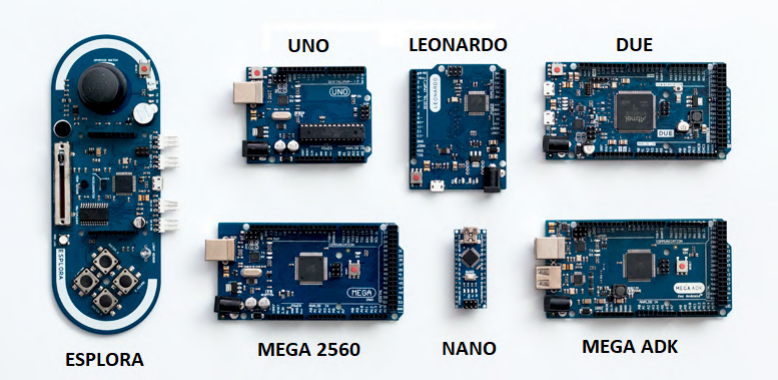
# 2 REVISÃO BIBLIOGRÁFICA

Para a realização deste trabalho, diversos conceitos e ferramentas foram utilizados, sem os quais não seria possível a execução do projeto. A definição de CNC, a linguagem de programação utilizada, o microcontrolador e a forma de acionamento dos motores de passo são alguns dos assuntos pertinentes ao projeto.

## 2.1 Microcontrolador Arduino

O microcontrolador Arduino é uma placa eletrônica de código aberto baseada em hardware e software utilizada para ler entradas diversas e gerar saídas de acordo com uma lógica pré-estabelecida (ARDUINO, 2019). Devido ao seu fácil acesso e programação, é amplamente utilizado para realizar pequenos projetos de automação através de sensores e atuadores. Apesar de existirem diversos modelos, como mostrado na Figura 2.1, a placa Arduino pré-montada inclui um controlador (ATmega8, ATmega168, ATmega328, ATmega1280 ou ATmega2560, dependendo o modelo), além de uma quantidade de pinos analógicos e digitais, que são utilizados para conectar os componentes que farão o sensoriamento do ambiente e a atuação sobre o mesmo. Além disso, também possui um conector de energia que fornece tensão de baixa intensidade para alimentar os componentes e a própria placa, um conector USB, utilizado para fazer a comunicação entre a placa e o computador, além de demais componentes como regulador de tensão, oscilador e gerador de *clock*, que não são diretamente interagidos pelo usuário mas que são essenciais para o correto funcionamento do sistema (OPENSOURCE, 2019).

Figura 2.1 – Modelos de Arduino




Fonte: (FILIPEFLOP, 2019)

O microcontrolador Arduino possui seu próprio ambiente de desenvolvimento, chamado Arduino IDE. A linguagem de programação utilizada para implementar o programa é baseada

na linguagem C++, que é muito tradicional e conhecida. Diversas bibliotecas estão disponíveis para a plataforma afim de simplificar o controle e supervisão dos diversos sensores e atuadores existentes no mercado suportados pelo controlador (CIRCUITAR, 2018).

Ele possui basicamente duas estruturas de programação: *setup()* e *loop()*. A função *setup()* é chamada quando o programa é iniciado. Nela, são definidas as variáveis, os pinos utilizados, as bibliotecas, dentre outras configurações que serão utilizadas durante todo o tempo de execução do programa. A função *loop()* faz com que o código implementado seja repetido consecutivamente enquanto a placa estiver ligada. Nela estará presente toda a implementação de leitura de sensores, acionamento de atuadores, dentre outras funções (ARDUINO, 2019).

## 2.2 Comando Numérico Computadorizado (CNC)

O Comando Numérico Computadorizado (CNC) é uma ferramenta eletrônica capaz de entender e processar comandos numéricos que possuem as informações geométricas e dimensionais de determinada peça ou projeto. Após processar os dados, o sistema aciona motores e outros atuadores através de pulsos elétricos, realizando a operação de impressão ou usinagem sem intervenção do operador, possibilitando a automação de processos (PEREIRA, 2003).

O conceito de máquina CNC não é recente. Já na década de 40, John C. Parsons, que trabalhava em uma empresa de produção de rotores de helicópteros, teve a ideia de ligar um computador da época a uma máquina. Por meio de cartões perfurados, foi possível transmitir as instruções para a máquina e realizar o seu controle.

Em 1949 a Força Aérea Americana contratou Parsons para fazer um estudo da aplicação dos sistemas de comando numérico para acelerar a produção de equipamentos de seus aviões e mísseis. O estudo resultou em uma fresadora de 3 eixos que se tornou um protótipo das máquinas de comando numérico atuais, surgindo o termo comando numérico.

No final da década de 50 a utilização de equipamento de comando numérico teve grande importância no mercado, principalmente no setor de aviação. Estas indústrias passaram a fabricar mais equipamentos em um tempo menor, conseguiram diminuir custos e alcançar grande precisão nos produtos fabricados. A partir daí, os equipamentos de comando numérico foram se tornando cada vez mais populares e indispensáveis em vários setores do mercado (PROTOPTIMUS, 2017).

Alguns fatores possibilitaram o rápido desenvolvimento destes equipamentos, dentre eles:

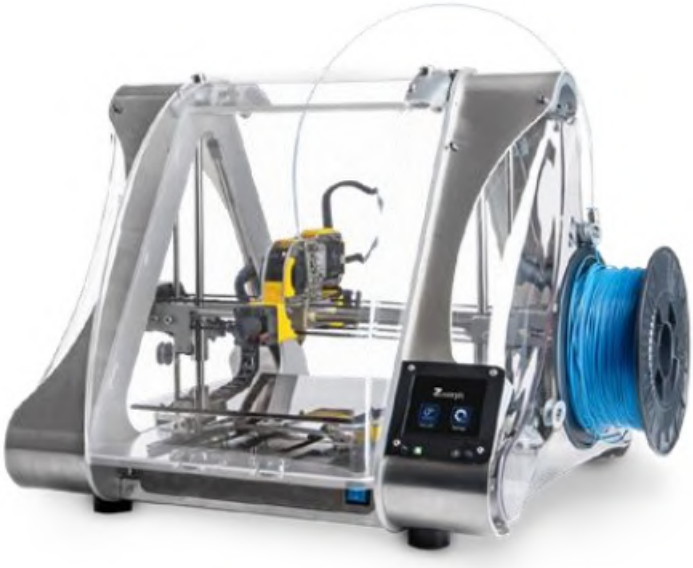
- Padronização do código *G Code*.
- Surgimento e popularidade do Desenho Assistido por Computador (CAD, do inglês *Computer Aided Design*).
- Aumento do poder de processamento de mini-computadores.

Alguns dos diferentes tipos de máquinas CNC são as impressoras 3D, os tornos e as fresadoras CNC (PROTOPTIMUS, 2017).

### 2.2.1 Impressoras 3D

As primeiras impressoras 3D trabalhavam no modo de subtração, ou seja, retiravam material de um bloco maciço e assim esculpiam o objeto desejado. O objeto é gerado camada por camada perante os modelos da peça obtidas pelo modelo 3D. Posteriormente, seu funcionamento passou a se basear na adição de material, utilizando, assim, menos material, porém tornando seu funcionamento muito mais complexo. Sua popularidade aumentou tanto que, atualmente, é possível encontrar uma infinidade de versões no mercado, para as mais diversas aplicações, a preços cada vez mais acessíveis. A Figura 2.2 mostra a impressora Zmorph 3D 2.0 SX, da fabricante Zmorph, sendo uma impressora pequena, compacta, de nível profissional, alta velocidade de impressão, e capaz de imprimir objetos de diversos materiais diferentes.

Figura 2.2 – Impressora Zmorph 3D 2.0 SX

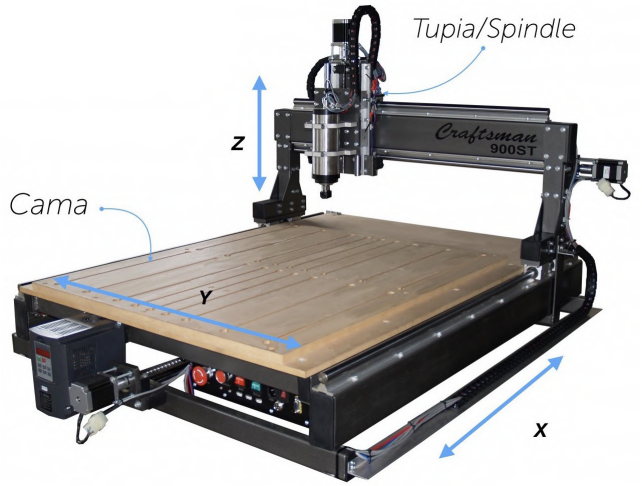


Fonte: (PROTOPTIMUS, 2017)

### 2.2.2 Fresadoras e tornos CNC

Uma fresasora CNC é um equipamento que utiliza um método subtrativo para modelar a peça, esculpindo em um bloco de material através de sua ponta de corte. A linguagem para controlá-la é *G Code*, que diz ao equipamento o movimento que a ferramenta deve realizar em coordenadas X, Y e Z. Esse controle permite operações precisas de corte em diversos materiais (madeira, plásticos, metais, etc) e em diversos formatos. Toda fresadora CNC possui pelo menos 3 eixos de movimentação, uma cama para anexar o material e uma tupia (ZMORPH3D, 2017). A Figura 2.3 mostra um modelo de fresadora CNC, já com alguns de seus componentes identificados.

Figura 2.3 – Fresadora CNC




Fonte: (ZMORPH3D, 2017)

O torno CNC é utilizado na usinagem de precisão na produção de peças cilíndricas. Ele é capaz de realizar diversos tipos de usinagem, como lixar, cortar, raspar, perfurar e deformar ferramentas para a criação de objetos que tenham simetria sobre um eixo de rotação (VERTEXUSINAGEM, 2017). Assim como a fresadora, ele utiliza a linguagem *G Code* para sua programação e os movimentos são realizados com base nos valores de X, Y e Z recebidos. A Figura 2.4 mostra um modelo de torno CNC industrial.

## 2.3 G Code

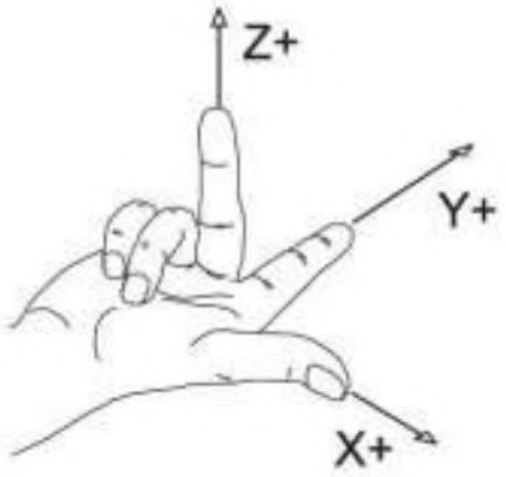
O *G Code* (ou Código G) é uma linguagem de programação utilizada principalmente por máquinas-ferramenta interpretada por meio de Comando Numérico Computadorizado. Ele descreve, de forma simples e direta, os movimentos que a máquina fará em relação ao plano de

Figura 2.4 – Torno CNC


Fonte: (ZMORPH3D, 2019)

trabalho nos eixos X, Y e Z. Esses movimentos são, geralmente, definidos pela regra da mão direita, como mostra a Figura 2.5. Com tais comandos, é possível determinar a geometria dos movimentos da ferramenta e o estado operacional do controlador, permitindo realizar operações como movimentos de corte e perfuração (DENFORD, 2000).

Figura 2.5 – Regra da mão direita




Fonte: (MACHMOTION, 2016)

Seu código é formado por blocos que consistem em instruções que descrevem o movimento. Cada bloco é constituído por palavras, que por sua vez podem possuir informações do número de sucessão, a função preparatória, a posição nos eixos X, Y e Z, a taxa de avanço e a função miscelânea. Dependendo a fabricante podem existir variantes em relação às variáveis.

O número de sucessão consiste na letra N seguida de um número inteiro com um ou mais dígitos. Ele é utilizado para identificação de linha, ou seja, para definir o local do código em que a palavra é utilizada.

A função preparatória consiste na letra G seguida de um número inteiro de dois dígitos e é utilizado para definir o modo de atuação da máquina. Para cada tipo de movimento existe um valor numérico correspondente. Alguns modos mais comuns são o *Linear Move*, definido geralmente por G1, que produzirá um movimento linear coordenado até o ponto de destino, o *Arc Move*, que produzirá um movimento circular no sentido horário e anti-horário, definido geralmente por G2 e G3, respectivamente, dentre outros. No Apêndice A é possível consultar uma lista com as funções de diversos códigos *G Code*. Cada fabricante, porém, pode possuir uma tabela de funções diferentes.

As letras X, Y e Z são seguidas de números que definem a posição final da ferramenta em cada uma das coordenadas. Eles podem ser decimais, dependendo o grau de precisão utilizada. Em sequência, os dígitos seguidos da letra F definem a velocidade da ferramenta.

A função miscelânea é utilizada para definir comandos *on/off* para a máquina. Por meio dele, é possível enviar ao programa comandos de *stop*, *reset*, *reverse*, dentre outros.

Como exemplo, podemos definir o código apresentado a seguir.

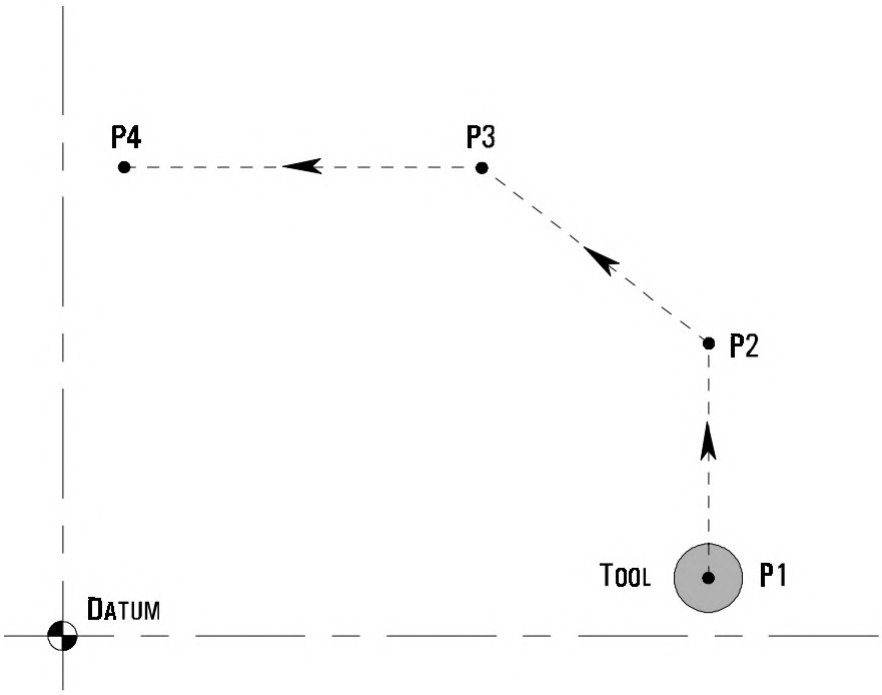
G01 Y ____;
X____ Y____;
X____;

Pode ser observado que o bloco consiste em três comandos de movimentos lineares: um em relação à coordenada Y, um às coordenadas X e Y e um à coordenada X. Definindo as coordenadas com base na regra da mão direita, temos que a ferramenta, que inicialmente se encontrará na posição P1, irá realizar os movimentos ilustrados na Figura 2.6, considerando os eixos verticais e horizontais como Y e X, respectivamente.

## 2.4 L293D

O controlador L293D é um circuito integrado utilizado para o controle de cargas indutivas como motores de passo, relés e senoides. Ele suporta tensão de alimentação de 4,5 a 7V e contém duas pontes H completas ou quatro *half H-bridges* que permitem fornecer aos motores

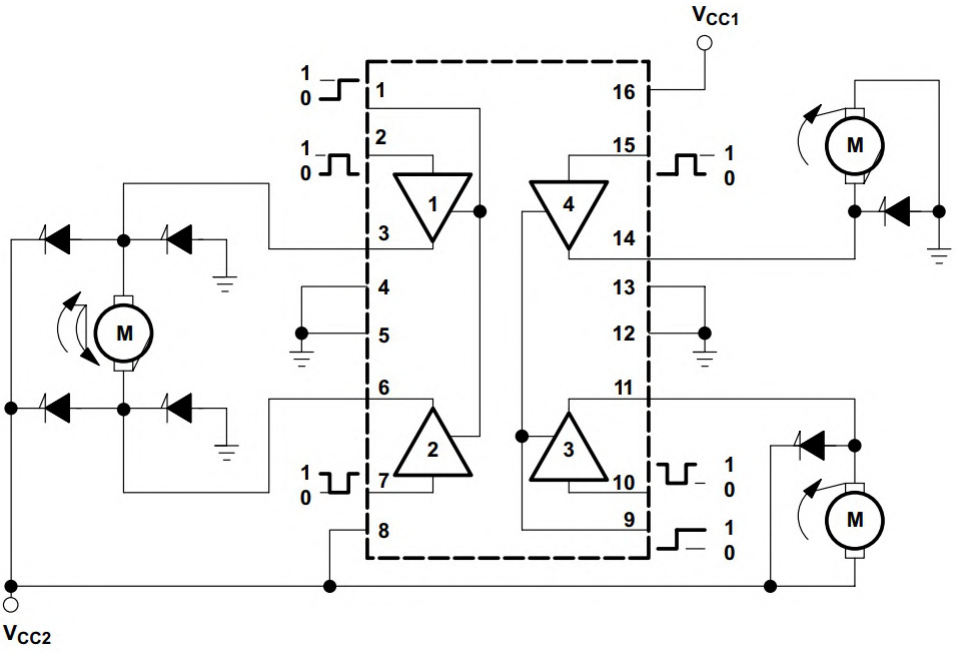
Figura 2.6 – Movimentos da máquina com base no G Code




Fonte: (DENFORD, 2000)

a corrente necessária, alterar o sentido de rotação e alterar a velocidade de rotação através se sinais de pulso (TEXAS INSTRUMENTS, 2016). A Figura 2.7 apresenta o esquemático do controlador.

Figura 2.7 – Diagrama do módulo L293D




Fonte: (TEXAS INSTRUMENTS, 2016)

Os pinos 8 e 16 fornecem a tensão para os dois pares de *drivers* presentes no sistema. Tais terminais são separados com a finalidade de diminuir a dissipação de energia no dispositivo. Cada par de *drivers* é responsável pelo controle de um motor, definindo, assim, características como seu sentido de rotação, velocidade, dentre outros. A tensão de alimentação do controlador é de 5V (máximo de 36V de pico) e a corrente máxima é de 600mA. Quando uma entrada de ativação é alta, os *drivers* associados a ele são ativados, assim como suas respectivas saídas. Da mesma forma, quando uma entrada é baixa, os *drivers* são desabilitados e suas saídas são alteradas para o estado de alta impedância.

Os pinos 2, 7, 10 e 15 são as chaves que farão o controle do motor. Através deles, sinais enviados pelo controlador Arduino serão recebidos e enviados ao motor através dos pinos 3, 6, 11 e 14, fazendo com que o motor seja acionado possuindo determinadas características de funcionamento. E, para finalizar, os pinos 1 e 9 são responsáveis por ativar ou desativar o funcionamento do controlador nos lados esquerdo e direito, respectivamente.

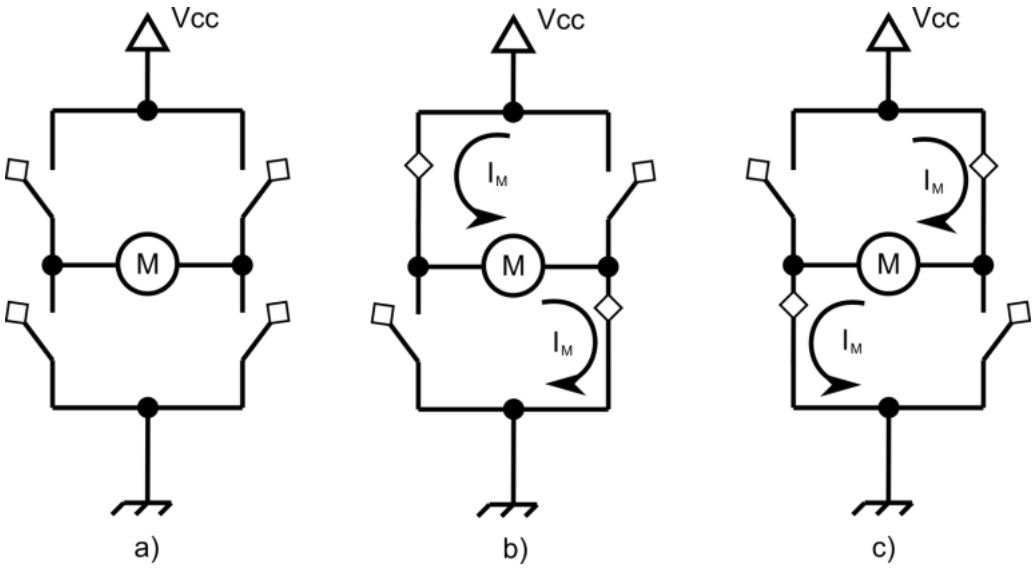
Através do esquemático representado na Figura 2.7, e considerando o pino 1 sempre ativo, pode-se concluir que, acionando o pino 2 com sinal de nível alto e mantendo o pino 7 com nível baixo, a corrente percorre certo caminho pelo circuito, fazendo com que o motor gire em determinado sentido. Caso a situação se inverta, mantendo o pino 7 em nível lógico alto e o pino 2 em baixo, o motor terá seu movimento invertido. A Figura 2.8 ilustra os sentidos possíveis para a corrente, sendo a letra A apresentando os circuitos abertos e mantendo o motor desligado, a letra B a corrente passando da esquerda para a direita pelo motor e fazendo-o rotacionar em determinado sentido, e a letra C a corrente passando no sentido oposto e fazendo-o rotacionar inversamente.

## 2.5 Motor de Passo

Motores de passo são motores de corrente contínua que fornecem a possibilidade de controle preciso de ângulo e velocidade de rotação. Eles são recomendados para equipamentos que exigem um posicionamento preciso e erro mínimo, como impressoras e equipamentos robóticos que necessitam de alto grau de precisão nos movimentos (BRITES; SANTOS, 2008). Eles possuem diversas vantagens em relação aos outros tipos de motores, dentre elas:

- Precisão no torque: as variações no torque do motor são mínimas.

Figura 2.8 – Sentidos possíveis para a corrente do motor




Fonte: (ATHOSELECTRONICS, 2019)

- Segue uma lógica digital: seu funcionamento é baseado em pulsos elétricos recebidos, que ativam sequencialmente suas bobinas e assim faz a gerar a rotação do motor.

- Alta precisão no posicionamento: o motor realizará o movimento de rotação baseado em quantidade de passos gerados. Cada passo será responsável por realizar a rotação do motor por um pequeno ângulo. Quanto mais passos forem recebidos, maior será o ângulo de rotação do motor. O ângulo que ele se movimentará por cada passo depende do motor, mas no geral o erro é pequeno e não cumulativo.

- Rápida resposta: a resposta do motor para os sinais recebidos, seja para acelerar ou desacelerar, é rápida, pois o motor se alinha rapidamente com as bobinas se que encontram energizadas.

Existem três diferentes tipos de motores de passo: os de relutância variável, de ímã permanente e os híbridos. Cada um possui suas características e são mais indicados à determinadas aplicações.

- Relutância Variável: este tipo de motor possui um estator com enrolamentos e um rotor com múltiplos dentes. Ao passar corrente por um dos enrolamentos, seus pólos ficam magnetizados e isso faz com que os dentes do rotor sejam atraídos para os pólos do estator, gerando o movimento no eixo.

- Ímã Permanente: os motores de ímã permanente são mais baratos, simples e possuem baixa resolução. Seu rotor é construído por ímãs e não possui dentes. A vantagem deste tipo de motor é o fato de ter um campo magnético permanente que, ao somar com o campo magnético das bobinas, gera um torque maior na partida.

- Motor híbrido: este tipo de motor mistura a mecânica mais sofisticada do motor de relutância variável com a potência do motor de ímã permanente, gerando um motor com grande torque e elevada precisão nos passos, podendo variar entre 3,9 e 0,9 graus.

Uma característica importante do motor de passo é o número de posições que o eixo do motor pode ocupar ao final de uma volta completa de 360 graus. Esta informação pode ser encontrada na documentação do motor, mas também pode ser calculada pela Fórmula 2.1.

$$A_P = \frac{360}{P} \tag{2.1}$$

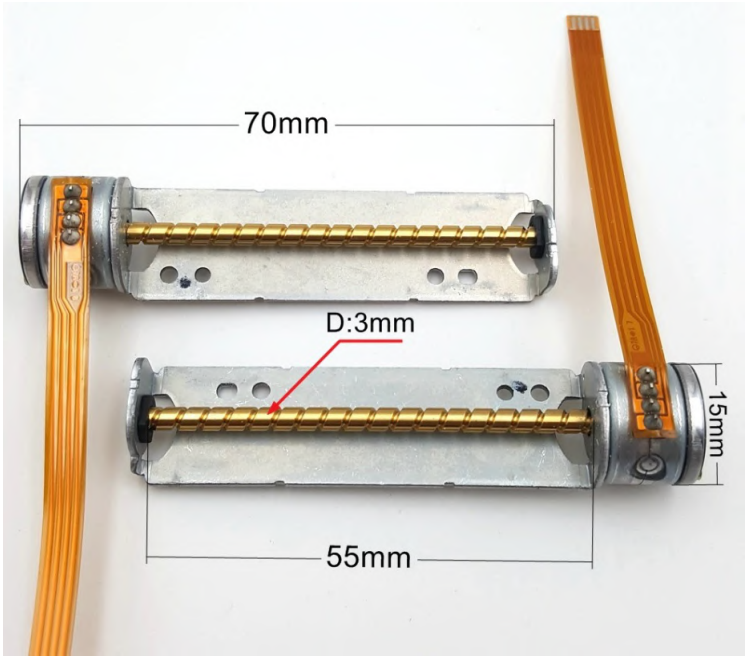
Onde $A_P$ é o ângulo de passo e P é o número de passos necessário para se completar uma rotação completa. Quanto menor for o ângulo de passo, mais preciso será o motor.

O motor de passo utilizado neste projeto, encontrado nos *drivers* de CD, é semelhante ao mostrado na Figura 2.9. O motor possui uma estrutura cilíndrica de 15mm de diâmetro por 15mm de comprimento. Seu eixo possui 3mm de diâmetro e aproximadamente 55mm de comprimento.

O motor é bipolar, de ímã permanente e possui 2 pares de fios. Cada par de fio corresponde à uma bobina. A tensão de alimentação pode variar de 3V à 6V e a corrente é de cerca de 500mA. Para completar uma volta completa, o motor precisa de 20 passos. Ou seja, de acordo com a Fórmula 2.1, o ângulo de passo é de 18 graus.

Uma outra característica importante a se observar é a quantidade de passos necessários ao motor para que a ferramenta seja deslocada em 1mm. Esta característica depende principalmente das características construtivas do eixo deslizante. Uma forma de se fazer o cálculo é enviar uma quantidade de passos definida ao motor, e verificar o quanto a ferramenta se deslocou. A partir destes dados, é possível obter o valor de passos por milímetro. No presente trabalho, foram enviados ao motor 240 passos para verificar o deslocamento da caneta. Observou-se que a mesma se deslocou por 40mm, ou seja, para que a caneta se mova por 1mm serão necessários 6 passos do motor. Este valor foi essencial para gerar uma impressão com tamanho fiel ao projeto.

Figura 2.9 – Motor de passo utilizado no projeto e suas dimensões




Fonte: (ALIEXPRESS, 2019)

## 2.6 Servomotor SG90

Um servomotor é um dispositivo elétrico capaz de realizar movimentos com grande precisão. Após atingir determinada posição desejada, possui ainda a capacidade de manter sua posição, mesmo sofrendo uma força de sentido oposto. Em geral, os servomotores de corrente contínua possuem como componentes internos os materiais listados a seguir. A Figura 2.10 ilustra tais componentes internos.

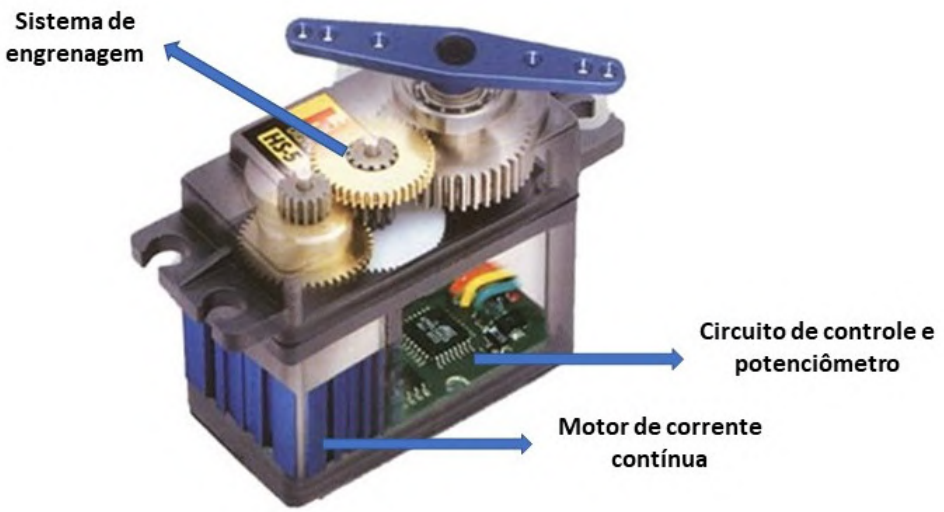
- Circuito de Controle - Responsável por receber os sinais do potenciômetro ou de um gerador de PWM e enviar sinais para acionamento do motor de acordo com a posição pré-estabelecida.

- Potenciômetro - Dispositivo responsável por produzir uma tensão correspondente ao ângulo desejado do motor. Tal tensão variável também pode ser produzida por um gerador de sinal PWM, que gera pulsos com *Duty Cicle* proporcional à velocidade desejada do motor, oferecendo um controle mais preciso.

- Motor de Corrente Contínua - Responsável por gerar o torque que farão as engrenagens e o eixo principal do motor se movimentarem.

- Sistema de Engrenagem: Possuem a função de reduzir a rotação transferida para o eixo principal, fazendo, assim, com que o motor passe a transferir maior torque ao eixo.

Figura 2.10 – Componentes internos do servomotor




Fonte: (ENGLEANDROALVES, 2012)

O servomotor utilizado neste projeto foi o SG90. Ele é um servo de alta qualidade e excelente para aplicações em aeromodelismo ou em projetos de robótica com Arduino, PIC, Raspberry e entre outros.

# 3 METODOLOGIA

Para a montagem estrutural do equipamento, foram utilizados dois *drivers* de CD. Tais equipamentos são facilmente encontrados em computadores *desktops* antigos que são frequentemente descartados ou encontrados a preços acessíveis. Apenas na Universidade Federal de Lavras, estão armazenadas dezenas de máquinas *desktops* já não mais utilizados pela Universidade e que podem ter suas peças reutilizadas pela comunidade acadêmica.

Os materiais utilizados são mostrados na Tabela 3.2 e foram pensados de forma a utilizar de forma mais consciente possível os recursos disponíveis e reutilizando equipamentos já não utilizados pela Universidade, auxiliando na sua reciclagem e criando um equipamento útil para grande parte da comunidade acadêmica.

Tabela 3.1 – Materiais Necessários

| Quantidade | Material / Componente |
|---|---|
| 2 | *Drivers* de CD |
| 1 | Servomotor TowerPro SG90 |
| 1 | Microcontrolador Arduino Uno ou similar |
| 2 | Controladores L293D |
| 1 | Buzzer 5V |
| 1 | *Protoboard* |
| 1 | Ferro de solda |
| * | Demais materiais para fixação/montagem |

Fonte: Do Autor (2019).

Caso tais componentes sejam comprados no mercado, os preços médios encontrados no período de realização deste projeto se encontra na Tabela 3.2, resultando em um total de cerca de R$ 97,00.

Tabela 3.2 – Preços dos materiais encontrados no mercado brasileiro

| Material (unidade) | Preço (R$) |
|---|---|
| *Driver* de CD | 20,00 |
| Servomotor TowerPro SG90 | 11,00 |
| Microcontrolador Arduino Uno | 24,00 |
| Controlador L293D | 9,90 |
| Buzzer 5V | 2,50 |
| *Protoboard* | 14,90 |
| Ferro de solda | 15,00 |

Fonte: Google Shopping (2019).

A construção de toda estrutura física é descrita a seguir, assim como a utilização dos *softwares* utilizados.

## 3.1 Estrutura física

Primeiramente, foi necessário desmontar os dois *drivers* de CD (DOBITAOBYTE, 2018). Eles são semelhantes ao mostrado na Figura 3.1 e sua desmontagem foi feita cuidadosamente a fim de não danificar sua estrutura e seus componentes internos.

Figura 3.1 – Driver CD

Fonte: Do Autor (2019).

Após a desmontagem, identificou-se o motor de passo, o circuito de controle, dentre outros componentes do equipamento. O motor de passo, juntamente com sua estrutura metálica responsável pelo deslizamento do componente, foram demovidos da carcaça e os componentes foram separados.

Separados os componentes necessários dos dois *drivers* de CD, soltou-se, cuidadosamente, seus quatro terminais. Com o auxílio de ferro de solda e estanho, fixou-se um *jumper* em cada um dos quatro terminais do motor. O processo de solda deve ser feito com cuidado e precisão, visto a delicadeza dos componentes.

Os *drivers* podem possuir pequenas diferenças entre eles, variando conforme a marca e modelo. No presente projeto, foram utilizados dois modelos distintos, diferenciando-se quanto

à posição do motor e a haste de deslizamento. Suas características de funcionamento, porém, não apresentam qualquer diferença.

Para a montagem da estrutura física na qual serão fixados os componentes, utilizou-se a própria carcaça dos *drivers* de CD. Posicionou-se os dois formando um ângulo de 90º entre eles e uniu-se em formato de "L". Para a fixação, foi descartada a possibilidade de soldagem pelas chances de deformação da mesma pelo fato de a espessura do metal ser consideravelmente fina. Desta forma, concluiu-se que a melhor forma de unir sem avarias seria utilizar uma pequena chapa de metal de aproximadamente 2cm por 10cm para unir as duas estruturas através de rebites. Fazendo o procedimento nos dois lados, a estrutura se mostra fixa o suficiente para realização do projeto.

Estando a estrutura física devidamente fixada, realizou-se quatro pequenos furos em cada uma das duas suas faces internas, batendo com os furos dos componentes metálicos do motor. O motor que teria a caneta fixada a ele deve ficar na posição horizontal, enquanto o que ficaria rente ao chão teria a posição invertida. Assim, cada motor foi responsável pela movimentação da caneta em um dos eixos X ou Y.

Os componentes do *driver* foram fixados a uma certa distancia da face, de forma que o motor não ficasse diretamente em contato com a mesma e que fosse possível o deslizamento da base sem obstruções. Para isso, foram utilizados parafusos com 3cm de comprimento, juntamente com calços anti-vibração de silicone e porcas. Os calços deram um amortecimento ao conjunto, impedindo que os motores façam a estrutura vibrar com seus movimentos, fazendo, inclusive, com que o sistema se torne mais silencioso.

Como parte de seus componentes internos, o *driver* de CD possui um ou mais pequenos ímãs juntamente ao leitor de mídia, com cerca de 0,7cm a 1,5 cm. Tais componentes foram reservados e utilizados posteriormente.

A caneta utilizada para realizar o desenho faria o movimento de subida e descida através do movimento gerado pelo servomotor. Ela deslizaria por dentro de uma estrutura cilíndrica, que deve ser fixada ao lado do motor de passo. Esta estrutura não pode ser excessivamente justa, causando o comprometimento do movimento da caneta, e nem excessivamente larga, causando o movimento não apenas vertical, mas horizontal da ferramenta, comprometendo o resultado final. Para fazer este papel, foi utilizado um canudo com um diâmetro suficiente para ser possível inserir a caneta em seu interior. Foi realizado o ajuste a fim de deixar a caneta mais justa.

Uma vez ajustada dentro do canudo, inseriu-se um pino na caneta que fez o contato dela com o eixo do motor. Para isso, foi rosqueado um pequeno parafuso no local onde será feito o contato. O conjunto do canudo com o servomotor foi fixado em uma pequena plataforma reta, podendo ser de acrílico ou PVC. Em seu eixo foi presa uma pequena haste que possuía um comprimento suficiente para, ao girar, ter contato com o parafuso da caneta e a fazer se deslocar para cima.

Esta estrutura foi fixada ao componente deslizante do motor. Porém, caso seja fixado diretamente nele, não haveria espaço o suficiente para deslocar a placa abaixo da caneta. Para isso, a caneta foi distanciada a cerca de 5cm do motor, ficando, assim, próxima ao centro da estrutura metálica. Para fazer este distanciamento, foi fixado ao motor uma caixinha de apontador de lápis. Seu fundo foi preso por um parafuso, e a outra extremidade encaixou o servomotor de forma quase perfeita.

Para servir como base para a impressão, foi utilizada uma placa metálica de cerca de 15x10cm. Ela foi recordada da carcaça de um terceiro *driver* que não foi utilizado para este projeto, porém pode ser utilizada qualquer superfície reta. Por ser metálica, após colocar sobre ela a placa de circuito para a impressão, basta colocar um ímã em cada ponta para fixar e não permitir que a mesma se desloque durante a impressão.

Para a conexão dos componentes aos controladores, foi utilizada uma *protoboard*. Desta forma, não seriam necessárias diversas soldas e a montagem se tornaria mais rápida, segura e dinâmica.

O motor de passo contido nos *drivers* possui duas fases e quatro terminais, sendo estes conectados aos pinos 3, 6, 11 e 14 do módulo L293D. Através destes pinos, o sinal foi recebido pelo motor de passo e lhe fez funcionar com determinadas características.

Os terminais 1, 8, 9 e 16 do módulo são responsáveis por alimentar os motores conectados à ele. Como os controladores podem ser alimentados com tensão de 5V, seus terminais podem ser conectados diretamente à fonte de 5V do Arduino. Como opção, também podem ser alimentados por uma fonte externa de mesma tensão.

Os terminais 2, 7, 10 e 15 foram conectados às entradas digitais do Arduino, enquanto os terminais 4, 5, 12 e 13 foram conectados ao neutro do Arduino (GND). Desta forma, o módulo L293D pôde receber os sinais do Arduino e fazer o controle dos motores de acordo com uma lógica pré-estabelecida.

Um *buzzer* foi adicionado ao projeto como forma de sinalizar ao usuário a operação que está sendo feita na *Plotter*. Sempre que um projeto for submetido para impressão, o *buzzer* emite dois sinais sonoros curtos de determinada frequência, com o objetivo de alertar o usuário que ela entrou em funcionamento. Ao finalizar a impressão, um sinal sonoro longo de mesma frequência indica que a mesma finalizou seu trabalho.

Por fim, foi conectado o servomotor. Seu fio vermelho foi conectado à fonte de 5V do Arduino, o fio preto foi conectado ao neutro, e o fio amarelo foi conectado à alguma entrada analógica do Arduino. No Apêndice B é possível consultar o esquemático das conexões elétricas dos componentes utilizados para o projeto.

O controlador e toda a fiação foram fixados na parte traseira da *Plotter*. A escolha de deixar o controlador e sua fiação à mostra foi pelo fato de constantemente ser necessário algum tipo de manutenção ou alteração nos fios, tornando assim o processo mais rápido. Porém, podem também ser fixados no interior da carcaça metálica, fazendo nela pequenas aberturas por onde devem passar os cabos de alimentação. Desta forma o controlador ficaria mais protegido e seria visualmente mais agradável.

## 3.2 Softwares

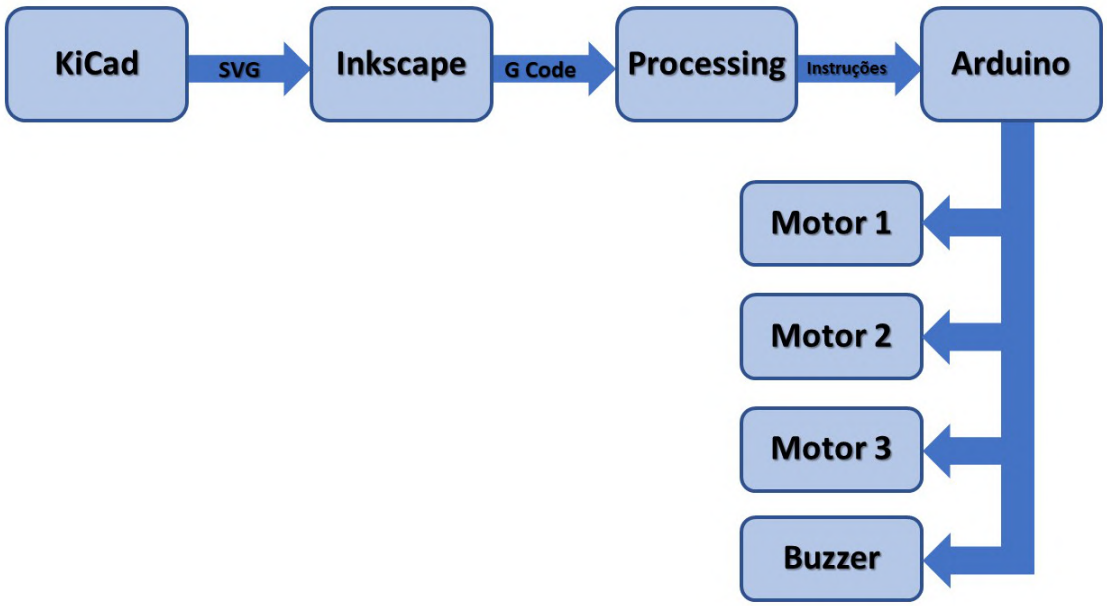
Para o processo de projetar a placa até o momento da impressão, foram utilizados quatro *softwares* diferentes, cada um com sua finalidade. A Figura 3.2 representa o esquemático do processo.

### 3.2.1 KiCad

O *KiCad* é um *software* de código aberto multiplataforma que permite a criação de diagramas eletrônicos, ilustrações e visualizações 3D de placas de circuito impresso (KICAD, 2019). Ele possui quatro ambientes principais, sendo eles:

- KiCad: Gerencia o projeto.
- Eeschema: Editor do esquemático da placa, incluindo seus componentes e definindo suas conexões.
- Pcbnew: Editor de *layout* da placa, distribuindo os componentes pela placa e definindo os trilhos de conexão entre eles.

Figura 3.2 – Esquemático dos *softwares* utilizados no projeto




Fonte: Do autor (2019).

- GerbView: Visualizador do projeto em formato Gerber.

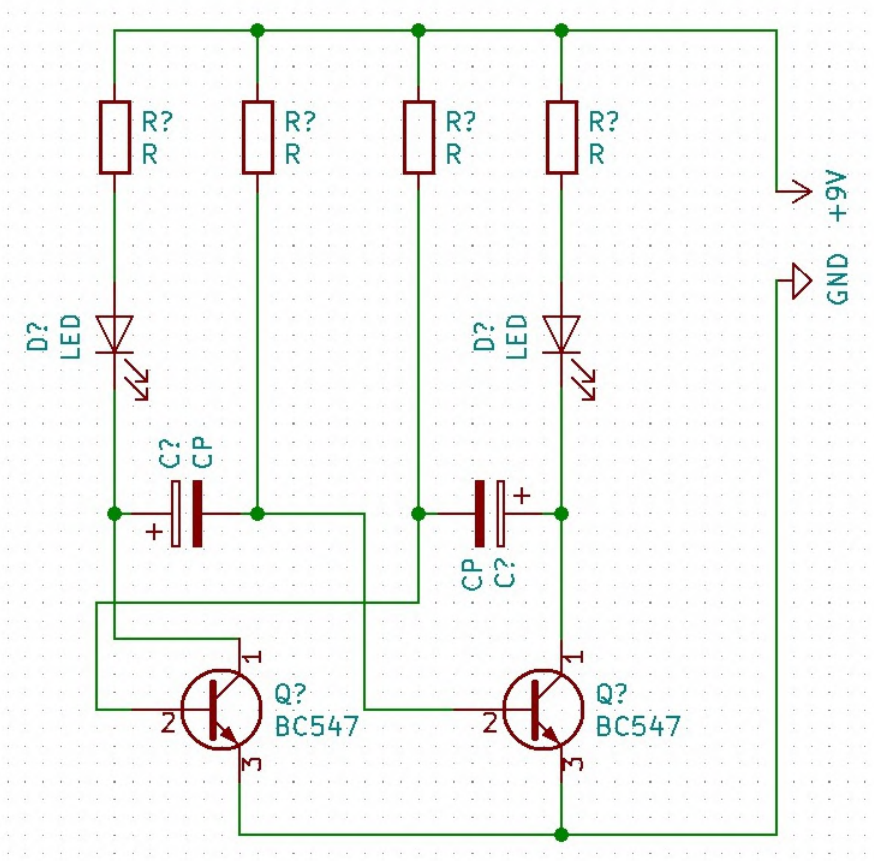
Para o presente projeto, foram utilizados os ambientes *Eeschema* e *protoboard* para elaboração da placa. Como exemplo para testar o funcionamento da *Plotter*, foi feito o projeto de uma placa para um circuito *Flip-Flop*, que consistirá em basicamente acender dois LED's alternadamente.

A placa conterá dez componentes, sendo quatro resistores, dois LED's, dois capacitores e dois transistores BC547. Dois dos resistores serão de 480 *ohms* e dois de 10 *kilohms*, enquanto os capacitores serão de 100 microfarads.

No *KiCad*, os componentes foram adicionados ao ambiente *Eeschema* e foram definidas suas conexões. Nesta etapa, o trilho apenas indica as conexões, e não o *layout* final do mesmo. Este processo será realizado na etapa seguinte. A Figura 3.3 ilustra o esquemático da placa no ambiente *Eeschema*, com todas as conexões definidas.

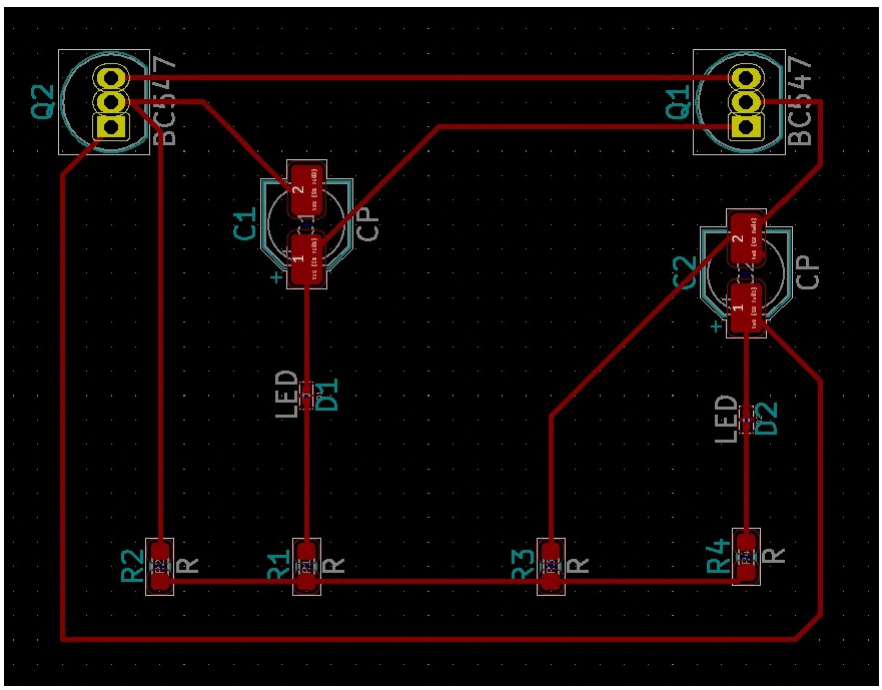
Finalizado o esquemático da placa, o projeto foi salvo e aberto do ambiente *Pcbnew*. Nesta etapa, os componentes foram distribuídos da forma como serão fixados na placa, assim como o desenho dos trilhos. Lembrando que, por ser uma placa com apenas uma camada, os trilhos não podem se cruzar, a menos que seja necessária uma conexão entre eles. A Figura 3.4 ilustra o projeto no ambiente *Pcbnew*, com os locais dos componentes e os trilhos definidos.

Figura 3.3 – Projeto da placa no ambiente *Eeschema*




Fonte: Do autor (2019).

Figura 3.4 – Projeto da placa no ambiente *Pcbnew*




Fonte: Do autor (2019).

Finalizado o projeto no referido ambiente, ele foi exportado com extensão SVG. Esta linguagem realiza a conversão do projeto em um grupo de vetores. Cada linha reta será formada por um ou mais vetores independentes. Por conta disso, formas circulares irão gerar um arquivo muito maior e mais complexo, visto que as curvas serão convertidas em dezenas ou centenas de pequenos vetores.

Após a exportação, o arquivo foi aberto e lido pelo *software Inkscape*.

### 3.2.2 Inkscape

O *Inkscape* é um *software* de código aberto multiplataforma editor de gráficos vetoriais que é amplamente utilizado em diversos segmentos da indústria, como engenharia, publicidade, desenhos animados, dentre outros. O desenho com vetores é um dos mais indicados para criação de logotipos e ilustrações que exigem alta escalabilidade (INKSCAPE, 2019).

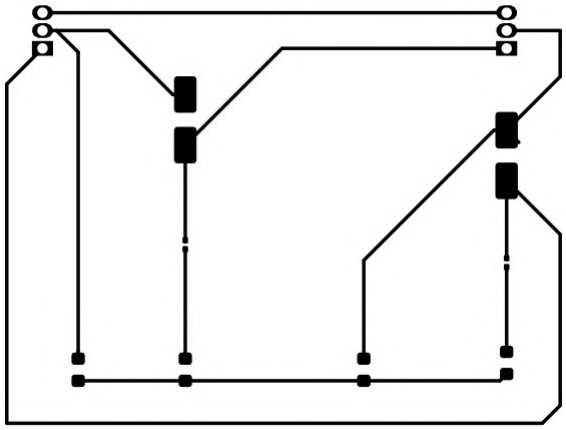
Uma das suas principais características é oferecer a capacidade de trabalhar com arquivos SVG e exportá-los em formato *G Code*. Tal arquivo foi lido e interpretado para gerar os comandos da *Plotter*.

Após importado o arquivo em formato SVG, o *Inkscape* permite que o projeto da placa seja editado de forma livre. Como a extensão SVG é baseada em desenho com vetores, é possível, por meio dele, visualizar todos os vetores existentes no projeto da placa. Baseado nesta visualização, é possível fazer algumas modificações no projeto com o objetivo de facilitar a impressão da placa e minimizar possíveis distorções. A Figura 3.5 ilustra o projeto aberto no *Inkscape*, sem modificações.

As posições X e Y que a ferramenta deverá percorrer serão definidas de acordo com a posição na página onde o projeto estará desenhado. Por isso, é interessante que a página criada possua o tamanho máximo de impressão possível pela *Plotter*, para que não corra risco do projeto da placa exceder o tamanho máximo permitido. Logo, para o presente projeto utilizando *drivers* de CD, foi criada uma página de área 40x40mm. Apesar de ser possível a impressão chegar a 50mmx50mm, foi adotado um tamanho menor por segurança.

A partir da página criada, as posições X e Y de cada segmento de impressão foram definidas. Os pontos zero de cada um dos eixos foram automaticamente setados na metade de cada eixo da folha. Ou seja, para uma página de 40x40mm, o eixo X se iniciará em X=-20mm e terminará em X=20mm, tendo um tamanho total de 40mm. A mesma coisa será para o eixo Y. A Figura 3.6 ilustra as posições da folha em que os valores de X e Y serão positivos e negativos,

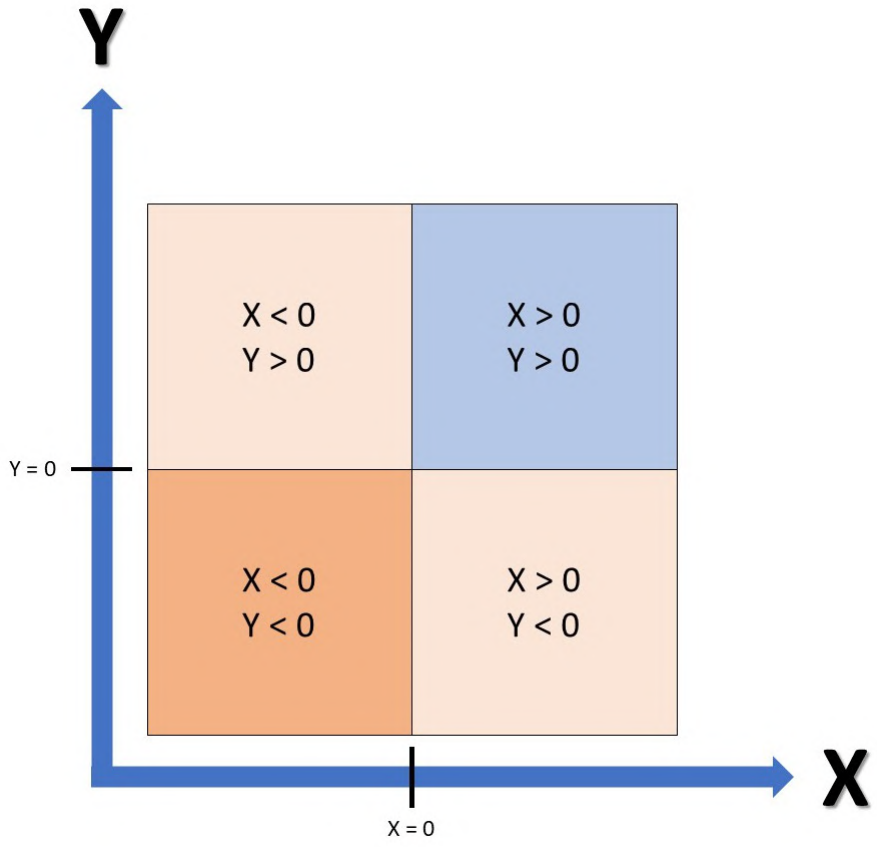
Figura 3.5 – Projeto da placa sem modificação no *Inkscape*



Fonte: Do autor (2019).

assim como seus pontos zero. Tal fator foi importante para interpretar os comandos do arquivo *G Code*.

Figura 3.6 – Valores dos eixos X e Y na página do *Inkscape*




Fonte: Do autor (2019).

Outras configurações de impressão podem ser definidas por este programa, como os ângulos em que o servomotor irá trabalhar para fazer com que a ferramenta suba e desça, o *delay* que existirá após a ferramenta subir e descer e a velocidade tanto de impressão nos eixos X e Y quanto de subida e descida da ferramenta. Todas essas informações foram utilizadas ao gerar o arquivo *G Code*.

Por ser um projeto bidimensional, o próprio programa exporta o *G Code* desconsiderando o eixo Z. Ele também entende que alguma ferramenta fará um movimento de subida e descida para ter contato com a placa e também irá gerar os comandos para tal. Como forma de facilitar o entendimento do código por uma pessoa com pouca experiência, ele também introduz algum comentários dentro do arquivo gerado. Esses comentários foram tratados pelo controlador a fim de não interferir nos comandos de impressão.

### 3.2.3 Processing

O *Processing* é um *software* que permite a integração da linguagem de programação, ambiente de desenvolvimento e metodologia. Ele possui uma linguagem de programação simplificada baseada em C++, que permite um iniciante em programação escrever seu próprio programa em apenas alguns minutos de estudo, mas que também permite que usuários mais avançados possam utilizar recursos mais avançados e complexos (THE MIT PRESS, 2007).

Muitas técnicas de computação gráfica podem ser discutidas, incluindo desenho vetorial, processamento de imagens e programação orientada e objetos, além de possuir a capacidade de importar/exportar arquivos em formato 2D e 3D.

Por meio dele, é possível estabelecer uma comunicação entre o computador e o microcontrolador (AUTOCOREROBOTICA, 2018). Seu funcionamento para este projeto se resume basicamente em estabelecer uma conexão com o controlador através de uma porta selecionada pelo usuário, fazer o carregamento do arquivo *G Code* a partir da pasta selecionada e fazer a leitura do arquivo linha por linha. A cada linha que possuir algum conteúdo, ele é enviado ao controlador via porta USB, utilizando a biblioteca *processing.serial*, específica para este fim. Também é possível visualizar cada linha que é enviada ao controlador em tempo real, facilitando a identificação de algum problema que possa dar durante a impressão. Por ele também é possível enviar comandos específicos à *Plotter*, como fazer a ferramenta retornar à posição *home* ou ir para qualquer outro ponto desejado.

### 3.2.4 Arduino

O Arduino é responsável por receber linha por linha do arquivo *G Code* através da porta USB, fazer as devidas modificações caso seja necessário e interpretar e enviar os comandos aos motores responsáveis pela impressão. Como ele controla todos os atuadores, nele são definidas as pinagens de cada um dos controladores e motores conectados a ele, o valor máximo e mínimo que cada coordenada pode possuir para realizar a impressão e o valor de passos por milímetro referente ao motor.

Como o arquivo gerado pelo *Inkscape* possui, além dos comandos padrões, comentários e caracteres que não serão utilizados para a impressão, eles foram identificados e desconsiderados antes de interpretar e gerar o comando de impressão aos motores. Para isso, foi implementada uma lógica para que todo e qualquer caractere que estiver entre parênteses seja ignorado, inclusive os parênteses. Como o *KiCad* transforma o projeto todo em um conjunto de vetores que produzirão um movimento linear, todos os comandos do *G Code* se iniciam com G1. Além destes, possuem os comandos iniciados com M, que são o comando para a caneta subir ou descer. Portanto, toda e qualquer linha que não iniciar com G ou M foi desconsiderada para a impressão, assim como caracteres como ponto, vírgula e barras.

O arquivo *G Code* possui a informação da velocidade de impressão e ângulos que o servomotor deve atuar para subir e descer a caneta. Entretanto, optou-se por não utilizar esta informação do arquivo, visto que tais informações devem ser características da *Plotter* em específico. O usuário que gerar o projeto não necessita saber das características construtivas do equipamento. Tais informações devem estar setadas no controlador do equipamento. Desta forma, os valores contidos no *G Code* de velocidade e ângulos de trabalho do servomotor são desconsiderados, e novos foram definidos no código do controlador.

Uma vez possuindo apenas as informações necessárias de impressão, foi analisada a posição atual da ferramenta e a nova posição que ela deveria estar de acordo com o comando recebido. No primeiro momento, a ferramenta se encontra na posição X=0.00 e Y=0.00, e esta é sua posição atual. Após os comandos de impressão começarem a ser recebidos, a ferramenta se desloca para a nova posição. Esta é, a partir de então, considerada a posição atual, e a nova posição será definida quando for recebido o próximo comando de impressão.

O motor de passo não reconhece a posição física em que a ferramenta deve ir para poder atuar sobre ela. Por isso, é realizado um cálculo baseado na quantidade de passos que ele gera para que seja deslocada uma determinada distância. A partir daí, tendo a informação da posição

atual da ferramenta e da posição final da mesma, foi calculada a quantidade de passos que o motor deve dar para atingir aquela distancia. Esta conversão de distância em passos é realizada para cada linha de comandos que chegarem ao controlador e para cada um dos eixos X e Y. Após todo o tratamento de dados, o comando de deslocamento foi dado aos motores baseado na quantidade de passos. Ao final da lógica, a próxima linha é recebida e o processo recomeça em um processo de *loop*.

# 4 RESULTADOS

Iniciando a desmontagem do *driver* de CD, foram observados os componentes internos do equipamento. A Figura 4.1 mostra sua estrutura interna.

Figura 4.1 – Componentes internos

Fonte: Do Autor (2019).

Os componentes que não foram utilizados para o projeto foram removidos, e o motor de passo foi separado, juntamente com a estrutura que o fixa. A Figura 4.2 apresenta o componente que será utilizado para a realização deste projeto já separado dos demais componentes, enquanto a Figura 4.3 mostra os fios conectados aos terminais do motor.

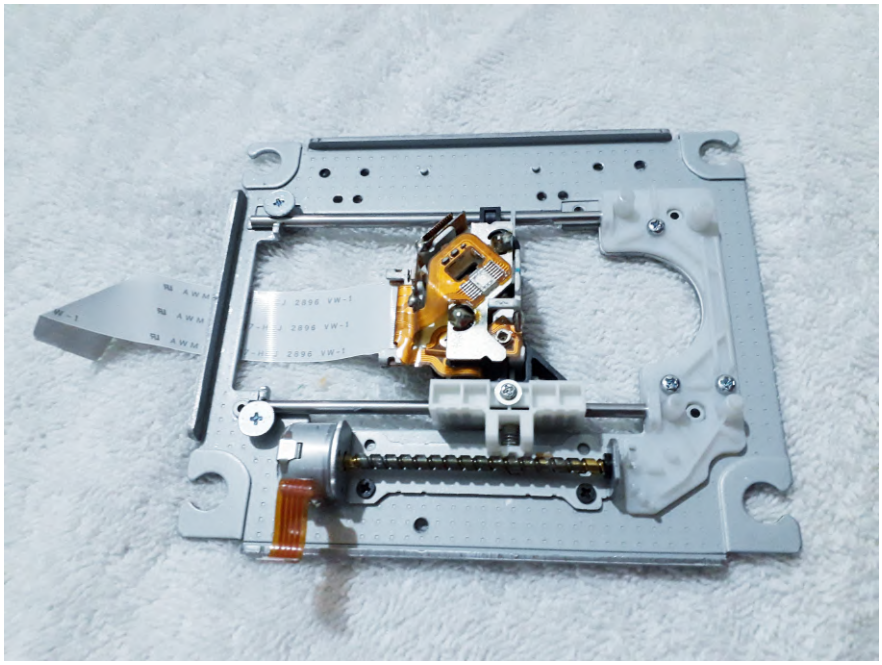
Para este projeto, foram utilizados dois modelos de *drivers*, sem alteração em seu funcionamento. A Figura 4.4 mostra os dois modelos de *driver* utilizados.

As carcaças fixadas em formato de "L"por meio de rebites é mostrado na Figura 4.5, enquanto os detalhes da fixação são mostrados na Figura 4.6.

As Figuras 4.7 e 4.8 exibem os componentes utilizados para fixação da estrutura do motor na carcaça e o motor já fixado na mesma, respectivamente. Os ímãs encontrados no interior dos *drivers* são mostrados na Figura 4.9.

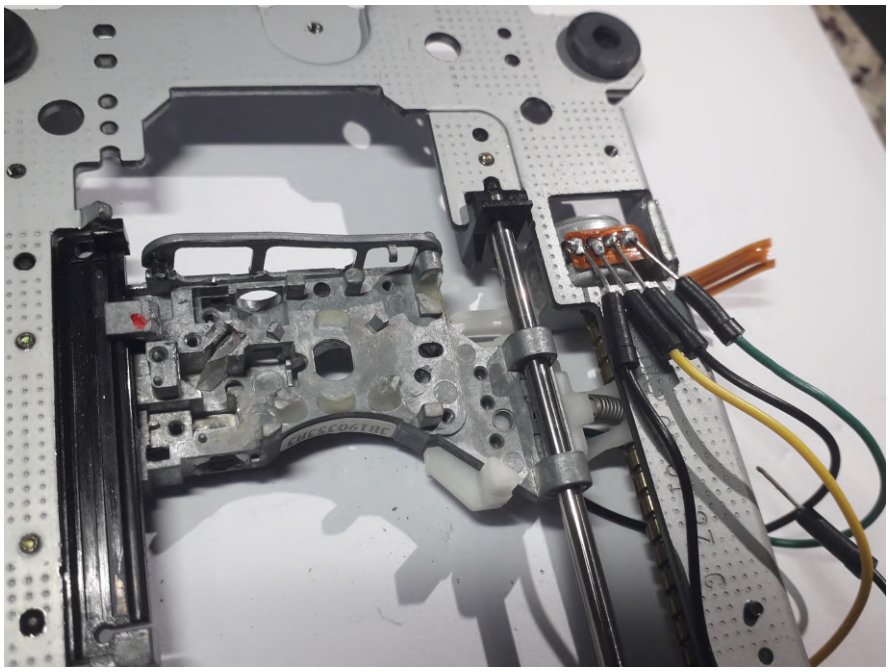
A Figura 4.10 mostra o canudo e o servomotor já ficados sobre a plataforma de acrílico. Pode-se observar, pela imagem, que ao girar o motor, a haste encostará no parafuso da caneta e fará com que a mesma se desloque para cima, fazendo o movimento de sobe e desce necessário para a impressão.

Figura 4.2 – Motor de passo anexado à estrutura deslizante



Fonte: Do Autor (2019).

Figura 4.3 – Detalhe da solda

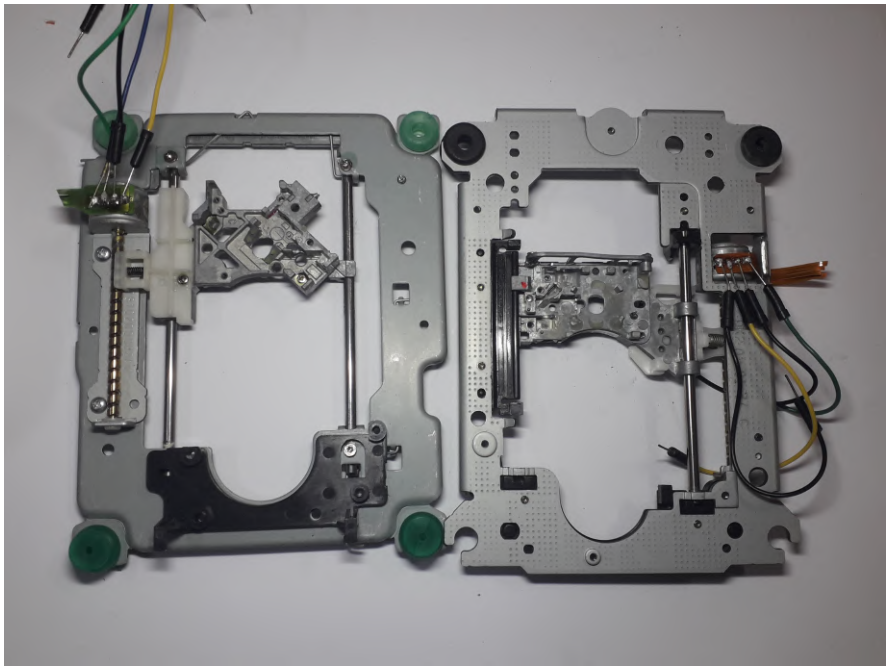


Fonte: Do Autor (2019).

A Figura 4.11 mostra a caixinha de apontador fixado na estrutura, e a Figura 4.12 mostra o servomotor já fixado na caixinha, criando o espaço necessário para a placa se deslocar sobre a caneta.

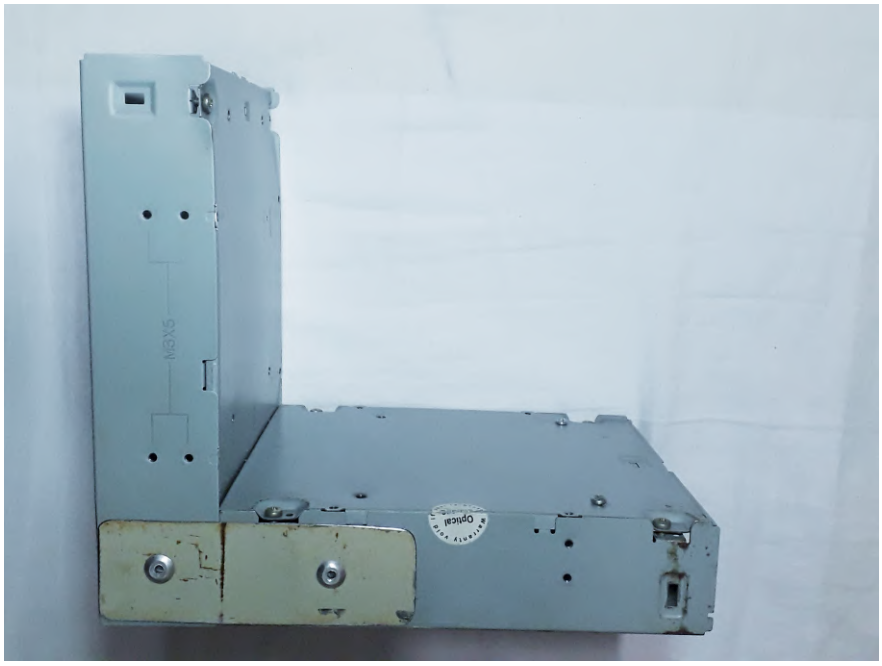
A Figura 4.13 mostra a placa metálica da base já fixada na *Plotter* e os ímãs sobre ela.

Figura 4.4 – *Drivers* desmontados



Fonte: Do Autor (2019).

Figura 4.5 – Estrutura fixada



Fonte: Do Autor (2019).

Com base do esquemático elétrico do projeto, as conexões dos controladores ao Arduino foram realizadas. A Figura 4.14 demonstra as conexões dos módulos ao Arduino já realizadas na *protoboard*.

Figura 4.6 – Detalhe da fixação

Fonte: Do Autor (2019).

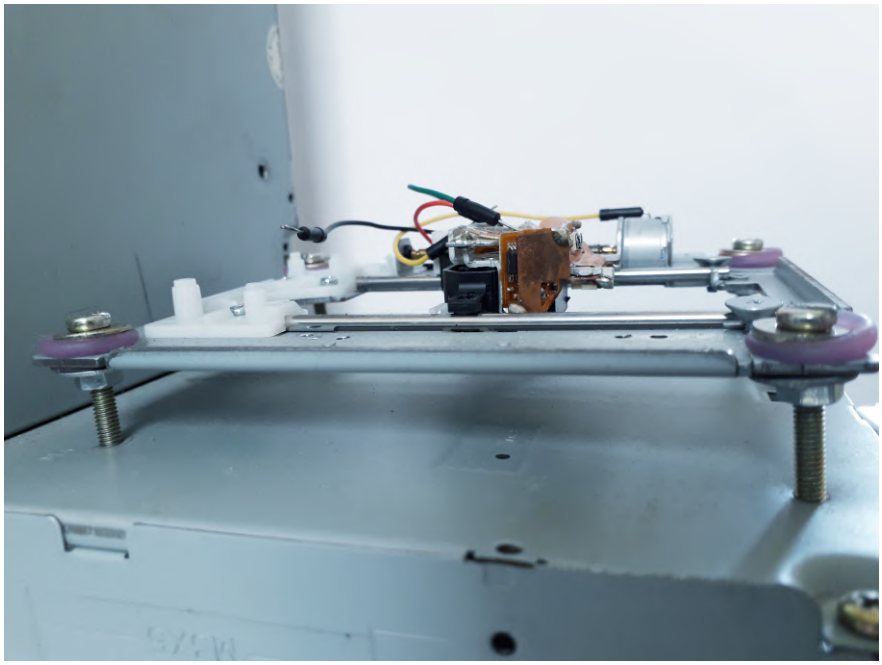
Figura 4.7 – Componentes para fixação do *driver*

Fonte: Do Autor (2019).

A montagem final, com todos os componentes fixados e conexões elétricas realizadas, pode ser observada na Figura 4.15.

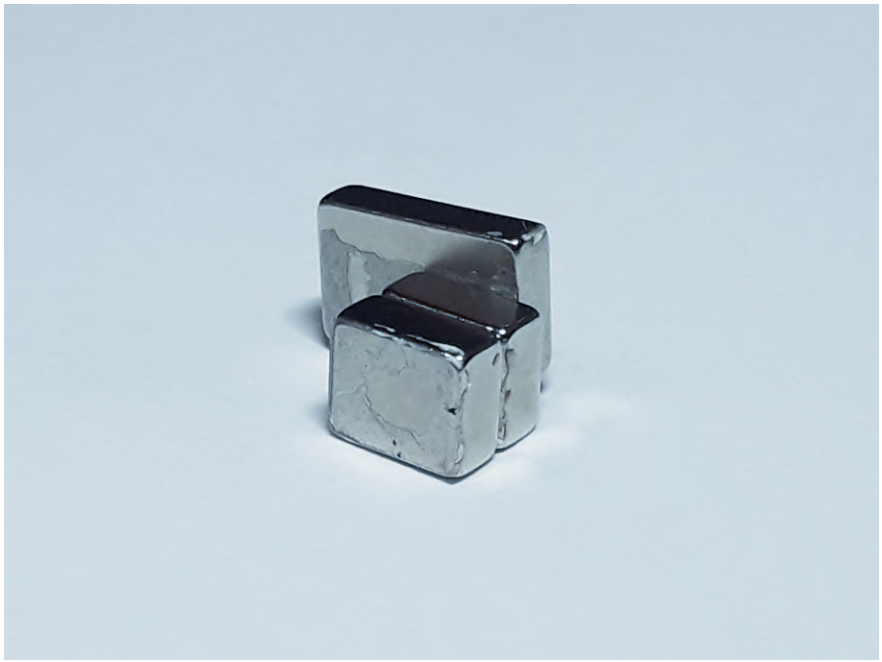
Através de testes de impressão realizados, foi definido que o ângulo para que o servo-motor suba a caneta é de 90º, e o para descer é de 20º. Além disso, a velocidade definida foi

Figura 4.8 – Motor fixado na estrutura final



Fonte: Do Autor (2019).

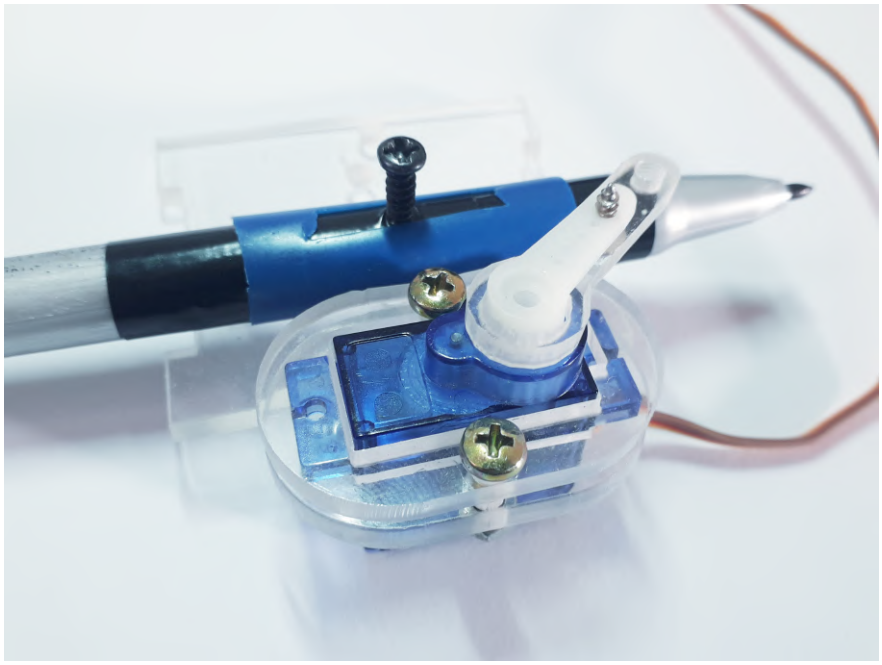
Figura 4.9 – Ímãs retirados do *driver* de CD



Fonte: Do Autor (2019).

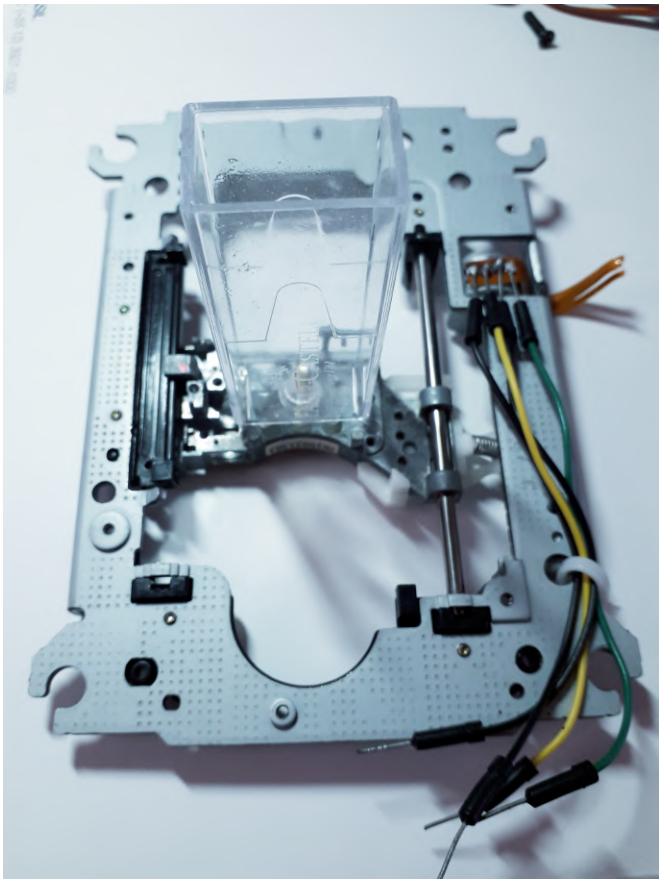
de 250 rotações por minuto. Velocidades inferiores tornaram a impressão excessivamente lenta, sem aumento perceptível da qualidade da impressão. Valores superiores dificultam a caneta em manter contato direto com a placa.

Figura 4.10 – Caneta e servomotor fixados lado a lado



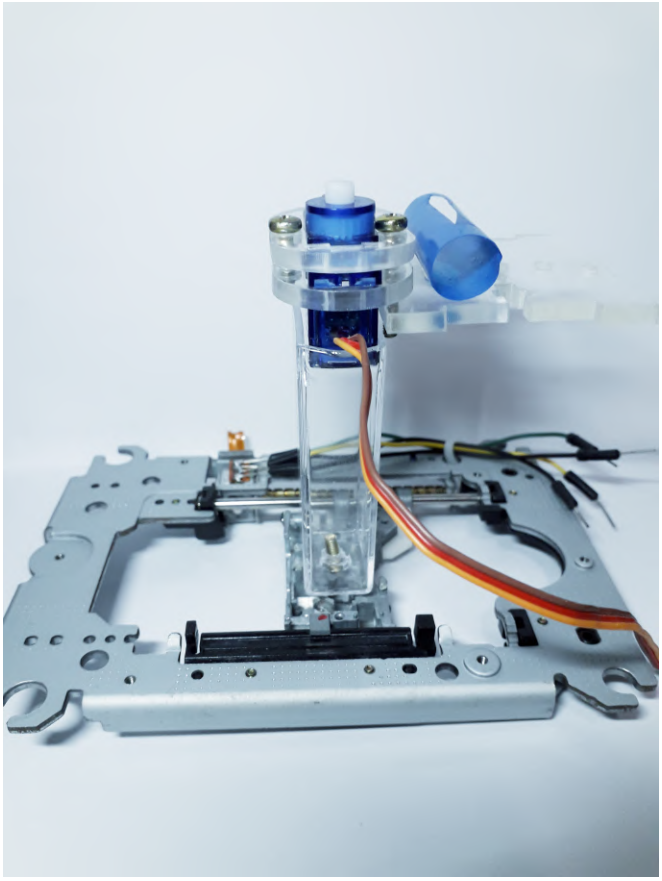
Fonte: Do Autor (2019).

Figura 4.11 – Caixinha de apontador fixado ao motor



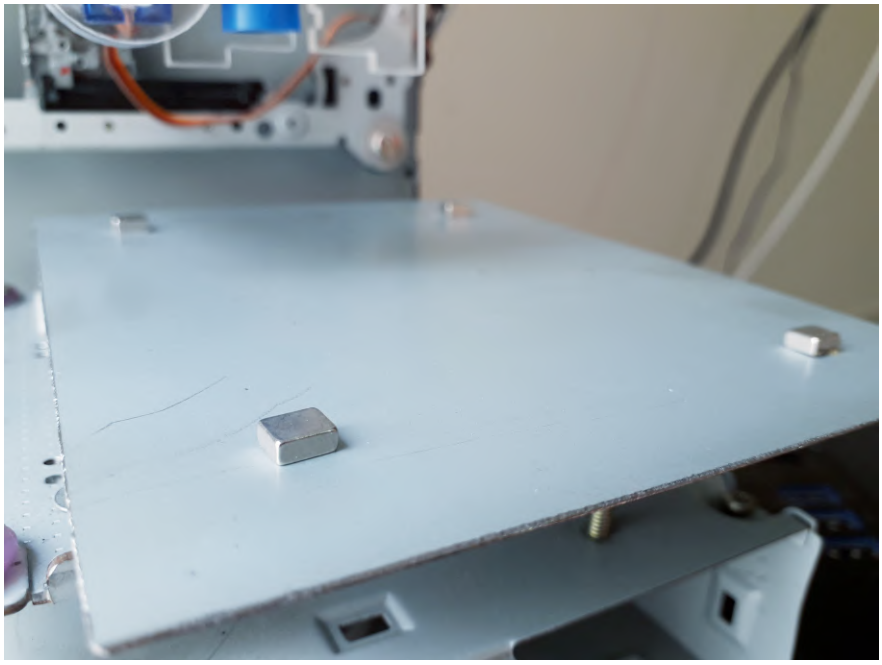
Fonte: Do Autor (2019).

Figura 4.12 – Servomotor fixado à caixinha de apontador



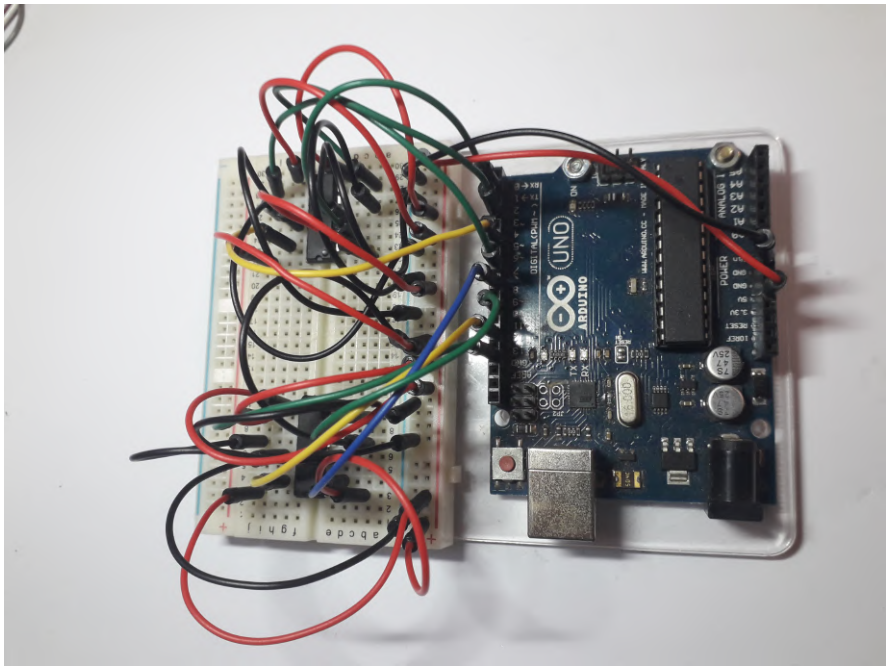
Fonte: Do Autor (2019).

Figura 4.13 – Base metálica para a impressão



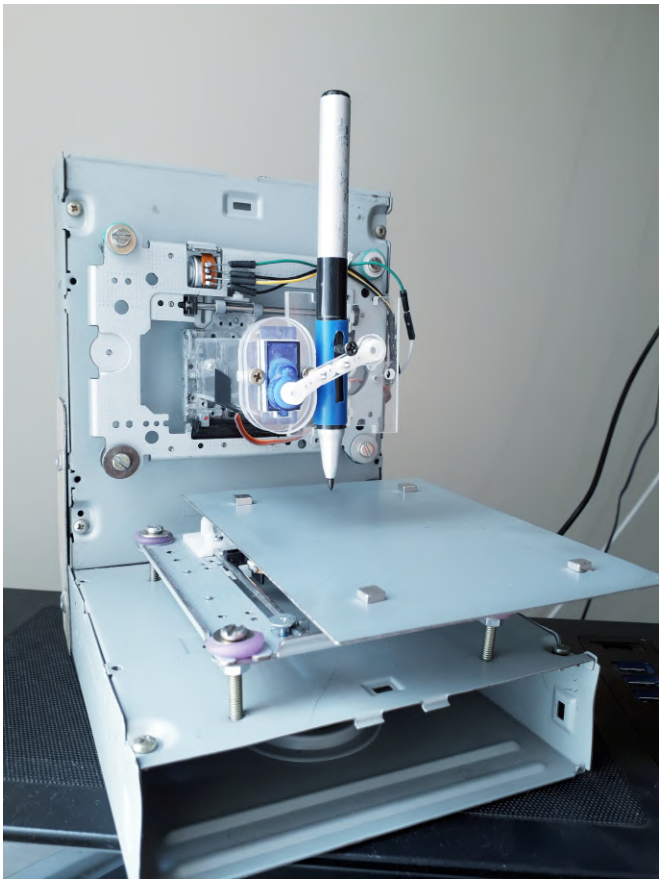
Fonte: Do Autor (2019).

Figura 4.14 – Conexões elétricas do projeto




Fonte: Do autor (2019).

Figura 4.15 – Montagem final



Fonte: Do Autor (2019).

Durante a implementação do *software* no Arduino, foram submetidos como teste códigos que continham apenas valores positivos de X e Y. Porém, realizando impressões de teste, foi observado que apenas a área de desenho que estivesse no primeiro quadrante da página do *software Inkscape* era realmente impresso. Nos demais casos, no mínimo um dos motores deixava de responder quando a impressão seguia para pontos negativos do eixo. Isso de deve ao fato de que a lógica implementada no controlador não estava preparada para receber valores negativos.

Uma forma simples que resolver a questão foi garantir que todos os pontos se tornassem positivos. Como a área de impressão da *Plotter* é de 40x40mm, e seus eixos vão de -20 à +20, bastou realizar uma soma de 20 unidades aos valores dos pontos recebidos. Assim, havia a garantia de que todos os pontos possuíssem valores de 0 à 40, tornando, assim, todos eles positivos e garantindo a impressão dos trilhos em todos os quadrantes.

Uma outra observação feita foi que, muitas vezes, para realizar a impressão de um trilho reto, a *Plotter* realizava diversas operações, imprimindo a linha reta em partes, segmentando um trilho que poderia ser feito em apenas uma operação, tornando a impressão mais lenta e causando um maior processamento.

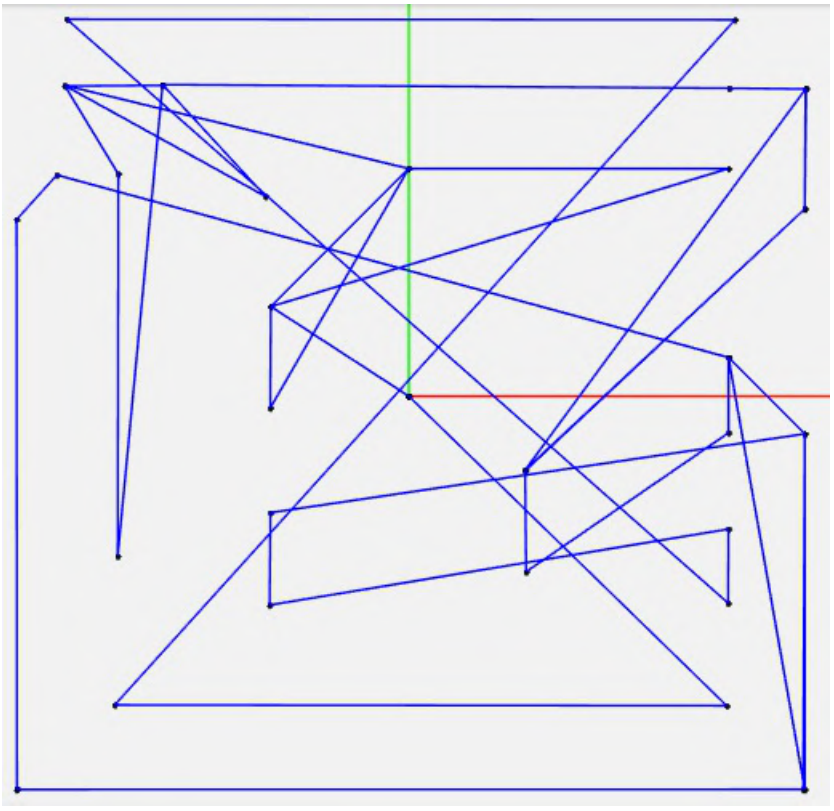
Foi observado, então, que o *software KiCad*, ao exportar o arquivo no formato SVG, nem sempre conseguia definir a melhor forma de traçar os vetores. Por isso, muitas vezes, um trilho longo e reto pode ser convertido em diversos vetores, todos eles possuindo a mesma direção e sentido. Ou seja, como se um único vetor estivesse dividido em vários. Como a impressora faz a leitura de ponto por ponto, se torna mais interessante que ao invés de possuir diversos vetores alinhados com a mesma direção e sentido, exista apenas um, com um comprimento referente à soma dos anteriores. Desta forma, a impressora fará a impressão do trilho por completo lendo apenas uma linha de código.

Outro fator observado é que, pelo fato do *software KiCad* gerar vetores independentes, sem conexão entre si, ao gerar o arquivo *G Code* referente ao projeto, cada segmento será considerado um grupo diferente. A impressora, ao descer a caneta, realiza a impressão passando por um grupo de pontos. Ao finalizar o grupo, a caneta é levantada, levada a outro ponto, e então inicia a impressão de outro grupo. Desta forma, ao finalizar a impressão de um segmento e for passar para outro, ela sempre irá levantar e abaixar a caneta, mesmo que os segmentos estejam conectados fisicamente entre si, pois pertencem a grupos diferentes. Uma alteração interessante a ser realizada no *software Inkscape* é conectar via *software* todos os segmentos que possuem

contato, para que assim, ao gerar o arquivo *G Code*, todos eles pertençam a um mesmo grupo, e assim a *Plotter* realize a impressão dos segmentos de forma direta, sem retirar o contato da caneta à superfície, tornando o processo mais rápido, eficiente e com melhor qualidade.

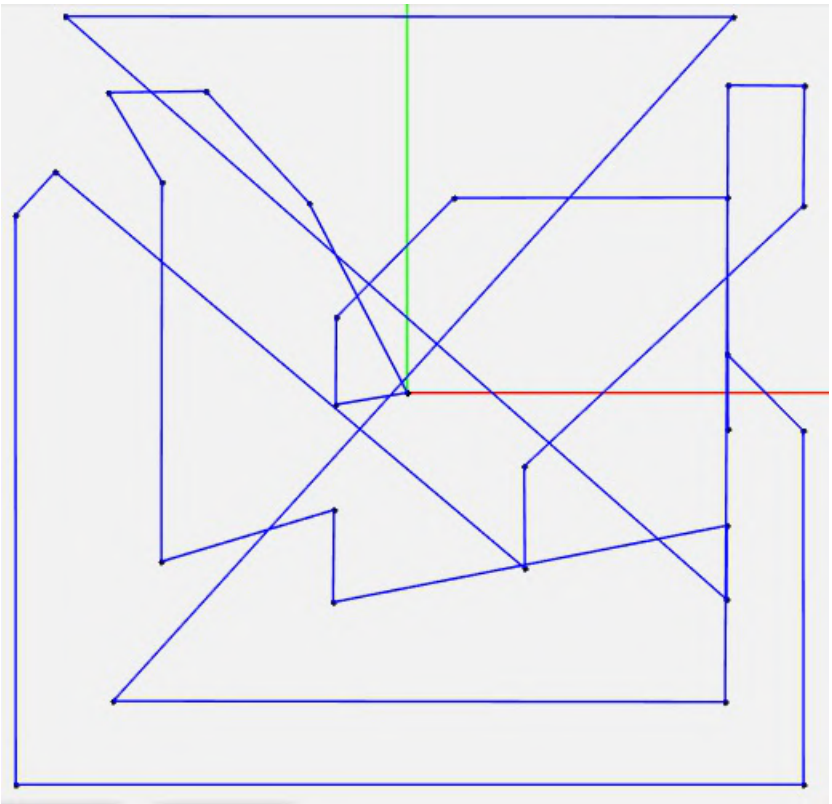
Com base nisso, no *software Inkscape* foram realizadas as devidas modificações, excluindo vetores fragmentados, conectando-os e mantendo a quantidade mínima possível para realizar a impressão. Desta forma, a impressão se mostrou mais rápida, direta e menos suscetível a erros. Esta modificação pôde ser feita facilmente com as ferramentas que o programa oferece. Com o auxílio do site www.ncviewer.com, foi possível verificar visualmente os caminhos que a ferramenta da impressora percorrerá com base no arquivo *G Code*. A Figura 4.16 ilustra tais movimentos no projeto da placa em questão antes da modificação dos vetores. Já a Figura 4.17 ilustra os movimentos após os vetores serem alterados. É possível observar uma grande queda nos caminhos percorridos pela ferramenta da *Plotter*. O código *G Code* passou de 181 linhas para 92, ou seja, uma diminuição de cerca de 50% no tamanho, e consequentemente no processamento, demonstrando que é uma modificação interessante a ser realizada, principalmente em projetos maiores.

Figura 4.16 – Caminhos percorridos pela *Plotter* antes da modificação



Fonte: Do autor (2019).

Figura 4.17 – Caminhos percorridos pela *Plotter* depois da modificação



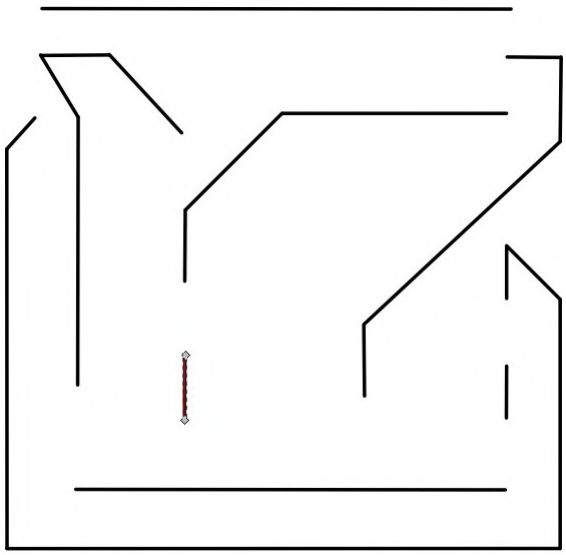
Fonte: Do autor (2019).

Outra modificação que pode ser feita é em relação aos pequenos círculos onde serão feitos os pontos de solda nos componentes. Por possuírem formatos circulares, eles serão formados por dezenas de pequenos vetores, muitas vezes de tamanho irrelevante para a impressão. Por isso, sua impressão se mostrou lenta e pode descalibrar os parâmetros de posição da ferramenta, tornando a impressão final desfigurada. Por isso, é interessante que tais pontos de solda circulares sejam apagados, e caso seja necessário para o projeto da placa, pode ser desenhado à mão posteriormente. Para a impressão, os trilhos foram distribuídos de forma a utilizar melhor o espaço disponível. A Figura 4.18 mostra o projeto aberto no *software Inkscape* já com as modificações feitas.

Como a velocidade setada dos motores foi de 250 rotações por minuto, e para cada rotação o motor deve gerar 20 passos, temos que, por minuto, o motor gera 5000 passos, ou aproximadamente 83 passos por segundo. Como a ferramenta se movimenta 1mm a cada 6 passos gerados, temos que para 83 passos ela se deslocará aproximadamente 14mm. Ou seja, a velocidade de deslocamento de ferramenta é de cerca de 14mm/s.

A impressão da placa em questão, após as modificações citadas, se mostrou precisa, apresentou boa qualidade e se mostrou relativamente rápida, levando cerca de 53 segundos para

Figura 4.18 – Projeto da placa modificado no *Inkscape*



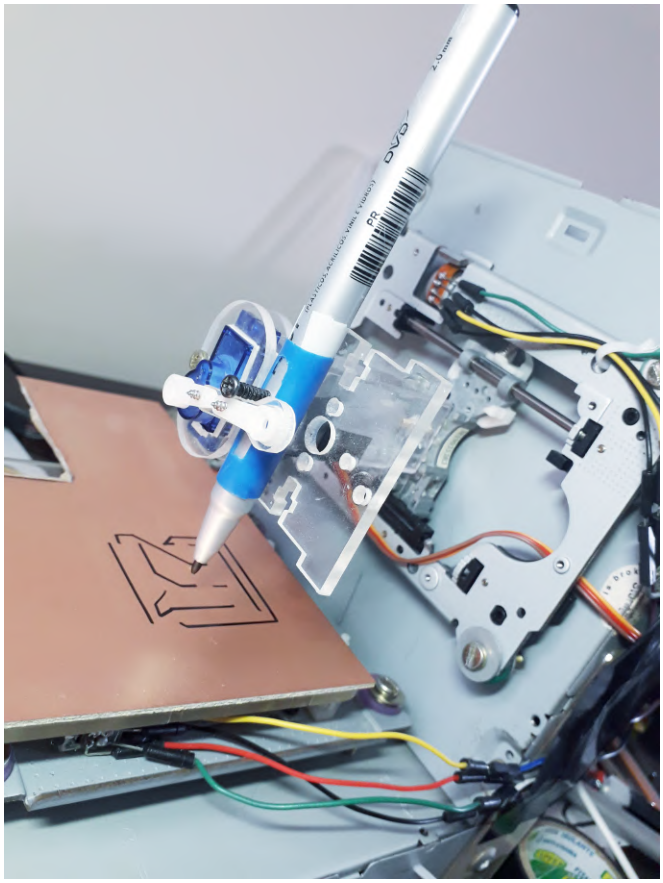
Fonte: Do autor (2019).

a impressão completa, adotando uma velocidade de 250 rotações por minuto para ambos os motores. A Figura 4.19 mostra a *Plotter* em seu processo de impressão em uma placa de circuito eletrônico. A Figura 4.20 mostra a placa já finalizada e já mergulhada em percloreto de ferro. Ela apresentou bom acabamento e linhas coerentes ao projeto criado. Foram observadas algumas pequenas deformações nos trilhos, causadas principalmente pelo movimento horizontal da caneta, que por mais que tenha sido ajustada, continua havendo uma pequena folga por onde ela se desloca.

Por conta da geração de muitos vetores para projetos com formatos circulares, ela não se mostrou eficiente para projetos com linhas curvas. Em testes realizados, ao tentar imprimir um círculo, ela se mostrou extremamente lenta, e o resultado não foi satisfatório, não gerando um circulo perfeito e não conectando o desenho circular ao final do processo. Porém, baseado no fato de que o objetivo principal é a impressão de trilhos de placas de circuito eletrônico, onde seus movimentos serão basicamente linhas retas, o projeto se mostrou eficiente para placas menos complexas e gerou um resultado com aspecto profissional.

Pelo fato do *KiCad* exportar o projeto como um grupo de vetores, a *Plotter* não terá a informação de espessura do trilho. Aliado a isso, a caneta utilizada não é capaz de alterar a espessura de sua ponta. Portanto, a grossura do trilho será comparável à da ponta da caneta, não sendo uma variável controlada. Uma forma se tornar os trilhos menos finos é fazer um pequeno corte na ponta da caneta, retirando sua ponta fina e assim fazendo com que os trilhos sejam impressos com maior espessura.

Figura 4.19 – *Plotter* em seu processo de impressão



Fonte: Do autor (2019).

Figura 4.20 – Placa impressa finalizada

Fonte: Do autor (2019).

Uma forma de alterar a espessura do trilho via *software* é adicionar vetores paralelos no projeto no *Inkscape*. Desta forma, adicionando vetores quase sobrepostos entre si, fará com que a *Plotter* imprima os trilhos uns sobre os outros e assim gere como resultado um trilho de maior espessura.

No Apêndice C é possível verificar o resultado de mais outras quatro impressões de teste. Estas impressões não possuem relação com qualquer projeto elétrico, sendo apenas contornos aleatórios para verificar a qualidade de impressão e fazer a comparação com o projeto no computador. A Figura 1 mostra o resultado de uma impressão sobre uma placa de circuito. As Figuras 2, 3 e 4 apresentam impressões realizadas sobre papel branco. Por conta da dissipação da tinta da caneta pelo papel, o resultado final apresentou maior distorção do que a impressão sobre a placa. Porém, é possível observar que os trilhos gerados são totalmente condizentes com o projeto realizado via *software*.

# 5 CONCLUSÃO

Desde a década de 50, máquinas CNC têm se tornado cada vez mais populares e essenciais nos mais diversos setores da indústria. Apesar de sua popularidade, os preços praticados pelo mercado se tornam muitas vezes enviável sua utilização, principalmente em setores com baixo investimento.

Considerando a dificuldade de encontrar equipamentos disponíveis e considerando sua grande utilidade dentro da Universidade Federal de Lavras, principalmente nos cursos de Engenharia, foi realizado o projeto de construção da uma *Plotter* de baixo custo reutilizando equipamentos da própria Universidade para realizar impressão de trilhos de circuito eletrônico, desde a sua construção física até a implementação de sua lógica de programação.

A estrutura da *Plotter* se mostrou bastante compacta, resistente e fácil de ser transportada, possuindo tamanho total de cerca de 15cm de largura, 20cm de altura e 20cm de comprimento, desconsiderando o espaço necessário para o controlador e os fios, que foi fixado na parte traseira da impressora.

O processo de impressão apresentou qualidade aceitável e se mostrou eficiente para a impressão de trilhos de circuitos pouco complexos e que contenha basicamente linhas retas. O motor de passo presente nos *drivers* são simples e possuem pouca precisão em relação aos demais encontrados no mercado, porém se mostraram suficientes para realização das tarefas propostas.

O tempo de impressão pode variar de acordo com o projeto feito, com a quantidade de blocos existentes e com determinadas configurações da *Plotter*, como velocidade dos motores e *delays* entre funções. No geral, o tempo de impressão foi curto, se mantendo próximo à 1 minuto para os exemplos realizados.

Considerando os objetivos propostos neste projeto, pode-se concluir que o mesmo apresentou resultados satisfatórios e foi capaz de realizar impressões de trilhos de circuito impresso reutilizando equipamentos já em desuso e garantindo um custo baixo de construção.

Houve grande dificuldade em implementar o código no microcontrolador Arduino pelas restrições de comunicação dos *softwares* com a placa controladora.

O ambiente de desenvolvimento Arduino permite manter uma comunicação constante com o controlador via porta USB, e assim é possível receber as informações da execução do projeto em tempo real na tela do computador, auxiliando na resolução de problemas. Porém, durante a impressão, o *software Processing* deve manter uma comunicação constante com o

controlador, visto que é ele quem envia cada linha de comando à impressora. Desta forma, quando o ambiente de desenvolvimento do Arduino tenta se comunicar com o controlador, gera uma concorrência que não permite que ambos os programas se comuniquem ao mesmo tempo via porta USB. Logo, não é possível obter retorno do controlador durante a impressão, apenas do que é enviado pelo *Processing*.

Isso tornou o processo de implementação bastante complicado. Uma forma de simular uma impressão e conseguir receber seus dados em tempo real foi não utilizar o *Processing* para enviar os códigos *G Code*. Algumas linhas de teste foram implementadas dentro do próprio controlador, de forma que em vez de receber os códigos via porta serial, passasse a utilizar as linhas inseridas dentro do código. Assim, ele não dependeria da comunicação com o *Processing* e tornou possível obter os resultados da execução do código em tempo real, e assim realizar as correções necessárias. Porém, tal processo fez com que algumas situações específicas não fossem tratadas, como quando o valor de X e Y fosse negativo.

Uma outra etapa do projeto que demonstrou maior dificuldade do que o esperado foi o de solda dos fios aos terminais dos motores. Pelo fato de os motores serem extremamente pequenos, com pouco mais de 1cm de diâmetro, seus terminais ficavam muito próximos uns aos outros. Isso fez com que a soldagem se tornasse bastante delicada, tornando difícil para quem não possui o costume de tal atividade. Além disto, mesmo após serem soldadas, as conexões se mostraram bastante frágeis, soltando a todo o momento. Um dos motores foi perdido durante o projeto pela quantidade de vezes que a solda foi refeita.

Uma possível melhoria a ser feita é estender a capacidade da *Plotter* para imprimir figuras curvas de forma direta e sem descalibragem do equipamento. Para isso, os *softwares* utilizados provavelmente deverão ser outros, e o código do controlador deve ser alterado para ter a capacidade de tratar esses novos comandos.

Uma segunda possível melhoria seria tornar a *Plotter* capaz de fazer a impressão de trilhos de diferentes espessuras. Uma solução seria utilizar uma ferramenta que liberasse uma quantidade constante de tinta quando em contato com a placa. Desta forma, para alterar a espessura, bastava alterar a velocidade de impressão do segmento pela impressora. Quanto mais lenta ela fosse, maior a quantidade de tinta liberada, e maior a espessura do trilho gerado. Tais melhorias podem ser realizadas em projetos futuros utilizando o equipamento.

# REFERÊNCIAS


ALIEXPRESS. **1 PC Neue Mini Micro DC 5-6 V 15mm 2-phase 4-draht Stepper Motor mit 55mm Lange Achse**. 2019. Disponível em: <https://de.aliexpress.com/item/32976002061.html?spm=a2g0o.detail.1000014.17.3b6874a7CJC7K2&gps-id=pcDetailBottomMoreOtherSeller&scm=1007.13338.141932.0&scm_id=1007.13338.141932.0&scm-url=1007.13338.141932.0&pvid=efbe4b4c-ec46-4074-b7fb-70d2dac974cd>. Acesso em: 09 ago. 2019.

ARDUINO. **Documentação de Referência da Linguagem Arduino**. 2019. Disponível em: <https://www.arduino.cc/reference/pt>. Acesso em: 05 ago. 2019.

ARDUINO. **What is Arduino?** 2019. Disponível em: <https://www.arduino.cc/en/Guide/Introduction>. Acesso em: 8 set. 2019.

ATHOSELECTRONICS. **Ponte H: Tutorial com Transistores TIP120**. 2019. Disponível em: <https://athoselectronics.com/ponte-h>. Acesso em: 02 set. 2019.

AUTOCOREROBOTICA. **Como utilizar o Processing com Arduino**. 2018. Disponível em: <http://autocorerobotica.blog.br/utilizando-processing-com-arduino-parte-1>. Acesso em: 18 ago. 2019.

BRITES, F. G.; SANTOS, V. P. d. A. **Motor de Passo**. 1. ed. Niterói, Rio de Janeiro: Universidade Federal Fluminense, 2008.

CIRCUITAR. **Programação para Arduino**. 2018. Disponível em: <https://www.circuitar.com.br/tutoriais/programacao-para-arduino-primeiros-passos/#linguagem-de-programao>. Acesso em: 28 ago. 2019.

DENFORD. **G and M Programming for CNC Milling Machines**. 1. ed. West Yorkshire, Inglaterra: Denford Computerised Machines and Systems, 2000.

DIGITALTRENDS. **The brief but building history of 3D printing**. 2019. Disponível em: <https://www.digitaltrends.com/cool-tech/history-of-3d-printing-milestones>. Acesso em: 02 set. 2019.

DOBITAOBYTE. **Como fazer uma CNC com drive de DVD**. 2018. Disponível em: <https://www.dobitaobyte.com.br/como-fazer-uma-cnc-com-drive-de-dvd-velho>. Acesso em: 25 ago. 2019.

ENGLEANDROALVES. **Controlando Servo Motores com PIC16F877A**. 2012. Disponível em: <https://engleandroalves.wordpress.com/2012/07/15/controlando-servo-motores-com-pic16f877a>. Acesso em: 13 set. 2019.

FILIPEFLOP. **O que é Arduino?** 2019. Disponível em: <https://www.filipeflop.com/blog/o-que-e-arduino/>. Acesso em: 20 ago. 2019.

INKSCAPE. **Inkscape - Visão geral**. 2019. Disponível em: <https://inkscape.org/pt-br/sobre/visao-geral>. Acesso em: 5 set. 2019.

KICAD. **KiCad - Reference manual**. 2019. Disponível em: <http://docs.kicad-pcb.org/5.1.4/en/kicad/kicad.html>. Acesso em: 5 set. 2019.

MACHMOTION. **G  M Code Reference Manual**. 1. ed. Newburgh, USA: MachMotion, 2016.

OPENSOURCE. **What is an Arduino?** 2019. Disponível em: <https://opensource.com/resources/what-arduino>. Acesso em: 17 ago. 2019.

PEREIRA, A. G. **Desenvolvimento e avaliação de um editor para programação CN em centros de usinagem**. 1. ed. Curitiba: Universidade Federal do Paraná, 2003.

PROTOPTIMUS. **Máquinas CNC: A história do Comando Numérico Computadorizado**. 2017. Disponível em: <http://www.protoptimus.com.br/maquinas-cnc-historia-comando-numerico-computadorizado>. Acesso em: 18 jul. 2019.

TEXAS INSTRUMENTS. **L293x Quadruple Half-H Drivers**. Dallas, Texas, 2016. Disponível em: <http://www.ti.com/product/L293>.

THE MIT PRESS. **Processing: a programming handbook for visual designers and artists**. 1. ed. Cambridge, Massachusetts: Massachusetts Institute of Technology, 2007.

VERTEXUSINAGEM. **Torno CNC e a sua função na usinagem de precisão!** 2017. Disponível em: <http://vertexusinagem.ind.br/noticia/torno-cnc-e-sua-funcao-na-usinagem>. Acesso em: 25 jul. 2019.

ZMORPH3D. **GUIA: Entendendo a fresadora CNC**. 2017. Disponível em: <https://medium.com/bsbfablab/guia-entendendo-a-fresadora-cnc-583fa153ef98>. Acesso em: 18 jul. 2019.

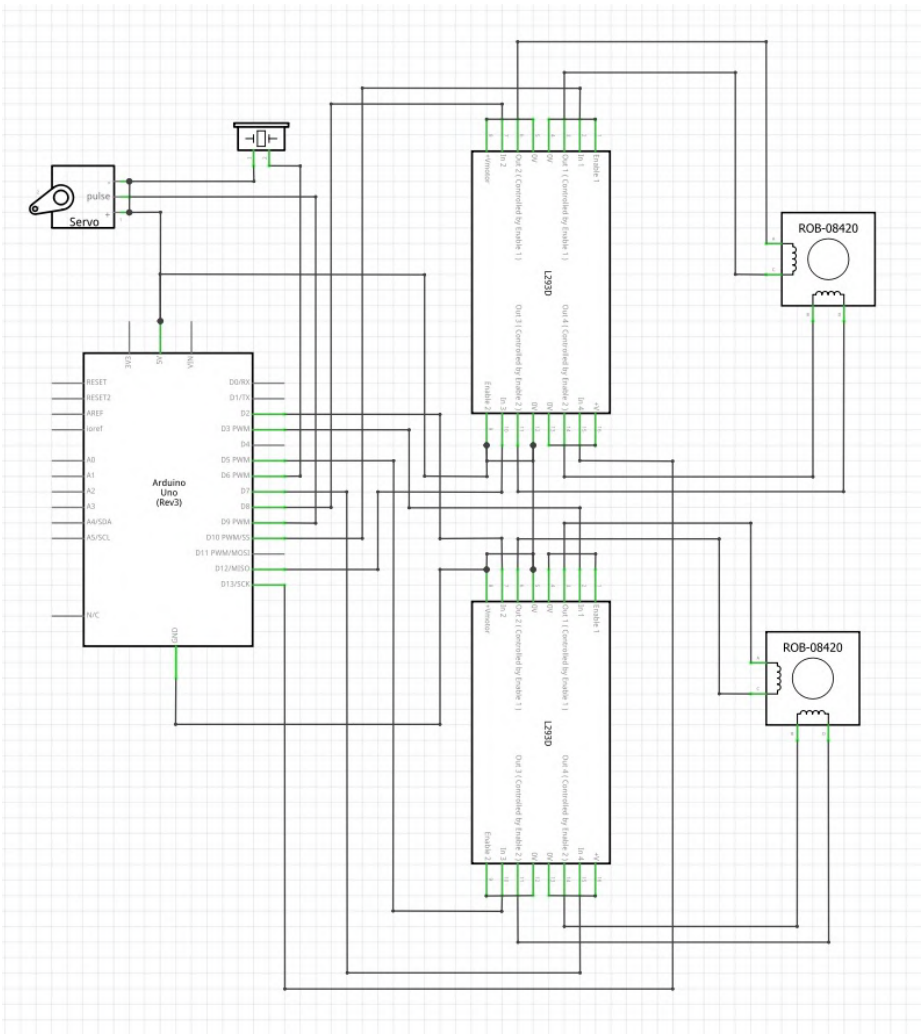
ZMORPH3D. **TORNO CNC**. 2019. Disponível em: <http://mesindustrial.com.br/torno-cnc>. Acesso em: 25 jul. 2019.

## APÊNDICE A – Lista de alguns dos códigos encontrados nos arquivos *G Code*

| Code | Group | Description | Modal | Page |
|---|---|---|---|---|
| G00 | 1 | Rapid Move | Y | 10 |
| G01 | 1 | Linear Feed Move | Y | 10 |
| G02 | 1 | Clockwise Arc Feed Move | Y | 11 |
| G03 | 1 | Counter Clockwise Arc Feed Move | Y | 11 |
| G04 | 0 | Dwell | N | 14 |
| G09 | 0 | Exact stop | N | 14 |
| G10 | 0 | Fixture and Tool Offset Setting | N | 15 |
| G12 | 1 | Clockwise Circle | Y | 18 |
| G13 | 1 | Counter Clockwise Circle | Y | 18 |
| G15 | 11 | Polar Coordinate Cancel | Y | 18 |
| G16 | 11 | Polar Coordinate | Y | 18 |
| G17 | 2 | XY Plane Select | Y | 20 |
| G18 | 2 | ZX Plane Select | Y | 20 |
| G19 | 2 | YZ Plane Select | Y | 20 |
| G20 | 6 | Inch | Y | 20 |
| G21 | 6 | Millimeter | Y | 20 |
| G28 | 0 | Zero Return | N | 21 |
| G30 | 0 | $2^{nd}$, $3^{rd}$, $4^{th}$Zero Return | N | 22 |
| G31 | 1 | Probe function | N | 22 |
| G32 | 1 | Threading* | N | 23 |
| G40 | 7 | Cutter Compensation Cancel | Y | 23 |
| G41 | 7 | Cutter Compensation Left | Y | 25 |
| G42 | 7 | Cutter Compensation Right | Y | 25 |
| G43 | 8 | Tool Length Offset + Enable | Y | 25 |
| G44 | 8 | Tool Length Offset - Enable | Y | 25 |
| G49 | 8 | Tool Length Offset Cancel | Y | 25 |
| G50 | 9 | Cancel Scaling | Y | 25 |

Fonte: (MACHMOTION, 2016)

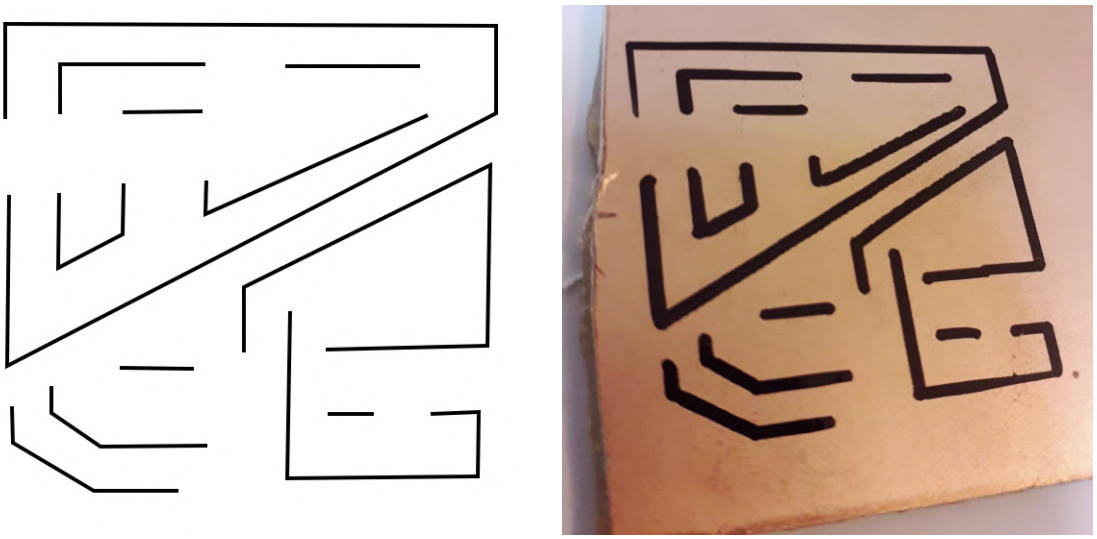
# APÊNDICE B – Esquemático do projeto




Fonte: Do autor (2019).

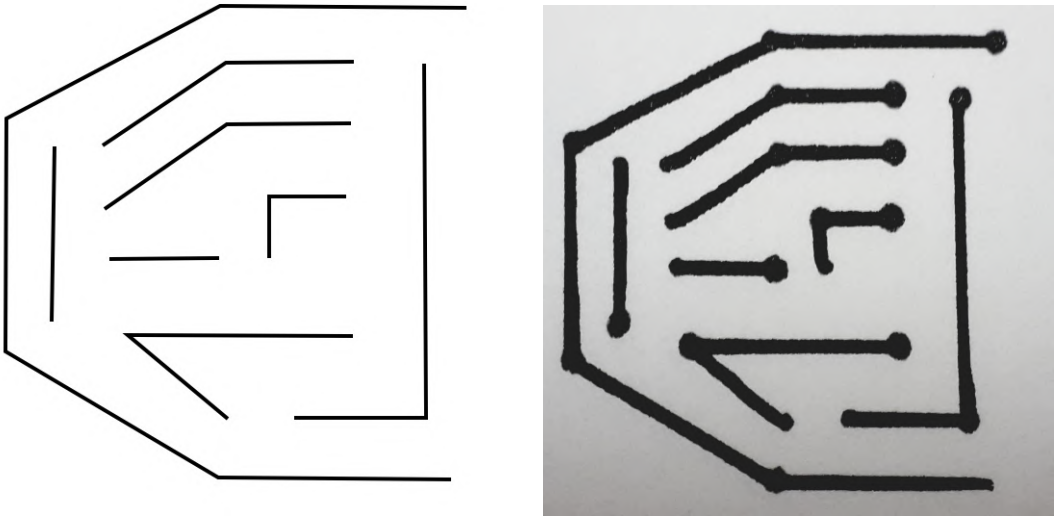
# APÊNDICE C – Impressões

Figura 1 – Impressão de teste 1



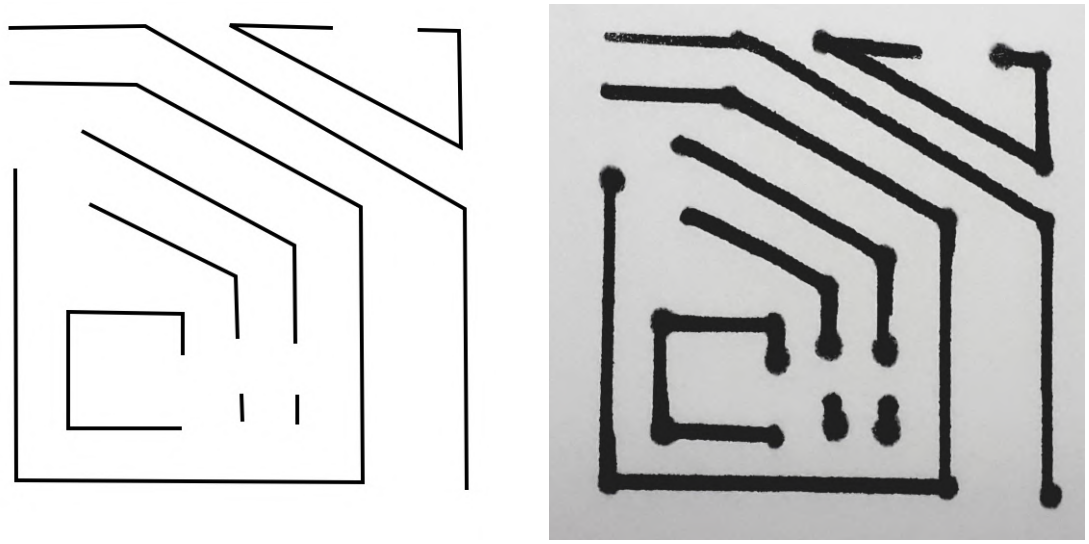
Fonte: Do Autor (2019).

Figura 2 – Impressão de teste 2



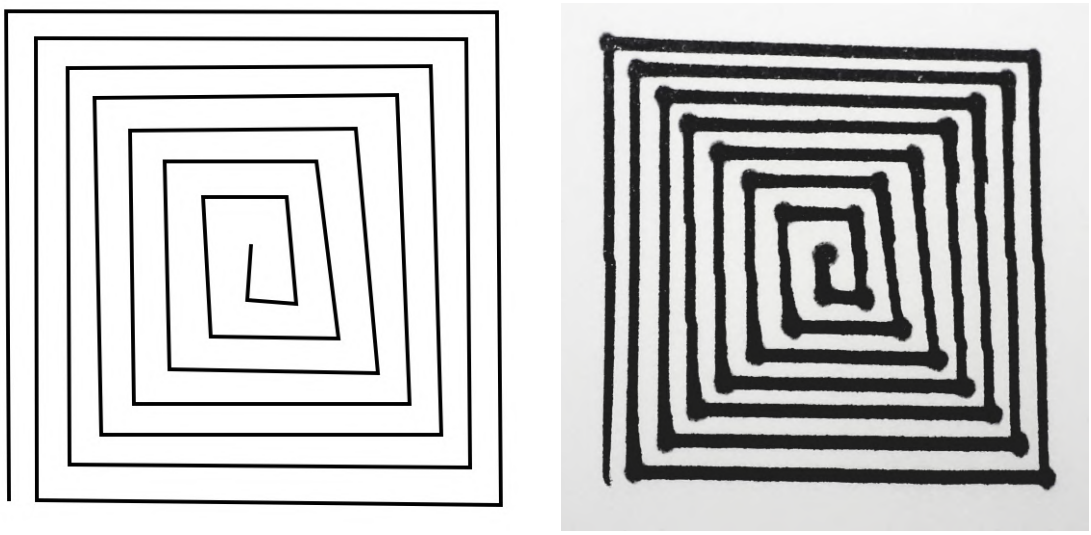
Fonte: Do Autor (2019).

Figura 3 – Impressão de teste 3



Fonte: Do Autor (2019).

Figura 4 – Impressão de teste 4



Fonte: Do Autor (2019).