PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS
ESCOLA POLITÉCNICA
GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO

CONSTRUÇÃO DE UMA MINI *PLOTTER* ARTESANAL PARA DESENHO

GIOVANNA DE SOUSA SAMPAIO

GOIÂNIA
2021

GIOVANNA DE SOUSA SAMPAIO

CONSTRUÇÃO DE UMA MINI *PLOTTER* ARTESANAL PARA DESENHO

Trabalho de Conclusão de Curso apresentado à Escola Politécnica, da Pontifícia Universidade Católica de Goiás, como parte dos requisitos para a obtenção do título de Bacharel em Engenharia de Computação.

Orientador:
Prof. M.E.E. Marcelo Antonio Adad de Araújo

GOIÂNIA
2021

GIOVANNA DE SOUSA SAMPAIO

CONSTRUÇÃO DE UMA MINI *PLOTTER* ARTESANAL PARA DESENHO

Trabalho de Conclusão de Curso aprovado em sua forma final pela Escola Politécnica, da Pontifícia Universidade Católica de Goiás, para obtenção do título de Bacharel em Engenharia de Computação, em 08/12/2021.

_____

Prof(a). Ma. Ludmilla Reis Pinheiro dos Santos
Coordenadora de Trabalho de Conclusão de Curso

Banca examinadora:

_____

Orientador Prof. M.E.E. Marcelo Antonio Adad de Araújo

_____

Prof(a). MSc. Mirian Sandra Rosa Gusmão

_____

Prof. M.E.E. Carlos Alexandre Ferreira de Lima

GOIÂNIA
2021

A Deus pela saúde e vida, a minha família que sempre me apoiou em tudo e ao meu orientador que me ajudou consideravelmente.

AGRADECIMENTOS

Agradecer a Deus por me permitir viver todos os dias, por me dar saúde e por ter me concedido uma família linda que amo de coração.

Agradecer aos meus pais e minha irmã por me apoiarem sempre, no caminho tortuoso e complicado da vida, de todos os modos e momentos em que estiveram comigo. Minha irmã Andressa, especialmente que me ajudou na correção ortográfica e revisão do grau de entendimento dos assuntos relatados neste TCC; e aos cuidados e carinhos dos meus pais Grasilvan e Mônica.

Agradecer ao meu orientador, Marcelo Adad Antonio de Araújo, que me ajudou e orientou desde o começo desta jornada. Presente desde janeiro, todas as quintas-feiras, mas não somente nesses dias, também me ajudava até mesmo no domingo.

Agradecer aos meus amigos, principalmente ao Alexandre Rodrigues que me acompanhou em grande parte desta trajetória, agradecer a minha amiga Clara por sempre estar comigo e as minhas amigas Amanda Kelly Cândido e Giulianni Oliveira que são pessoas incríveis que eu tive a oportunidade e o prazer de conhecer desde o primeiro período.

RESUMO

O presente trabalho de conclusão de curso apresenta o uso de um CNC (*Computer Numerical Control*) e a construção do protótipo mini *plotter* artesanal para criação de desenhos computadorizados. Estes tipos de máquinas ferramentas se destinam à utilização em trabalhos mais precisos de desenho, o que facilita em muito o trabalho que antes seria de forma artesanal e artística. A máquina ferramenta em questão será dividida quanto ao seu desenvolvimento em componentes de *hardware* e componentes de *software*. Nos componentes de *hardware* são descritos de forma a apresentar sua construção e o sistema elétrico utilizado. Para servir de carcaça a *plotter* CNC é construída de partes reutilizadas de um computador, usando dois *drivers* CD-ROM e um *driver* de disquete, assim como a utilização de um microcontrolador Arduino Uno, com a expansão CNC *Shield* V3, e os motores desses três *drivers*. Para os componentes de *software* optou-se por programas *open-source* para melhor manipulação de dados, sendo assim o *firmware* UGS (*Universal G-code Sender*) e o *Inkscape* são de suma importância para manipulação de desenhos gráficos para o envio na *shield*. O equipamento proposto possui um sistema mecânico de movimentação de três eixos (X, Y e Z). O movimento da máquina será executado através de comandos gerados a partir de arquivos no formado *G*-code, sendo este uma tradução para a máquina CNC. Neste propósito, o *firmware* GRBL, é alocado na memória do Arduino Uno realizando a conexão com o computador através da conexão USB, e este servirá como um interpretador e tradutor para *shield*, que após recebimento do arquivo *G-code* enviará à CNC *Shield* e realiza os movimentos nos eixos fazendo com que a máquina desenhe conforme especificado na criação do desenho. Os resultados obtidos são atendidos ao proposto, pois o protótipo funcionou de acordo com o especificado no trabalho.

Palavras-chave: Arduino Uno. CNC. CNC *Shield* V3. *G-code*. GRBL.

ABSTRACT

This course conclusion work presents the use of a CNC (Computer Numerical Control) and the construction of a handcrafted mini plotter prototype for creating computerized drawings. These types of machine tools are intended for use in more accurate drawing work, which greatly facilitates the work that would previously be done in a craft and artistic way. The machine tool in question will be divided as to its development in hardware components and software components. The hardware components are described in order to present their construction and the electrical system used. To serve as a frame, the CNC plotter is built from reused parts of a computer, using two CD-ROM drivers and a floppy disk driver, as well as the use of an Arduino Uno microcontroller, with CNC expansion Shield V3, and the motors of these three drivers. For the software components we opted for open-source programs for better data manipulation, so the firmware UGS (Universal G-code Sender) and Inkscape are of paramount importance for manipulating graphic designs for sending in the shield. The proposed equipment has a mechanical movement system with three axes (X, Y and Z). The machine movement will be executed through commands generated from files in the G-code format, which is a translation for the CNC machine. For this purpose, the GRBL firmware is allocated in the Arduino Uno's memory, making the connection to the computer through the USB connection, and this will serve as an interpreter and translator for shield, which after receiving the G-code file will send it to the CNC Shield and perform the movements on the axes causing the machine to draw as specified when creating the drawing. The results obtained are in line with what was proposed, as the prototype worked as specified in the work.

Keywords: Arduino Uno. CNC. CNC *Shield* V3. *G-code*. GRBL.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

M.E.E.	Mestre em Engenharia Elétrica
MSc	*Master of Science*
Prof.	Professor
µs	Microssegundos

LISTA DE SIGLAS

2D	2 Dimensões
3D	3 Dimensões
AGC	*Apollo Guidance Computer*
BIOS	*Basic Input/Output System*
C++	*C Plus Plus*
CA	Corrente Alternada
CAD	*Computer-Aided Design*
CAM	*Computer-Aided Manufacturing*
CC	Corrente Contínua
CD	*Compact Disc*
CI	Circuito Integrado
CNC	*Computer Numeric Control*
E/S	Entrada/Saída
EIA	*Eletronics Industry Association*
EUA	Estados Unidos da América
GRBL	*Garble*
I/O	*Input/Output*
IBM	*International Business Machines*
IDE	*Integrated Development Environment*
LED	*Light-Emitting Diode*
MCU	Microcontrolador
MIT	Instituto de Tecnologia de Massachusetts
MPU	Microprocessador
NC	*Numerical Control*
ROM	*Read Only Memory*
SPI	*Serial Peripheral Interface*
SVG	*Scalable Vetor Graphics*
TCC	Trabalho de Conclusão de Curso
UGS	*Universal G-code Sender*
USB	*Universal Serial Bus*
V	*Volts*

SUMÁRIO

1 INTRODUÇÃO

Recentemente, a tecnologia cresceu e encontrou seu caminho em quase todos os aspectos das vidas humanas. Empresas de tecnologia vêm apresentando novas ideias revolucionárias todos os dias e têm confiado mais nas inovações, a cada dia mudando a trajetória da humanidade. Muitas das atividades têm se transformado por causa da tecnologia. Acessos e trocas de informações ocorrem com maior facilidade e rapidez, possibilitando assim a construção de pequenas máquinas em casa, até mesmo dispensando a necessidade de grandes equipamentos (GOBI, 2018).

Por volta da segunda metade do século XVIII, a Inglaterra deu início à Revolução Industrial, acontecimento que transformou a economia mundial e a forma de vida das pessoas. Foi o avanço de grandes marcos tecnológicos, como por exemplo a invenção da máquina a vapor, que foi relevante para o mundo da época que não sabia ainda a significância da tecnologia ou de seu emprego. As indústrias transformaram parte da mão-de-obra humana em automatizada ou automática, desenvolvendo máquinas que tornaram o trabalho repetitivo e insalubre ou mesmo pesado, em trabalho mais leve, trazendo precisão e agilidade no serviço (HOBSBAWM, 2014).

Devido ao emprego de maquinários em indústrias, viabilizou-se o aumento na produção fabril, ocorrendo então, inovações tecnológicas que não existiam implementadas na época. Sucedeu naquele período, a substituição da força muscular humana ou de animais em forças hidráulicas, a vapor e posteriormente com o emprego de atuadores elétricos. Com o passar do tempo, a tecnologia se tornou mais necessária e utilizada em todos os lugares. Mais tarde, em 1948, as máquinas industriais passaram a ser controladas numericamente e posteriormente computadorizadas, usando a tecnologia CNC (HOBSBAWM, 2014).

Uma definição de controle numérico computadorizado, refere-se a máquinas que são controladas por um computador através de um código específico, Código G. Nas indústrias, é comum encontrar equipamentos que se utilizam desta tecnologia. Para máquinas CNC, pode-se destacar: *plotter*, fresadoras mecânicas, tornos, impressoras 3D, máquinas de cortes e uma infinidade de dispositivos que necessitem de um controle numérico adaptado. Essas máquinas controladas, possuem uma quantidade específica de graus de liberdade, tanto para movimentos de translação quanto de rotação, que define o alcance de aplicação do dispositivo de forma a gerar geometrias adaptáveis da melhor forma possível ao trabalho a ser realizado pela máquina ferramenta (FORD, 2016).

O Arduino é uma placa de *hardware* cuja função pode ser programada por possuir a capacidade de *software* livre. Nele é possível ligar diferentes tipos de entradas e saídas, como

por exemplo projetar um sistema com o uso de um leitor digital, um *buzzer*, controladores de temperatura, controladores de pressão, entre outros. Só é preciso configurar a placa para realizar as necessidades requeridas no projeto, enviando um conjunto de instruções em um código para que o microcontrolador leia (MOLLE; ADAMS; WARREN, 2011). Normalmente o *software* usado para tal finalidade é o Arduino IDE.

O CNC *Shield* V3 é uma placa de código programado usado juntamente com o Arduino Uno para transformar o microcontrolador em um controlador CNC, onde representa instruções para movimentos precisos realizados na máquina usando até 4 motores de passo A4988 (METZ, 2020).

No primeiro capítulo é apresentado os principais objetivos e métodos para realização deste trabalho, mostrando também os resultados esperados e lista de atividades.

No capítulo dois e em suas seções, são apresentados o estado da arte e assuntos importantes para maior entendimento do trabalho.

O capítulo três apresenta a proposta de solução, mostrando os componentes de *hardware* e *software* é utilizado na criação de uma *plotter* artesanal integrado a um CNC.

O capítulo quatro apresenta a organização do trabalho, mostrando como a máquina ferramenta em questão é construída.

O capítulo cinco informa as considerações finais do trabalho, detalhando os testes realizados.

Os capítulos seis e sete apresentam respectivamente a conclusão e trabalhos futuros deste.

1.2 Objetivos

Esta seção descreve o objetivo geral e objetivos específicos do trabalho.

1.2.1 Objetivo Geral

O presente trabalho tem por objetivo criar uma mini *plotter* artesanal que se utiliza de um Arduino Uno e um CNC *Shield* V3, sistemas semelhantes a um controle numérico computadorizado.

1.2.2 Objetivos Específicos

- Relacionar e utilizar os conhecimentos técnicos adquiridos no curso de Engenharia de Computação e desenvolver a máquina ferramenta em questão;
- Desenvolver uma ferramenta para realizar desenhos de forma automática;
- Experimentar ferramentas de *softwares* livre e *hardwares* a serem utilizadas no presente trabalho.

1.3 Justificativa

O trabalho apresentado tem como justificativa primordial a criação de desenhos gráficos, computacionais, pelo fato deles serem mais precisos e exatos, tornando o trabalho mais bem elaborado. Ou seja, facilita em muito o trabalho de quem estiver manufaturando ou desenhando em um objeto é o que se espera na finalização.

1.4 Método

Para elaboração do trabalho, são realizadas várias pesquisas, em vários meios, a fim de se obter conhecimentos suficientes e necessários para a realização do projeto, da elaboração da composição e finalmente da montagem e construção de uma *plotter* com CNC, assim como o desenvolvimento e elaboração da parte escrita, de forma a registrar o mais fiel possível todo esse processo. São usados diferentes textos nos formatos dissertativos ou argumentativos, monográficos, trabalhos de conclusão de curso, dissertações de mestrados, teses de doutorados, livros, artigos de revistas, publicações online, entre demais outros meios de pesquisa virtual ou presencial.

O presente trabalho de conclusão de curso é realizado através de uma montagem de elementos mecânicos, encontrados em sucatas de *hardwares* computacionais. É apresentado uma mini *plotter* artesanal integrado a uma CNC, confeccionada com as referidas partes recuperadas de *hardwares* de um *floppy disk* e *driver*s de CD-ROM.

Para o desenvolvimento, é utilizado uma tecnologia semelhante a um Controlador Numérico Computadorizado, sendo um sistema de controle automatizado de uma máquina por meio de um computador programável com uma linguagem de programação específica para máquinas CNC, o código G. Nessa linguagem, é possível o controle simultâneo dos eixos X, Y e Z para movimentação da máquina (FORD, 2016).

1.5 Resultados esperados

Ao final do presente trabalho é possível o uso de um controlador numérico computadorizado artesanal construído *in home*, pelo desenvolvedor acadêmico do trabalho, usando objetos reutilizados de um computador mais antigo. Com esses componentes, é possível a criação da máquina ferramenta em questão.

Espera-se que os resultados deste trabalho possam auxiliar alunos de colégios, universidades ou afins, na construção de máquinas semelhantes, de mesmo porte ou maiores, que possam servir em trabalhos específicos de manufatura de desenhos em madeira, papel ou outros materiais.

1.6 Lista de atividades

Neste item são apresentados os materiais utilizados no projeto e suas finalidades.

1.6.1 Materiais utilizados

A tabela 1 apresenta a lista de materiais utilizados para a confecção da mini *plotter* artesanal para desenhos.

Tabela 1 – Lista de Materiais.

QUANT.	DESCRIÇÃO
2	*Driver* de CD-ROM
1	*Driver* de disquete
1	CNC *Shield* V3 + 4 A4988
1	Arduino Uno
1	Base da área (70mm x 70mm)
1	Papel
8	Parafusos
2	Arrebites
32	Arruelas
24	Porcas
1	Caneta
2	Grampos
*	Fios finos e maleáveis

Fonte: Elaborada pela autora.

É descrito um breve relato funcional das partes utilizadas:

- *Driver* de CD-ROM: um para ser o eixo X e outro para ser o eixo Y;
- *Driver* de disquete: funciona como o eixo Z;
- CNC *Shield* V3 + 4xA4988: primeiro principal componente da máquina. Usado para controlar as direções e limitações dos eixos X, Y e Z. Um escudo CNC e 4 *driver*s de motores;
- Arduino Uno: segundo principal componente da máquina. Usado para receber o desenho em forma de código G e transmitir ao CNC as instruções do desenho;
- Madeira Fundo: usado para formar a base do papel para o desenho;
- Papel: onde será realizado o desenho;
- Parafusos, arrebites, arruelas e porcas: usados para construção da carcaça da máquina;
- Caneta: objeto usado para desenhar;
- Grampos: usado para segurar a caneta no eixo Z;
- Fios finos e maleáveis: usados para fazer as ligações de entradas e saídas do CNC *shield* com o Arduino.

2 REFERENCIAL BIBLIOGRÁFICO

Este capítulo apresenta informações necessárias para compreensão do estado atual de conhecimento e desenvolvimento de conceitos, informando as referências necessárias para realização dos objetos de análise e estudo deste trabalho de conclusão de curso.

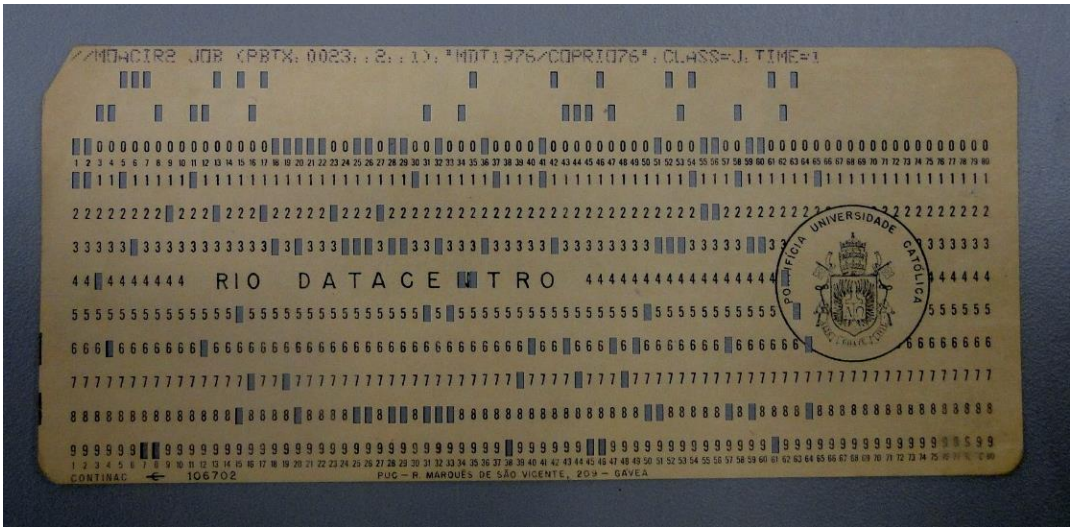
2.1 O que é CNC

CNC é a sigla para *Computer Numerical Control*, em português Controle Numérico Computadorizado, um dos objetos de estudo no qual está se direcionando o trabalho (SMID, 2003).

Antes das máquinas serem controladas com o auxílio de um computador, elas eram controladas por um conjunto de instruções na forma de números, letras ou símbolos como por exemplo um ponto, porcentagem ou outro e estas eram escritas em uma ordem lógica em fitas perfuradas, obtendo assim, o trabalho automatizado (SMID, 2003). Pelo fato dessas máquinas usarem cartões com furos para ler os comandos, elas eram conhecidas inicialmente como máquinas de Controle Numérico (NC, *Numerical Control*), posteriormente que se iniciou o uso de ferramentas computadorizadas e trocaram NC para CNC (MARCICANO, 2020).

Antes dos computadores possuírem grandes capacidades de armazenamento, memória, velocidade de processamento em pequeno porte, eles tinham um porte muito maior comparado com os atuais e obtinham dados e comandos das máquinas através de cartões com furos. Os cartões perfurados, como foram chamados, foram inicialmente projetados pela IBM e usados pela primeira vez em 1725 e aprimorados em 1801. As máquinas liam essas instruções pela presença ou ausência de furos nos cartões em posições predefinidas (TREJO, 2016). A figura 1 apresenta o modelo de um cartão perfurado.

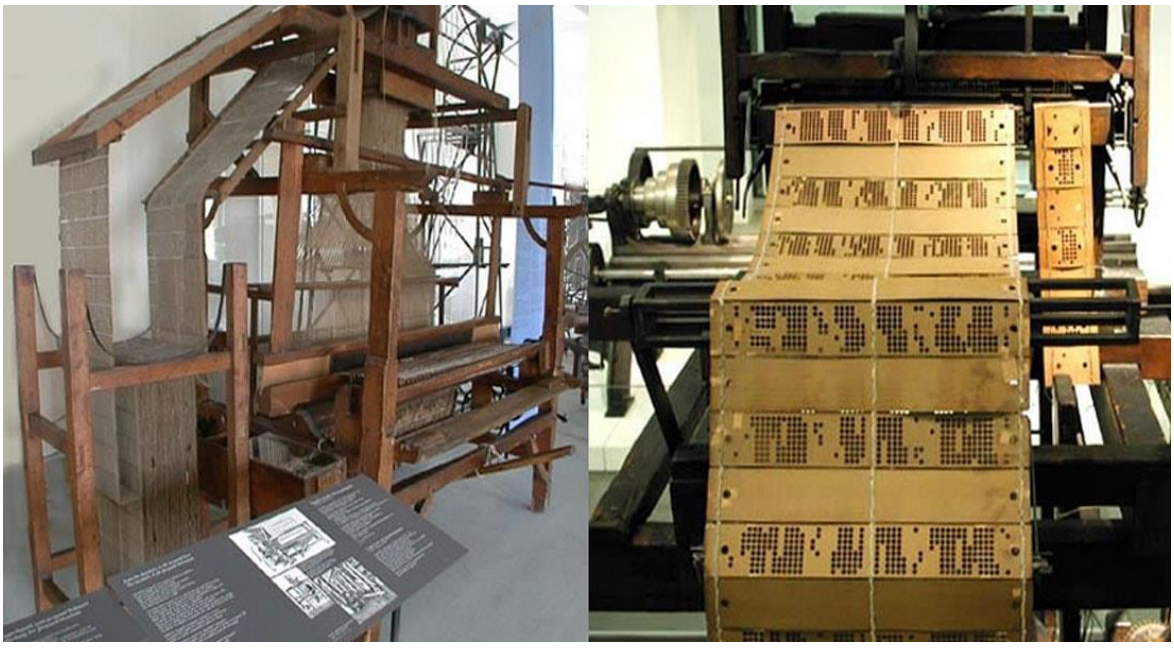
Figura 1 – Modelo de cartão perfurado.



Fonte: Trejo, 2016.

Em 1801, o tecelão e comerciante francês Joseph Marie Jacquard (1752-1834) apresentou o uso de cartões perfurados em uma máquina de tecer seda. A máquina de Jacquard, nomeada de *Jacquard Loom*, possuía um padrão de furos onde as cordas da máquina levantam ou abaixam a agulha de confecção na presença ou ausência de furos formando assim a tecelagem do tecido (CHM, 2021). A figura 2 ilustra a máquina e o mecanismo criado por Jacquard embutida nela.

Figura 2 – Máquina de tear de Jacquard.



Fonte: Cunha, 2017.

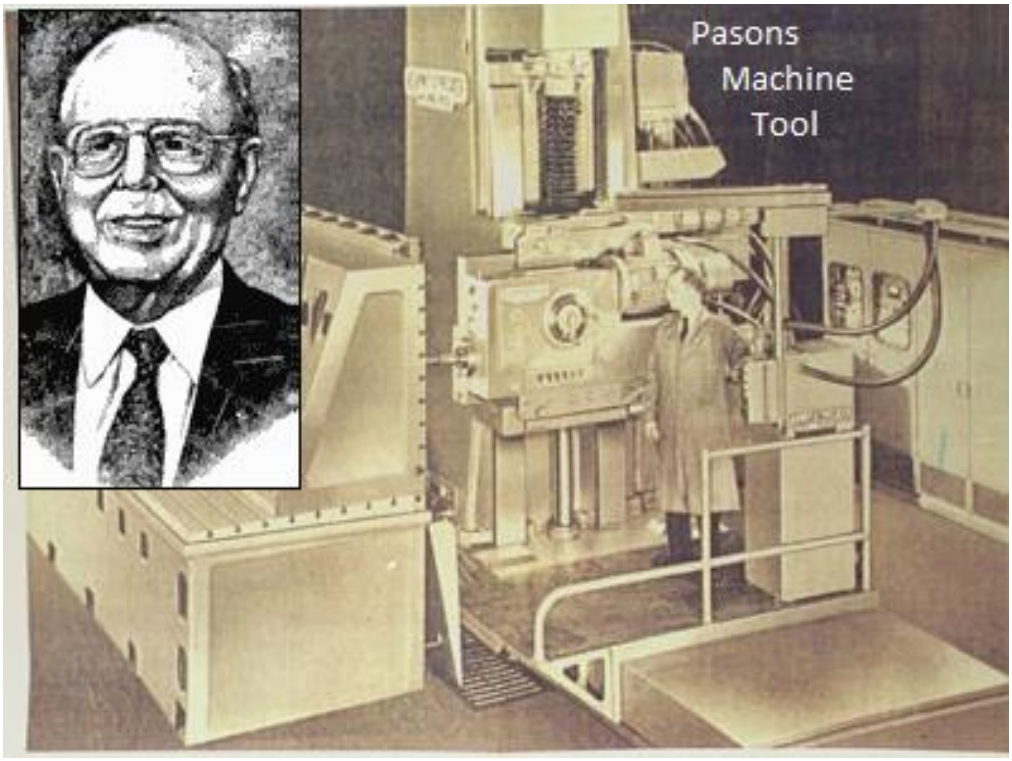
Outro uso de cartões perfurados, foi em um tocador automático de piano, criado por M. Fourneaux em 1863 que utilizava rolos de papéis com furos; conforme o ar entrava nesses furos, diferentes teclas eram ativadas formando assim uma melodia. Estas ideias de Jacquard e Fourneaux deram início a diversas inovações com uso de furos em papéis, cartolinas ou cilindros (AZEVEDO, 2008).

Inicialmente, existiam dois tipos de sistemas NC: o controle numérico computadorizado e o controle numérico direto. O controle numérico direto possuía um computador central que controlava inclusive outras máquinas, já no controle numérico computadorizado, cada computador tem seu próprio sistema em separado. Atualmente, existe o denominado controle numérico distribuído que consiste em um único computador central que controla todas as máquinas computadorizadas, permitindo maior capacidade de memória e processamento e ganho em desempenho, inclusive de manufatura (MARCICANO, 2020).

John Thoren Parsons, nascido em 1913 em *Michigan* nos EUA, foi conhecido como "Pai do Controle Numérico" por dar início a era do uso de máquinas controladas por computador, dando início a vários outros projetos importantes para a tecnologia atual (LEE, 2021).

No final de 1940, em meio a muitos trabalhos na tentativa de gerar uma curva automática em uma fresadora, Parsons teve a ideia de prover coordenadas em cartões perfurados para realizar movimentos em posições exatas. A máquina era capaz de ler esses comandos de furos e torná-los em instruções para coordenadas na máquina-ferramenta, e esta conseguia realizar pequenos incrementos, aumentando a precisão dos pontos do círculo ao desenhar (LEE, 2021). Inicialmente, a máquina foi chamada por John de “máquina fresadora *carda-mática*”, porém, insatisfeito com o nome, Parsons realizou um concurso e o vencedor ganhou $50 por dar a ideia do nome “Controle Numérico Computadorizado” (COFFLAND, 2017). A figura 3 apresenta a primeira máquina de controle numérico, criada por John T. Parsons.

Figura 3 – John T. Parsons e a primeira máquina NC com sistema de cartão perfurado.



Fonte: Smartec, 2018

Mais tarde, em 1948, Parsons apresentou a ideia para a Força Aérea dos Estados Unidos, que ficou interessada em adaptar esta tecnologia na fabricação de aviões e assim começou a apoiar John para realizar projetos dentro dos laboratórios do MIT. Em 1952, o Instituto Tecnológico lançou o primeiro protótipo NC que se consistia em uma fresadora vertical copiadora e no ano posterior, foi comprovada a eficiência do uso de sua aplicação (COFFLAND, 2017).

Com o acontecimento da segunda guerra mundial (1939 – 1945), eram necessários mais equipamentos de ponta, porém a tecnologia ainda era muito nova naquela época, as armas não tinham a mesma capacidade que as de hoje e por causa dessa necessidade, a segunda guerra mundial teve um grande impacto como fator de construção para novas tecnologias. Diante disto, Parsons teve a ideia de máquinas controladas por Controle Numérico Computadorizado e que foi o grande marco para a evolução de aviões, armas, tanques e outros. Incrementando a flexibilidade, precisão e economia no tempo fabril de produtos industriais, sem a necessidade de mão de obra, a máquina tinha praticamente todo o controle automatizado, significando que não era mais necessária tanta intervenção da mão humana (LEE, 2021).

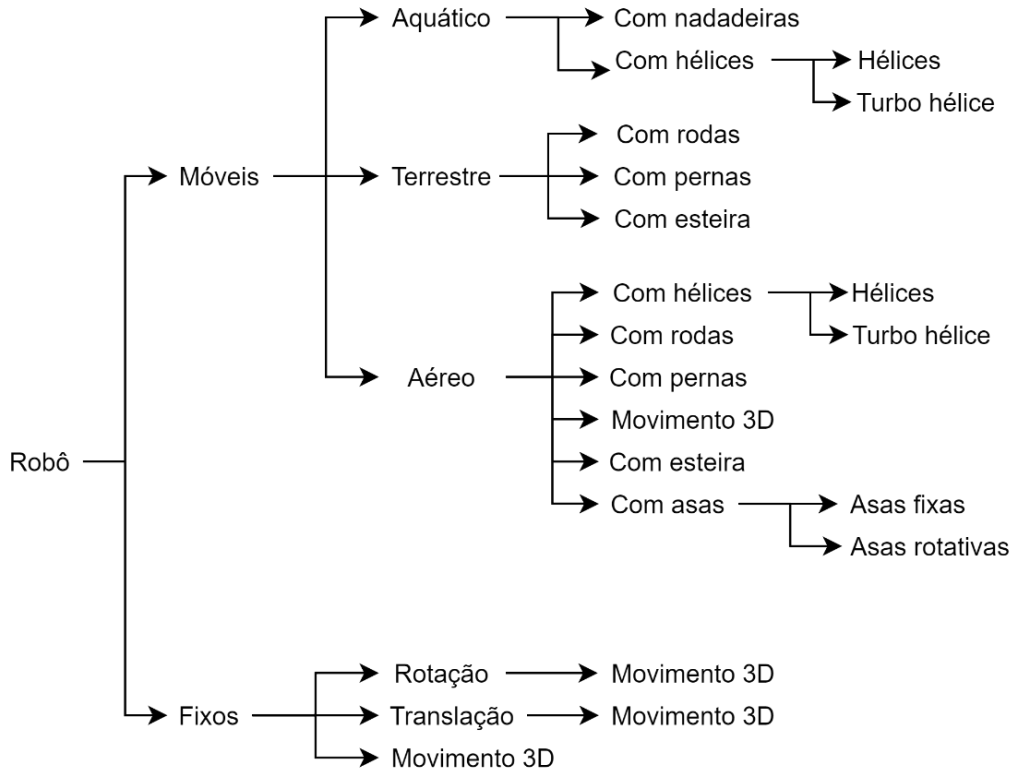
A sigla CNC, são as iniciais de um sistema bastante utilizado nas indústrias, atualmente onde a tecnologia se tornou necessária. Consistindo em um minicomputador acoplado em uma

máquina ferramenta que concede a esta ferramenta movimentos concisos para manufatura de um devido trabalho. Primeiramente, o programador responsável, cria um programa com linhas de comandos que servirão de instruções para ela. A máquina só consegue ler arquivos denominados de “Código G”, portanto, se não houver um *cross compiler* (o compilador cruzado, é o compilador que compila a linguagem para ser utilizada em outra máquina diferente da que está ocorrendo a compilação) que faça essa conversão, o programador cria as instruções nesse tipo de linguagem, permitindo assim, o controle simultâneo dos eixos X, Y e Z (TECNOLOGIA, 2014). *G-code* são criados em *softwares* de CAM (*Computer-Aided Manufacturing*) que são usados comumente para criação de desenhos 2D e 3D, pois o próprio programa CAM gera o código G a partir de desenhos CAD (*Computer-Aided Design*), sendo assim, não é necessário a intervenção humana (POLASTRINI, 2016).

2.2 Robô

As classificações dos tipos de robôs de acordo com seus movimentos são apresentadas de forma resumida na figura 4.

Figura 4 – Classificação dos tipos de robôs.



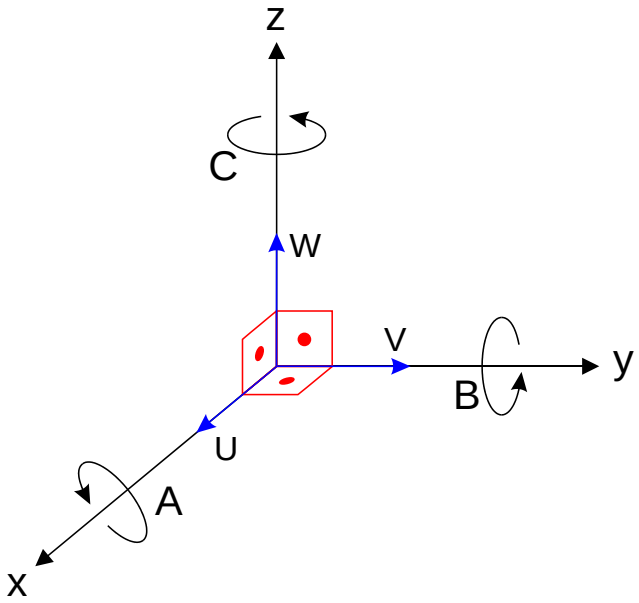
Fonte: Elaborada pela autora.

Os robôs cartesianos, também conhecidos como robôs retilíneos, possuem até três juntas deslizantes lineares (LLL) ou prismáticas (PPP) ou a combinação delas de forma a se obter três elementos e assim permitir o deslocamento em 3D dos eixos X, Y e Z perpendiculares entre si, formando e alcançando assim, a maioria dos pontos em uma área de trabalho de formato prismático, normalmente cúbico. Possuem vantagem na precisão de posicionamento do efetuador na área de trabalho (CORTÉS, 2011).

Na usinagem com CNC, algumas máquinas utilizam também de até cinco, seis ou nove diferentes eixos. Os eixos X, Y e Z que realizam movimentos em linha reta são os eixos lineares primários do controle numérico. Os eixos denominados de A, B e C são os eixos de rotação primários e os últimos U, V e W são eixos lineares secundários (FITZPATRICK, 2013).

Os eixos denominados de rotação, rotativos ou giratórios A, B e C se movimentam em torno de um dos eixos lineares primários X, Y ou Z, respectivamente (POLASTRINI, 2011). Os eixos U, V e W, também chamados de lineares auxiliares são destinados à produção multiaxial para adição de função de usinagem (FITZPATRICK, 2013). A figura 5 mostra os eixos lineares e os respectivos eixos giratórios A com X, B com Y ou C com Z.

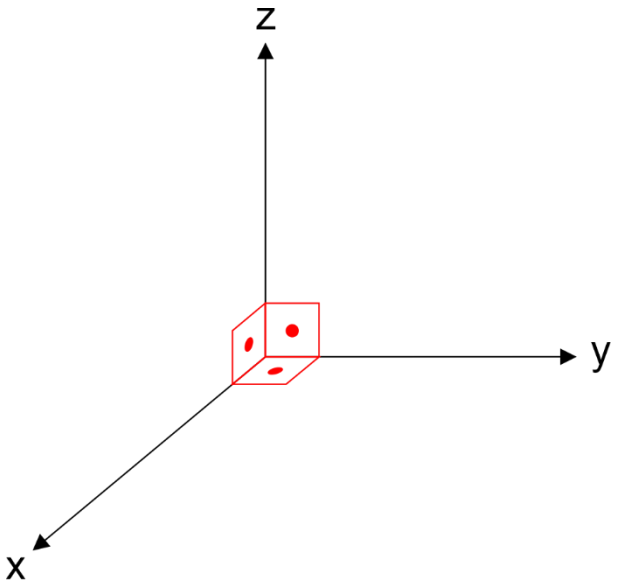
Figura 5 – Eixos lineares primários, auxiliares e eixos rotativos.



Fonte: Elaborada pela autora.

Para se referenciar um robô como cartesiano, o presente trabalho utilizará o plano cartesiano de apenas três eixos primários básicos X, Y e Z tri ortogonais. A figura 6 mostra os eixos lineares primários X, Y e Z que serão usados no trabalho (FITZPATRICK, 2013).

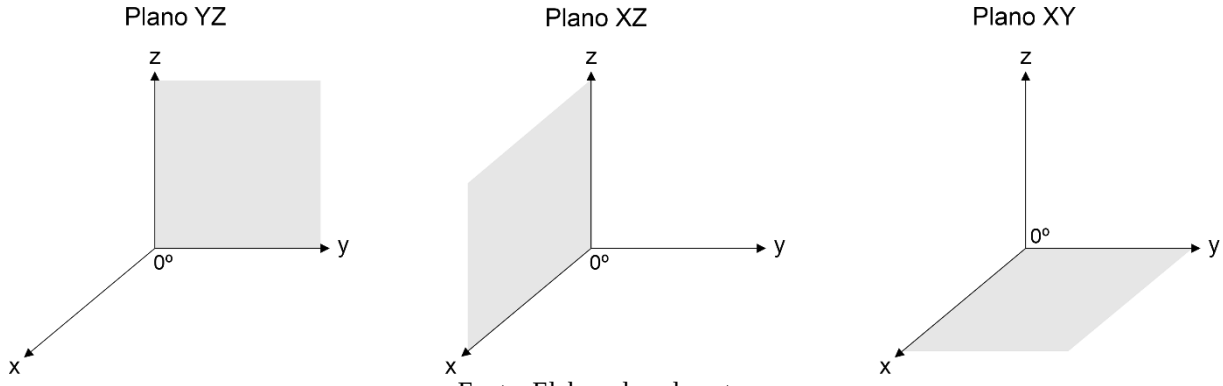
Figura 6 – Eixos lineares primários: X, Y e Z.



Fonte: Elaborada pela autora.

Os eixos primários formam um plano ortogonal (90°) entre si e são divididos em três planos formados conjuntamente com os eixos primários respectivos, sendo eles: XY, XZ e YZ. A figura 7 mostra os três planos de eixos lineares primários (FITZPATRICK, 2013).

Figura 7 – Três planos lineares primários: XY, XZ e YZ.



Fonte: Elaborada pela autora.

2.3 Código G

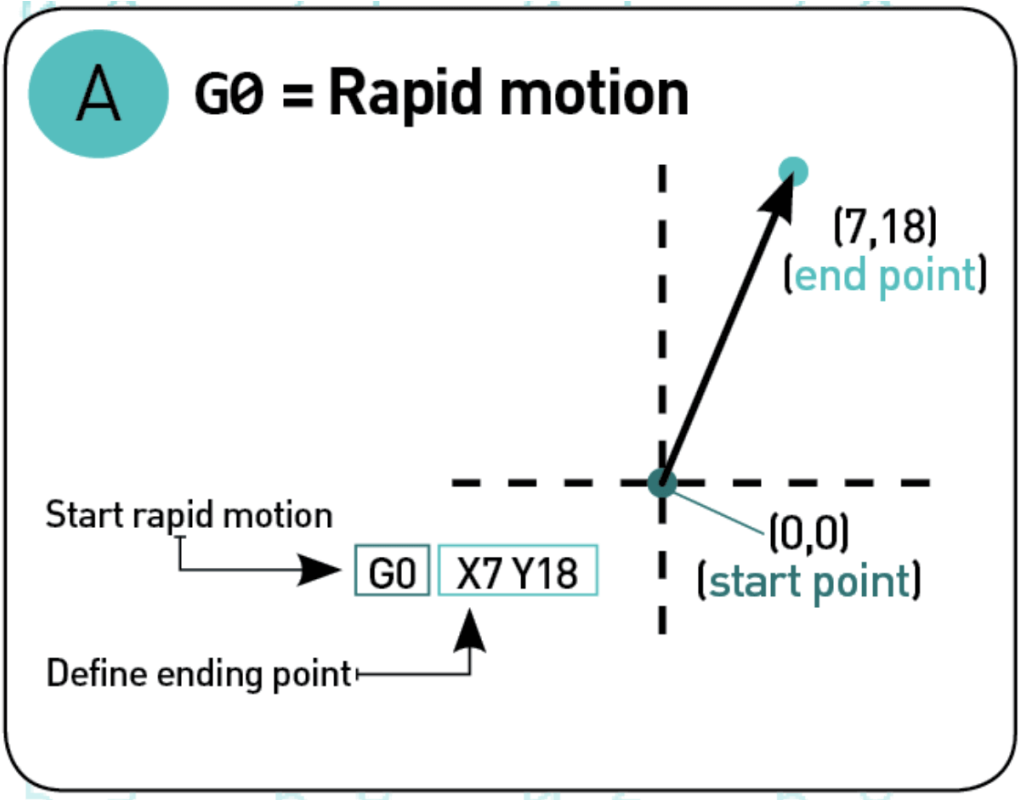
O código G, derivado de Código Geométrico, também conhecido como *G-code*, é uma linguagem de programação de fácil manuseio, criado especificamente para máquinas CNC. Esse código fornece ao cabeçote da máquina instruções de como, quando e onde mover-se geometricamente de forma tridimensional, ou seja, nos eixos x, y e z (CHAKRAVORTY, 2020).

A devida linguagem, foi implementada pela primeira vez pela *Electronics Industry Association* (EIA) em 1960, para dar instruções as máquinas de controle numérico computadorizado. Pelo fato das máquinas antigamente não terem capacidade de muita memória, o *G-code* foi criado para ser simples e conciso em suas operações, com instruções básicas de escritas para leitura na CNC. Antes de ser conhecido hoje pelo nome, em português, Código G, essa linguagem foi inicialmente chamada de RS-274D. Porém, pelo fato de muitas linhas de seus códigos serem inicializadas com a letra G, a linguagem passou a ser conhecida como "Código G" (DEANS, 2018).

A linguagem usa um mecanismo simples de leitura onde a máquina lê o código e executa os passos de forma sequencial, uma linha após a outra, para criação dos movimentos. O exemplo de código "G01 X1 Y1 F20 T01 M03 S500", diz a máquina que ela deve executar um movimento linear para as coordenadas x e y, e as letras F, T, M e S, que significam respectivamente, a taxa de alimentação da máquina, qual ferramenta será usada, o estado do fuso da máquina e a velocidade com que ela fará o serviço (DEANS, 2018).

A figura 8, representa como seria o deslocamento partindo do ponto *start point* e alcançando o *end point*, descrito no código G, usando o movimento rápido G0.

Figura 8 – Exemplo do código "G0 X7 Y18".



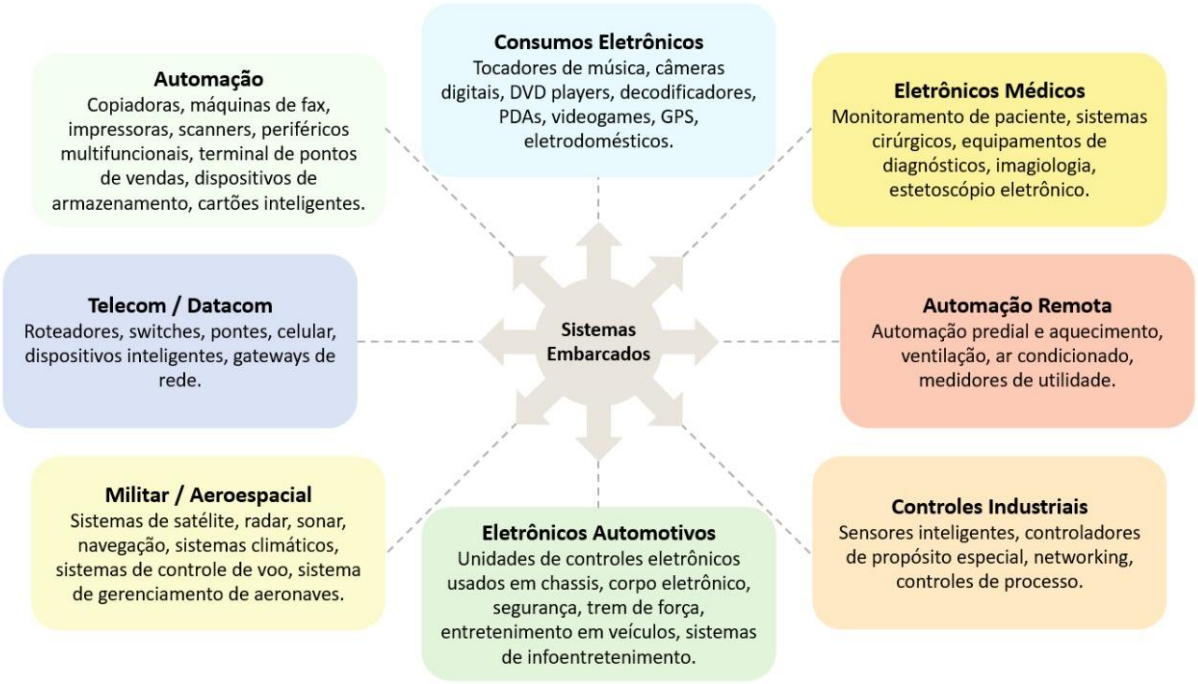
Fonte: Chakravorty, 2020.

2.4 Sistema Embarcado

Sistemas embarcados, sistemas embutidos ou em inglês *embedded systems*, são sistemas que contêm programas alocados ou embutidos em microprocessadores ou microcontroladores que executam um conjunto de tarefas específicas predefinidas pelo programador em um determinado dispositivo (WOLF, 2012). Os sistemas embarcados são controlados por um microprocessador ou microcontrolador encapsulado no corpo deste dispositivo, de forma a realizar uma tarefa específica definida pelo usuário e programador. Sendo assim, pode-se considerar também como um sistema microprocessado (POZZEBOM, 2014).

Os sistemas embarcados são conhecidos por embutir um minicomputador ou microcontrolador no seu dispositivo. Têm se tornado cada vez mais presentes no dia a dia, porém muitos desses dispositivos passam despercebidos ao usuário, como por exemplo impressoras, semáforos e smartphones que são considerados sistemas microprocessados por possuírem memória, processador e muitas outras partes naturalmente embutidas de um computador (WOLF, 2012). A figura 9 apresenta algumas áreas de aplicação de sistemas embarcados.

Figura 9 – Áreas de aplicação de sistemas embarcados.

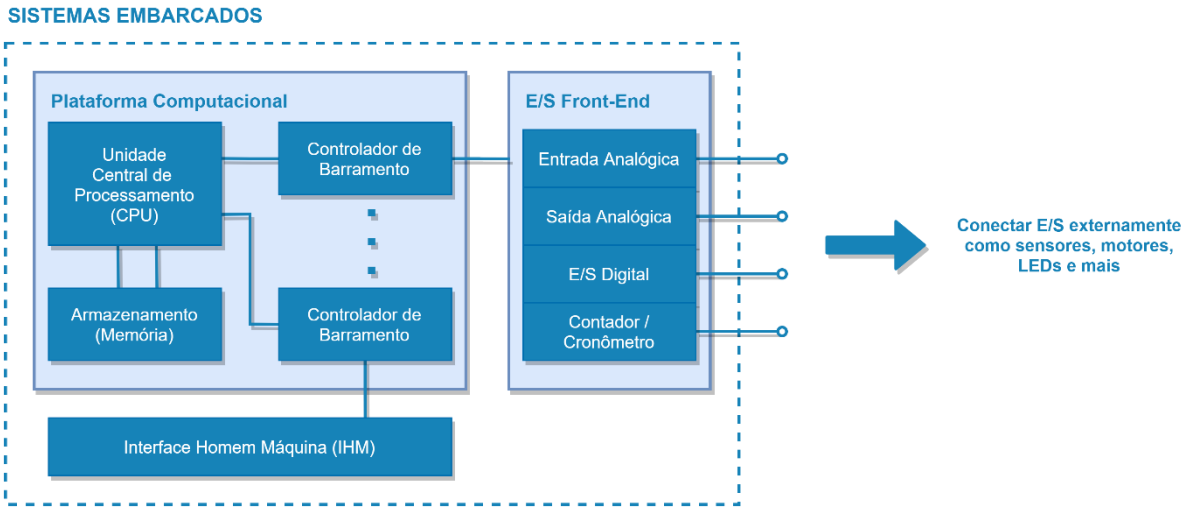


Fonte: Elaborada pela autora.

Por esta razão, considera-se que um sistema embutido tenha um minicomputador acoplado a sua placa, é assim, a definição de diversos aparelhos eletrônicos como um sistema embarcado. Eles possuem a vantagem de poder operar em máquinas que trabalham durante longos períodos, como é o caso de semáforos que ficam ligados 24h por dia (WOLF, 2012).

Os sistemas baseados em microcontroladores (MCU) contêm a unidade de processamento (CPU), memórias e interfaces *input* e *output* (I/O); já os sistemas baseados em microprocessadores (MPU), diferente dos MCU, contêm apenas a unidade de processamento; memórias e interfaces I/O precisam ser conectados ao MPU (FLORENCIO, 2017). A figura 10 apresenta um diagrama de blocos de um sistema embarcado.

Figura 10 – Diagrama de blocos sistemas embarcados.



Fonte: Elaborada pela autora.

O primeiro sistema embarcado a existir é o AGC (*Apollo Guidance Computer*), um computador responsável pelo controle das espaçonaves Apollo que operava a 1,024 MHz, fornecendo interfaces eletrônicas para navegação, orientação e controle da espaçonave (LIMA, 2014).

Inventado pelo laboratório de Instrumentação do MIT (*Massachusetts Institute of Technology*) e fabricado pela Raytheon a partir de 1966, o AGC foi feito, totalmente, com circuitos integrados discretos compostos de portas NOR de 3 entradas. O primeiro microprocessador em encapsulamento único, foi lançado pela Intel em 1971, por isso o uso de CIs (Circuitos Integrados) no AGC (LIMA, 2014).

Para a criação da mini *plotter* artesanal, será necessário embarcar um sistema de *software* e expansão *shield* no microcontrolador Arduino Uno para a operação desejada da máquina.

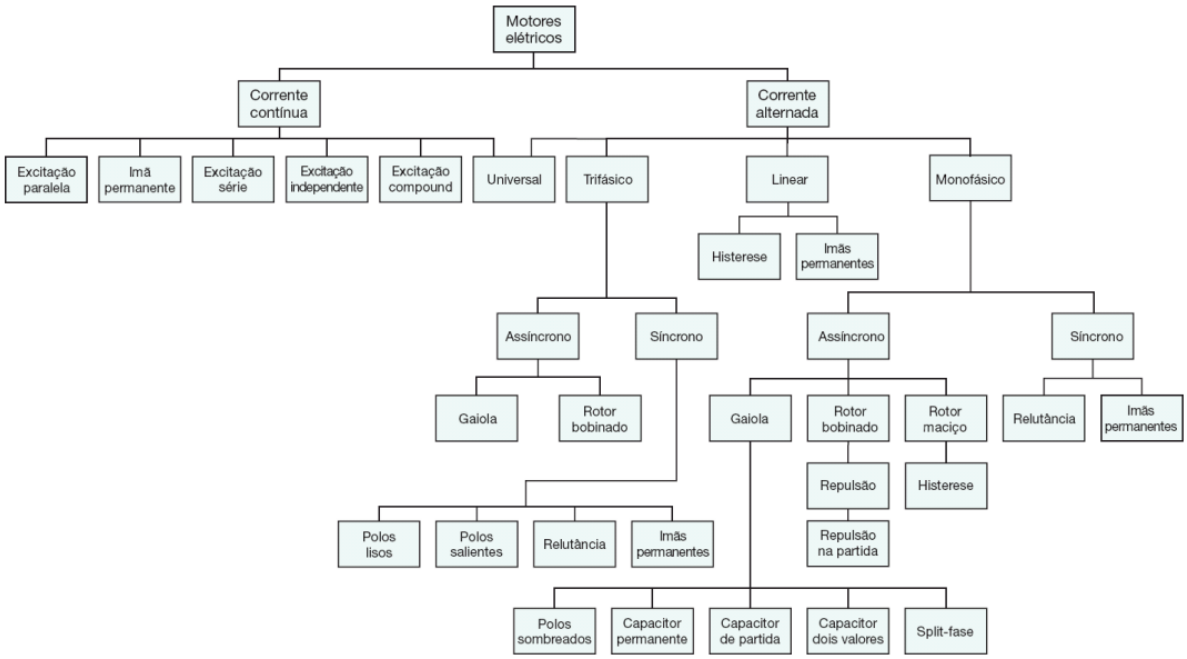
2.5 Motores

O motor elétrico tem a função de converter energia elétrica em energia mecânica e dentre todos os outros tipos de motores pneumáticos e hidráulicos, sendo que o motor elétrico é o mais utilizado. Ele foi criado por Michael Faraday em 1821 e se move de forma contínua ou como no caso do presente trabalho, em deslocamentos discretos (RODRIGUES; DELMONDES, 2016).

Os motores elétricos se movimentam através de forças eletromagnéticas, ou seja, pelo princípio do eletromagnetismo. Seus condutores estão localizados em um campo magnético, em que a junção da força eletromagnética mais a corrente elétrica gera uma força mecânica fazendo o motor girar. Essa força, chamada de torque, realiza movimentos contínuos à medida que a corrente elétrica é invertida de sentido; isso pode propiciar a inversão do sentido de rotação do motor (RODRIGUES; DELMONDES, 2016).

Os motores elétricos são divididos em dois tipos quanto a sua forma de tensão: o motor de Corrente Contínua (CC) e o motor de Corrente Alternada (CA). Os dois usam o princípio do campo eletromagnético, mas no de corrente contínua o fluxo da corrente é de forma ordenada, onde todos os elétrons se movimentam na mesma direção, já na corrente alternada o sentido da corrente varia ciclicamente com o tempo (MAMEDE FILHO, 2017). A figura 11 apresenta a classificação dos motores elétricos.

Figura 11 – Classificação dos motores elétricos.



Fonte: MAMEDE FILHO, 2017.

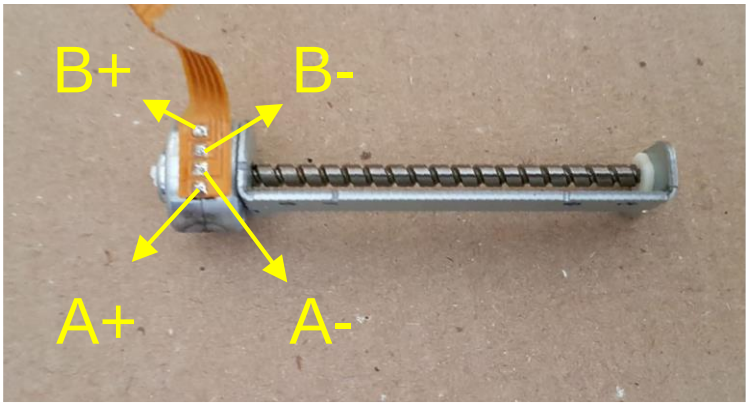
2.5.1 Motor de passo

Após a revolução industrial, com a necessidade da modernização das máquinas e com o intuito de atender diferentes aplicações, surge a necessidade de controlar com exatidão a posição dos eixos dos motores e nesse propósito, o francês Marius Lavet, desenvolveu em 1936 o motor de passo, ou em inglês, *stepper motor* (RODRIGUES; DELMONDES, 2016).

Assim como os motores elétricos que transformam uma forma de energia em outra, os motores de passo são dispositivos eletromecânicos que também transformam sinais elétricos digitais em torque, convertendo assim, energia elétrica em mecânica e realizando a movimentação em passos no eixo (RODRIGUES; DELMONDES, 2016).

O motor de passo é um dispositivo síncrono, cuja velocidade de rotação é proporcional a frequência de sua alimentação, capaz de girar em um número específico de graus, 7,5° ou 15° a cada pulso, movendo-se em passos. Eles recebem os dados do computador e esses sinais devidamente condicionados, são transformados em movimentos nos eixos, por exemplo da CNC ao qual é acoplado (RODRIGUES; DELMONDES, 2016). Um dos motores utilizados para o projeto é apresentado na figura 12, tendo este de ímã permanente.

Figura 12 – Motor de passo bipolar utilizado no projeto.



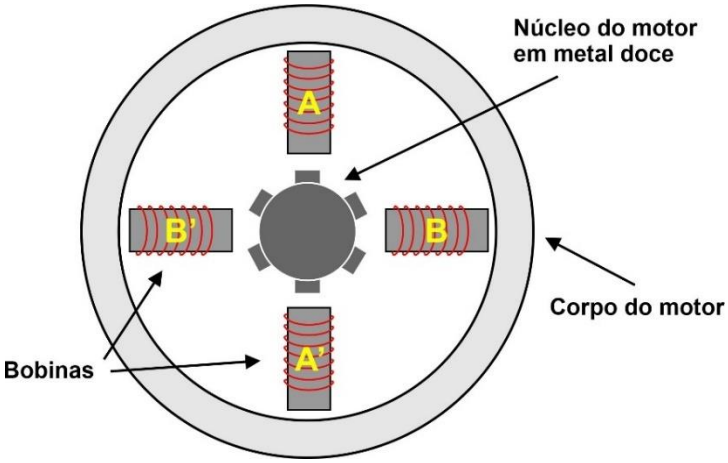
Fonte: Elaborada pela autora.

Os motores de passo são divididos em três tipos construtivos: ímã permanente, relutância variável e híbrido. O tipo ímã permanente possui um rotor com ímã permanente, o motor de passo de relutância variável possui um rotor de ferro doce e o tipo híbrido usa a combinação de ímã permanente e relutância variável (CHAPMAN, 2013).

2.5.1.1 Relutância variável

O motor de passo de relutância variável possui estator de metal férreo laminado com polos com ranhuras, sendo o rotor dentado. Esses dentes metálicos são utilizados para melhorar o caminho magnético, melhorando a relutância do circuito magnético, vem daí o nome do motor. Quando energizado, o rotor gira e quando os polos do rotor se alinham com os do estator, significa que o motor rotacionou um passo. Os polos do motor possuem alguns alinhamentos entre si; quanto mais alinhamentos, mais preciso o motor é ou mais passos ele possui. A figura 13 apresenta um motor de relutância variável sendo AA’ e BB’ duas fases (RODRIGUES; DELMONDES, 2016).

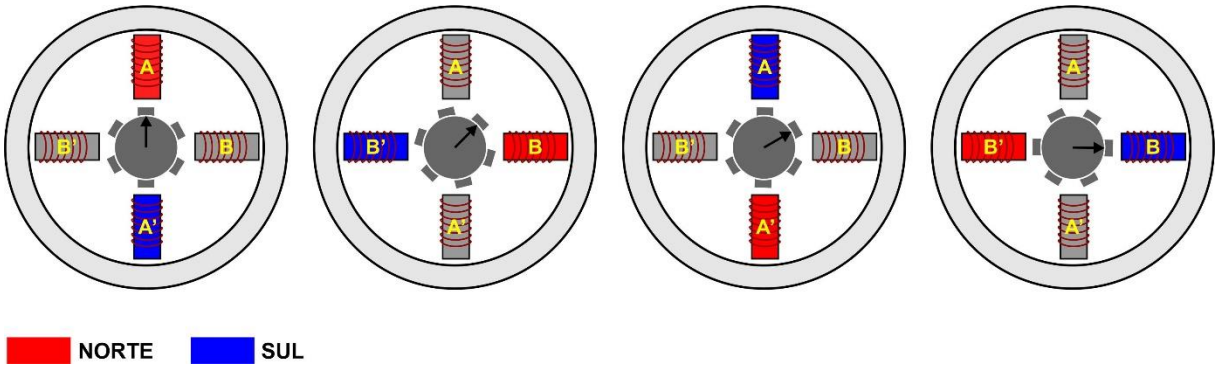
Figura 13 – Motor de passo de relutância variável com duas fases.



Fonte: Adaptado de RODRIGUES; DELMONDES, 2016.

Quanto ao seu funcionamento, os motores de relutância variável possuem dentes no rotor e esses dentes do eixo são alinhados com os polos do estator. Quando o desalinhamento ocorre, o motor sendo energizado, retorna para o alinhamento mais próximo. Isso faz com que o motor de relutância variável possa girar em passos (RODRIGUES; DELMONDES, 2016). A figura 14 apresenta o princípio de funcionamento do motor de passo de relutância variável. Observa-se na imagem quatro etapas simulando o funcionamento do motor.

Figura 14 – Princípio de funcionamento do motor de passo de relutância variável.

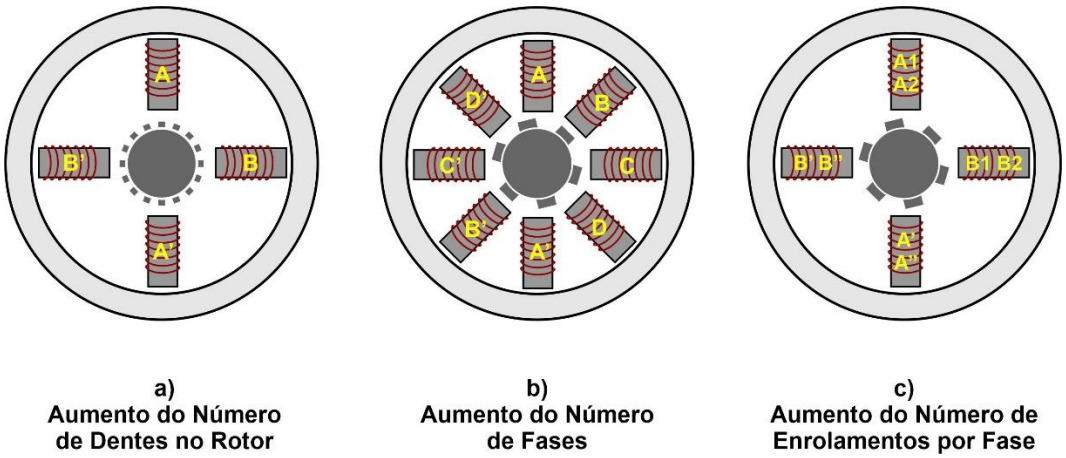


Fonte: Adaptado de RODRIGUES; DELMONDES, 2016.

A resolução do motor de relutância variável pode ser aumentada através do aumento dos dentes no rotor ou pelo aumento do número de fases (RODRIGUES; DELMONDES, 2016). A figura 15 apresenta os três métodos possíveis para aumento da resolução. A figura 15-a apresenta o aumento no número de dentes no rotor. A figura 15-b apresenta o aumento no número de fases, onde aumentou as bobinas e tem mais dois tipos de pares, CC’ e DD’. A figura

15-c apresenta o aumento do número de enrolamento por fase e é possível observar que aumentou a quantidade de bobinas, mas os pares continuaram A1A2 e A'A'' ou B1B2 e B'B''.

Figura 15 – Aumento da resolução do motor de passo de relutância variável.

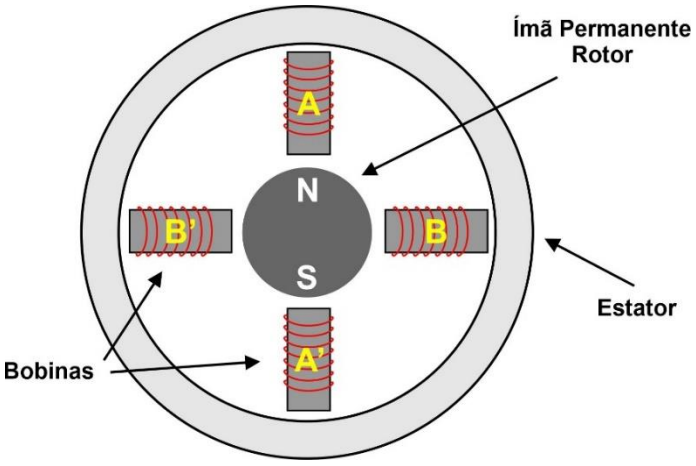


Fonte: Elaborada pela autora.

2.5.1.2 Ímã Permanente

O motor de ímã permanente, possui um rotor construído com ímãs permanentes; ele mantém a última posição atingida sem perturbações, mesmo não estando energizado. Os polos do rotor e do estator possuem uma quantidade menor de alinhamento comparado ao motor de relutância variável, tornando-o menos preciso. Neste tipo de motor, o rotor tende a girar até que os campos magnéticos fiquem alinhados. A figura 16 apresenta o motor de passo com ímã permanente sendo AA' e BB' duas fases (RODRIGUES; DELMONDES, 2016).

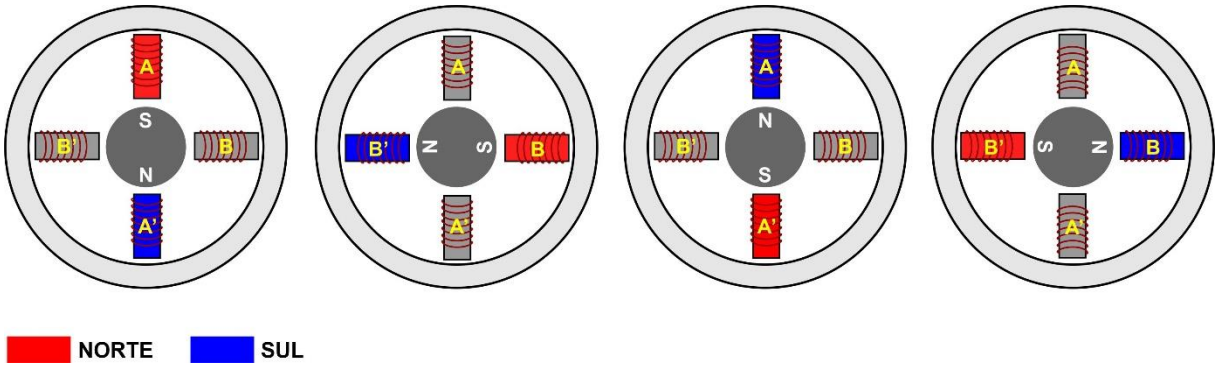
Figura 16 – Motor de passo com ímã permanente.



Fonte: Adaptado de RODRIGUES; DELMONDES, 2016.

Quando o motor é energizado, a bobina do estator é ativada fazendo com que o eixo se alinhe com o campo magnético presente no ímã permanente do rotor, até que a próxima fase seja energizada. A figura 17 apresenta o princípio de funcionamento do motor de passo com ímã permanente (RODRIGUES; DELMONDES, 2016).

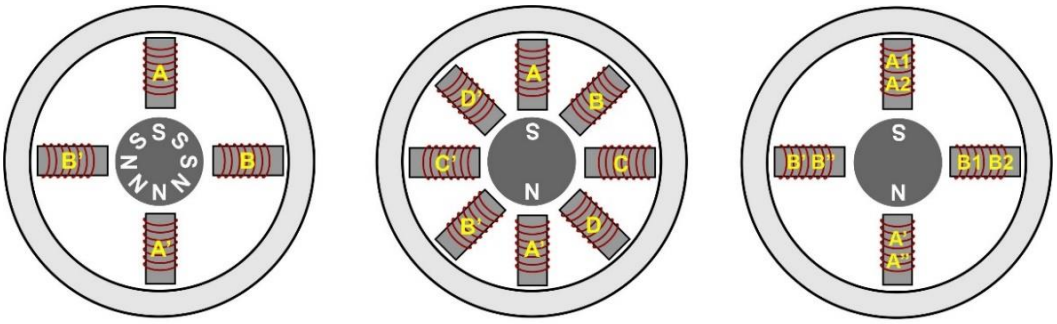
Figura 17 – Princípio de funcionamento do motor de passo com imã permanente.



Fonte: Adaptado de RODRIGUES; DELMONDES, 2016.

A resolução do motor de ímã permanente pode ser aumentada através do aumento no número de polos no rotor ou pelo aumento do número de fases (RODRIGUES; DELMONDES, 2016). A figura 18 apresenta os três métodos para aumentar a resolução do motor de passo. A figura 18-a apresenta o aumento do número de polos de rotor, observando que no ímã permanente há mais polos (Norte e Sul). E as figuras 18-b e 18-c apresentam o aumento no número de fases e o aumento do número de enrolamentos por fase, respectivamente.

Figura 18 – Aumento da resolução do motor de passo de ímã permanente.

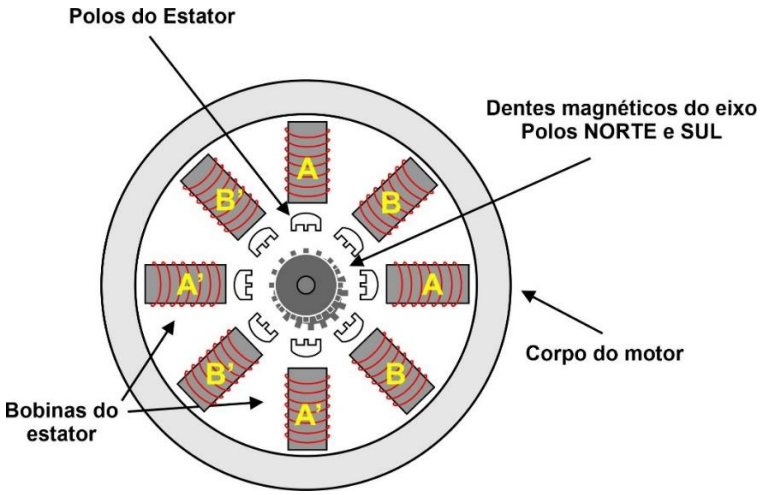


Fonte: Elaborada pela autora.

2.5.1.3 Híbrido

Os motores híbridos podem ser de quatro fases, sendo este o mais comum, de três ou cinco fases também. Por ser a junção do motor de ímã permanente com o motor de relutância variável, de forma a obter as melhores características dos dois tipos de motores. A figura 19 apresenta o motor de passo híbrido (RODRIGUES; DELMONDES, 2016).

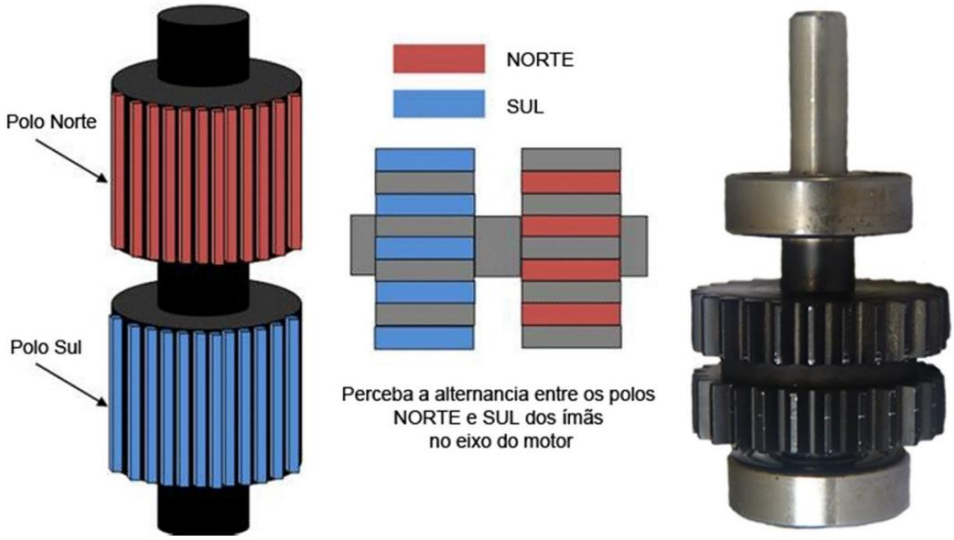
Figura 19 – Motor de passo híbrido.



Fonte: Adaptado de RODRIGUES; DELMONDES, 2016.

O eixo do rotor do motor de passo híbrido possui dois grupos de dentes, sendo um o polo Norte e outro o polo Sul, de modo que os dentes fiquem alternados. A figura 20 apresenta o eixo do motor híbrido (RODRIGUES; DELMONDES, 2016).

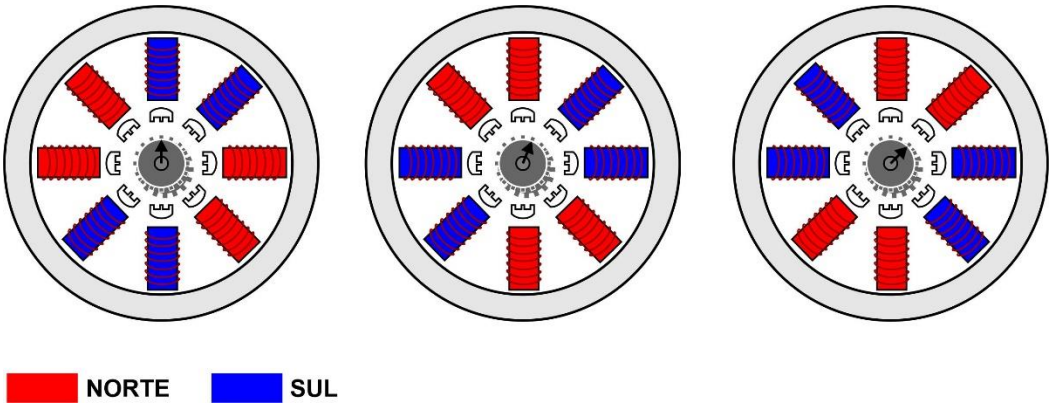
Figura 20 – Eixo do motor híbrido.



Fonte: RODRIGUES; DELMONDES, 2016.

O funcionamento do motor híbrido é semelhante ao de relutância variável e ímã permanente. As bobinas precisam ser ativadas em sequência para o eixo do motor poder girar. A figura 21 apresenta o funcionamento do motor de passo híbrido.

Figura 21 – Funcionamento motor de passo híbrido.



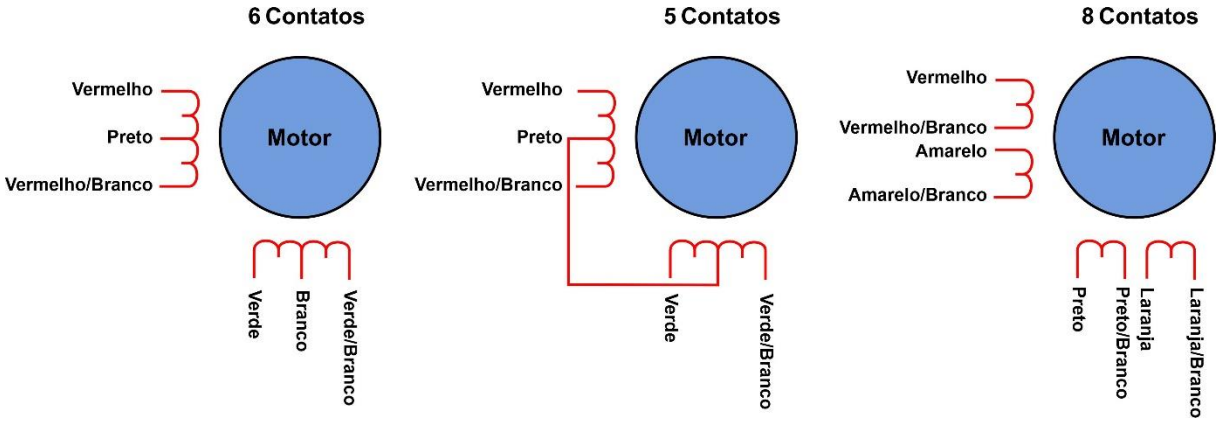
Fonte: RODRIGUES; DELMONDES, 2016.

2.5.1.4 Unipolar e Bipolar

Os motores de passo podem ser também do tipo unipolar ou bipolar, a forma de ligação dos enrolamentos estatóricos destes motores é o que caracteriza que eles sejam unipolares ou bipolares (RODRIGUES; DELMONDES, 2016).

Os unipolares geralmente possuem dois enrolamentos por fase para cada sentido da corrente e contatos em comum de cinco, seis ou oito conexões (RODRIGUES; DELMONDES, 2016). A figura 22 apresenta essas conexões.

Figura 22 – Motor de passo unipolar.

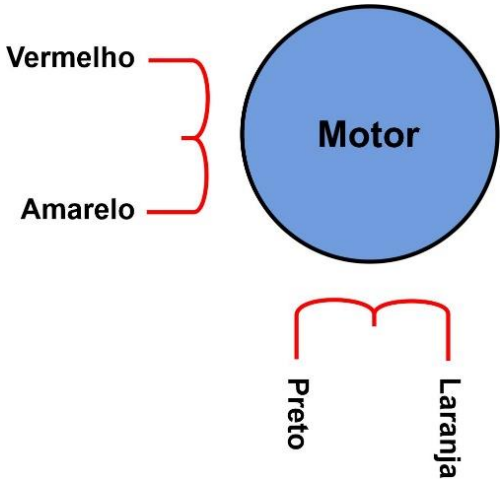


Fonte: Adaptado de RODRIGUES; DELMONDES, 2016.

Nos motores de seis contatos, a conexão dos dois polos é separada, os de cinco contatos a conexão comum é soldada internamente e os de oito contatos possuem conexão comum dos dois polos separados (RODRIGUES; DELMONDES, 2016).

O motor bipolar possui um único enrolamento por fase, usando ligação por polo e para reverter o sentido da corrente é necessário que o circuito de controle acione as bobinas de forma a inverter seu sentido de circulação de corrente, daí o nome do motor (RODRIGUES; DELMONDES, 2016). A figura 23 apresenta um motor bipolar.

Figura 23 – Motor de passo bipolar.



Fonte: Adaptado de RODRIGUES; DELMONDES, 2016.

3 PROPOSTA DE SOLUÇÃO

Este capítulo apresenta a proposta de solução do referido trabalho de conclusão de curso.

3.1 Componentes de *hardware*

Esta seção descreve os componentes de *hardware* para a criação do trabalho.

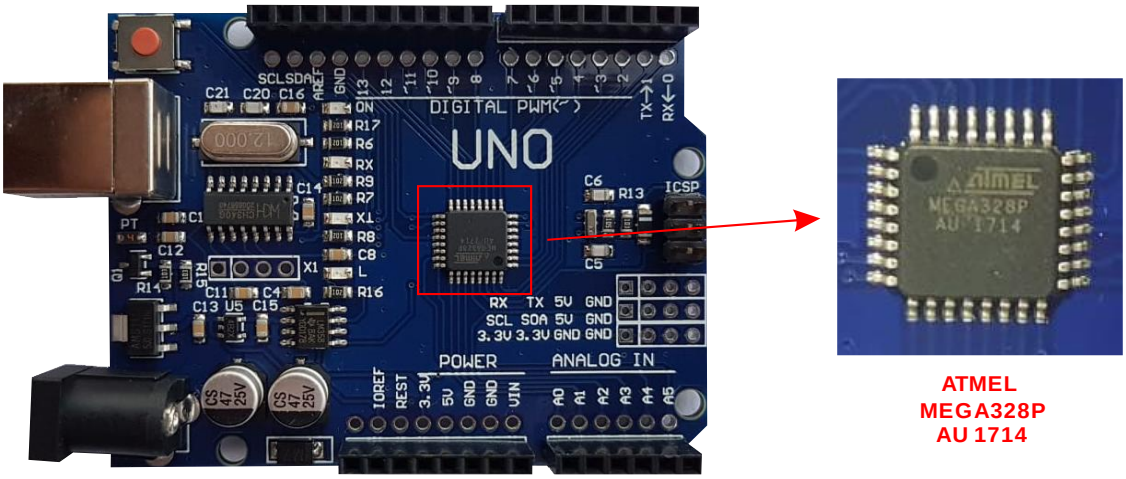
3.1.1 Arduino Uno

Um microcontrolador consiste em um pequeno computador, onde são concentradas todas as funções principais em um único chip de circuito integrado (CI). Com um ou mais núcleos de processadores e memória, os microcontroladores são usados em produtos que realizam seu trabalho de forma autônoma e possuem periféricos de entrada e saída (E/S), que podem ser programáveis para tal finalidade desejada. (ARAUJO; CAVALCANTE; SILVA, 2019). Para o referente trabalho, é utilizado o microcontrolador Arduino Uno.

O Arduino Uno é uma placa de microcontrolador que possui código aberto, *software* livre, um circuito físico programável. Ele é baseado no microcontrolador *Microchip ATmega* e foi desenvolvido pela Arduino e o Uno é um tipo de placa dessa família Arduino. O microcontrolador citado possui um conjunto de pinos de entrada e saída que são programáveis com o *software* Arduino IDE usando a linguagem de programação C++ com pequenas modificações. Sua entrada é um cabo USB do tipo B e pode ser alimentado tanto pelo cabo USB quanto por uma bateria externa com tensão elétrica entre 7V e 12V. O Arduino Uno é o sistema mais utilizado da série Arduino, e usado muitas vezes em institutos educacionais por sua fácil manutenção e integração com diversos projetos eletrônicos, incluindo as máquinas CNC e por isso, o uso dele é tão importante nesse trabalho (POLASTRINI, 2011).

O Arduino usado no trabalho, possui um microcontrolador MEGA328P da Atmel acoplado nele. Este é considerado o elemento mais importante da placa, é julgado como o cérebro do circuito por receber todas as informações atribuídas a ele e interagir de diversas formas. Pode-se utilizar dispositivos de entrada e saída como motores, LED, botão entre outros e possibilitar a ele instruções de interação com o projeto do usuário (BOXALL, 2013). A figura 24 apresenta a placa de Arduino usada e o referido microcontrolador.

Figura 24 – Arduino Uno e o microcontrolador da Atmel.

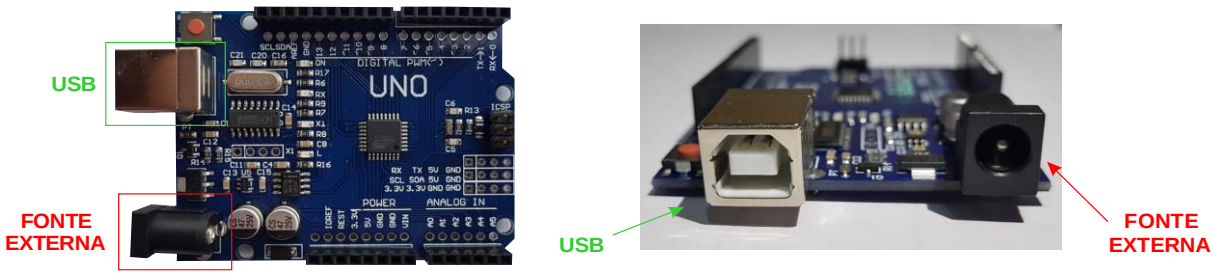


Fonte: Elaborada pela autora.

3.1.1.1 Entradas

O Arduino Uno possui duas entradas para alimentação: a entrada USB (*Universal Serial Bus*) e a Fonte Externa. A Fonte Externa possui conector tipo Jack e sua tensão de entrada tem de estar entre 7V e 12V. Se a tensão da placa estiver abaixo de 7V, representa uma instabilidade para o regulador de tensão. Por esse motivo, a fonte de alimentação adequada para a fonte externa está entre 7V e 12V (POLASTRINI, 2011). A entrada USB de tipo B serve tanto para alimentar a placa, quanto para enviar e receber dados, e é através desta entrada que o presente trabalho irá enviar as instruções para placa. A figura 25 apresenta os dois tipos de alimentação citados acima (BOXALL, 2013).

Figura 25 – Dois tipos de alimentação do Arduino Uno.

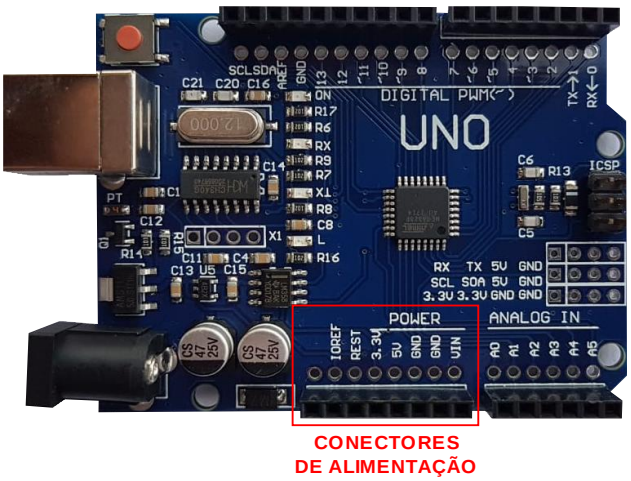


Fonte: Elaborada pela autora.

A placa de microcontrolador também possui 7 conectores de alimentação, sendo eles: um IOREF, um RESET, um 3,3V, um 5V, dois GND e um VIN. A entrada IOREF serve como

uma referência de tensão para que o determinado circuito selecione a fonte de alimentação adequada. O RESET serve para resetar o microcontrolador externamente, as entradas de 3,3V e 5V fornecem a tensão para a placa, os GNDs são pinos de terra e o VIN é a tensão de entrada da placa quando se usa uma bateria externa (RODRIGUES; DELMONDES, 2016). A figura 26 mostra os conectores de alimentação da placa.

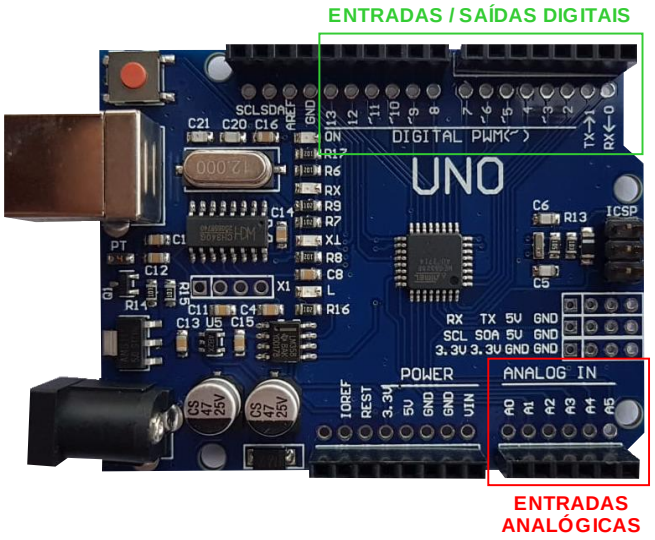
Figura 26 – Conectores de alimentação presentes no Arduino UNO.



Fonte: Elaborada pela autora.

O Arduino possui seis pinos para entradas analógicas e quatorze pinos para entradas ou saídas digitais, dos quais seis podem ser também usados para entradas de valores analógicos, são eles os pinos 3, 5, 6, 9, 10 e 11 que asseguram uma saída PWM de 8 bits, variação da intensidade de um dispositivo usando-se a função *analogWrite*(), como por exemplo aumenta a velocidade do motor (BANZI, 2011). Os pinos digitais podem ser tanto para entrada quanto para saída, sendo somente importante informar na programação se será de entrada ou saída, usando as funções *pinMode*(), *digitalWrite*() ou *digitalRead*() (RODRIGUES; DELMONDES, 2016). Os pinos 0 e 1 também podem ser utilizados para comunicação porta serial (*serial port*) e têm a função de receber (RX) e transmitir (TX) dados para outros dispositivos e para o computador através da comunicação via USB (BOXALL, 2013). Os pinos 10, 11, 12 e 13 podem também ser destinados à comunicação SPI (comunicação entre dois Arduino) (POLASTRINI, 2011). A figura 27 apresenta os pinos analógicos e digitais.

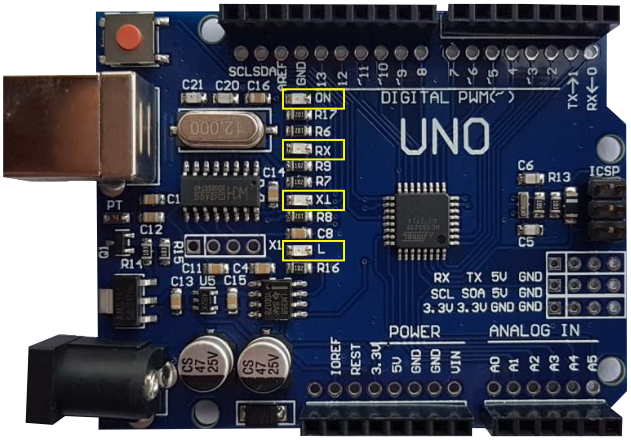
Figura 27 – Pinos analógicos e digitais do Arduino Uno.



Fonte: Elaborada pela autora.

A placa também possui quatro LEDs, (*Light-Emitting Diodes*, em português, Diodos Emissores de Luz). Um, denominado ON, indica se há ou não corrente na placa, informando assim, se está ou não funcionando. Os LEDs TX e RX acendem quando há carregamento de dados, tanto transmitindo quanto recebendo pela porta serial USB ou dispositivos. Por último, o LED L é para uso próprio e está ligado ao pino 13, localizado nas entradas/saídas digitais (BOXALL, 2013). A figura 28 destaca os LEDs presentes na placa Arduino Uno.

Figura 28 – LEDs: ON, RX, TX e L.

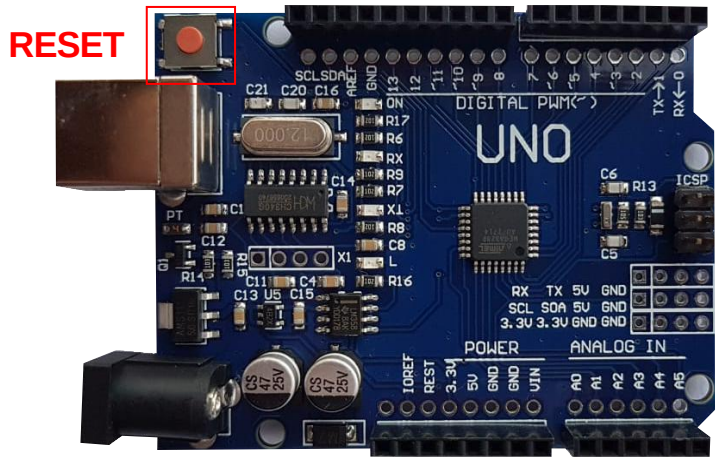


Fonte: Elaborada pela autora.

O botão *reset* do Arduino serve para resetar o sistema, em especial quando o sistema estiver com problemas ou necessitar de uma reinicialização, ele é de suma importância; basta

resetar a placa que é feito o *reset* (BOXALL, 2013). A figura 29 apresenta o botão de *reset* da placa de microcontrolador.

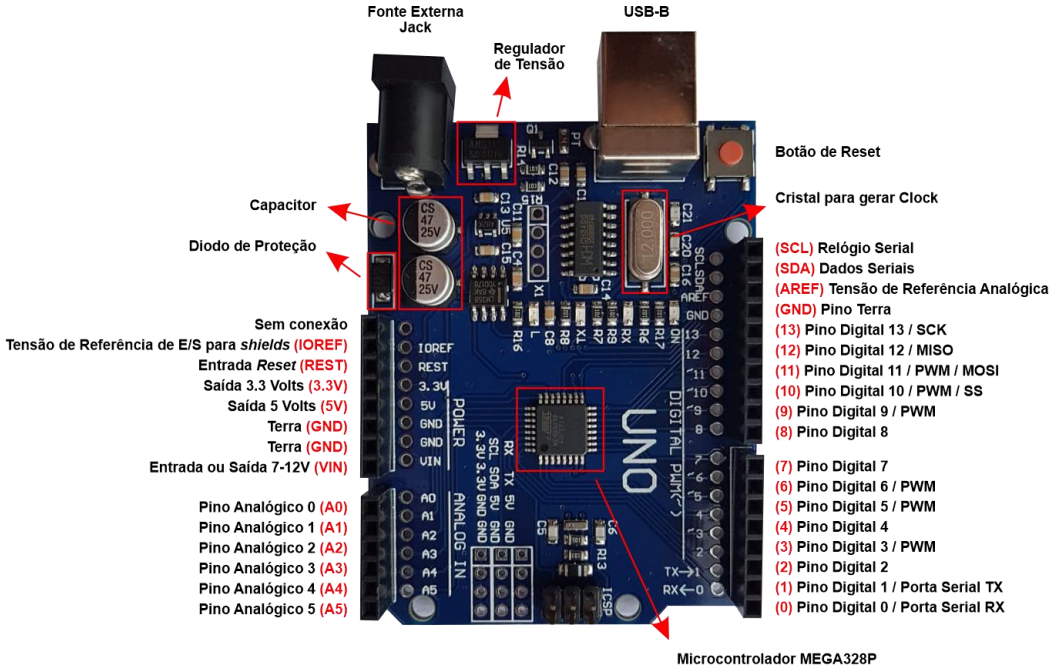
Figura 29 – Botão de *Reset* do Arduino Uno.



Fonte: Elaborada pela autora.

A figura 30 apresenta um resumo das pinagens e componentes pertencentes a placa Arduino Uno a ser usada nesse projeto.

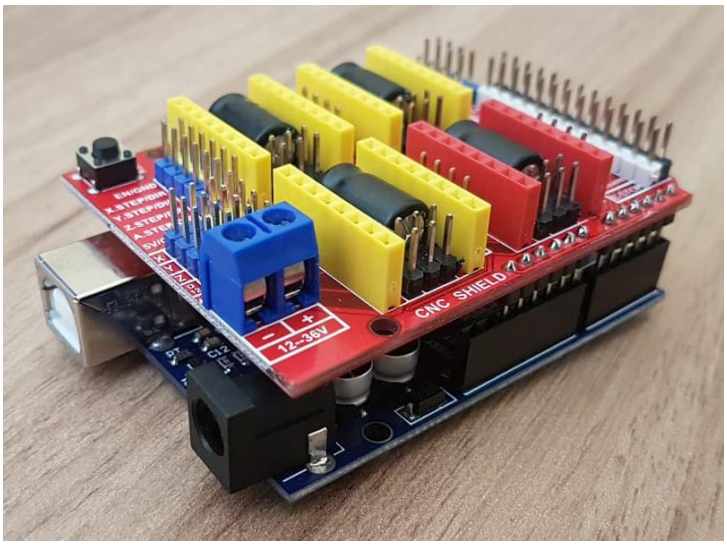
Figura 30 – Pinagens e componentes Arduino Uno.



Fonte: Elaborada pela autora.

A placa de microcontrolador Arduino Uno também permite a expansibilidade com *shields*, adicionando assim, mais funções de *hardware* (POLASTRINI, 2011). A *shield* deve ser conectada nas pinagens do Arduino como apresentado na figura 31. A *shield* CNC permite ao Arduino Uno aumentar função para controle numérico computadorizado, o qual só é capaz com a expansão dela e o presente trabalho se utiliza dessas duas.

Figura 31 – Arduino Uno com expansão da *shield* CNC.



Fonte: Elaborada pela autora.

3.1.2 CNC Shield V3

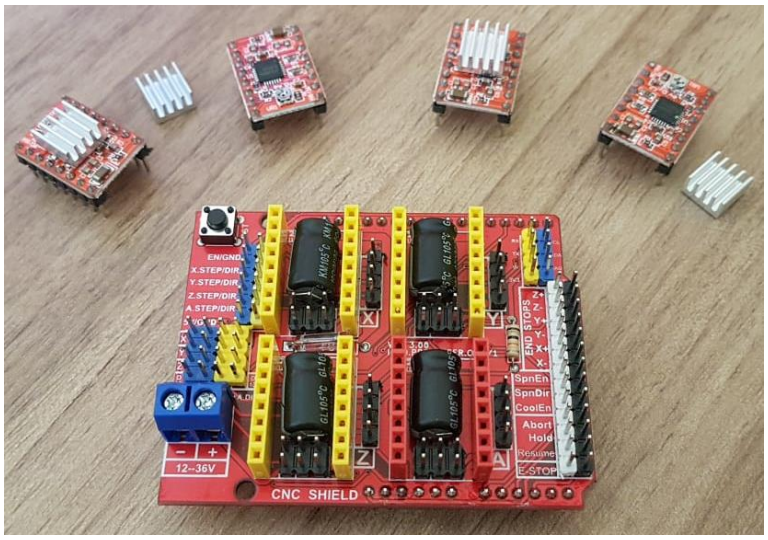
A expansão do Arduino CNC *Shield* é conectada a placa Arduino energizando e acionando os motores de passo para realizar todas as funções necessárias na operação da CNC. É possível usar controladores ao invés do *shield*, porém estes são mais baratos, portanto, mais acessíveis aos projetos menores. Existe diferentes placas CNC e neste trabalho é usado o tipo *open-source* V3.51, que possui uma fácil usabilidade e uma simples configuração, sendo assim mais hábil o uso desta placa para vários projetos de Controles Numéricos Computadorizados. (METZ, 2020).

Para criação da mini CNC artesanal, o *shield* faz integração com o Arduino Uno como tipo de expansão, requisitando uma fonte de alimentação entre 12V e 36V para acionamento de seus quatro motores de passo *plug-in* A4988. GRBL, acrônimo de *Garble*, é um *firmware* de controlador CNC que trabalha junto ao Arduino para funcionamento na máquina. Assim, o *shield* executa o *firmware* GRBL do Arduino na CNC pelo fato de o GRBL ser facilmente

programado para o Arduino e possui vários periféricos de entrada e saída para a ligação desejada e exata do equipamento (METZ, 2020).

Movimentos precisos são realizados na máquina que possui acionamentos simultâneos dos eixos X, Y e Z, os quais são configurados inicialmente através do código G carregado na placa. Com suporte para quatro eixos, o CNC *Shield* V3 também possui o eixo A para duplicar os eixos X, Y e Z, caso seja necessário. Porém, para este trabalho são usados somente três motores A4988 para serem plugados nos *slots* dos eixos X, Y ou Z, três dimensões. O CNC *Shield* V3 vem equipado com quatro motores de passo e quatro dissipadores de calor. Por ser um projeto pequeno, não será preciso o uso de um *cooler* para esfriamento de todo o processo (HARDWARE, 2018). A figura 32 apresenta o CNC *Shield* usada no trabalho e os quatro *drivers* A4988.

Figura 32 – CNC *Shield* mais 4 motores A4988.



Fonte: Elaborada pela autora.

3.2 Componentes de *Software*

Esta seção descreve os componentes de *software* para a criação do trabalho.

3.2.1 Arduino IDE

Arduino *Integrated Development Environment* (IDE) é um ambiente de desenvolvimento escrito na linguagem C, C++ e Java que permite enviar comandos ao microcontrolador Arduino através de códigos, também conhecido como *sketches*, compilados

nesse editor e geralmente salvos em formato “.ino” (ARDUINO, 2021). Neste trabalho o micro não trabalha com a extensão “.ino” e sim com a extensão GRBL para ser possível a tradução do código G e para isso é instalado o *firmware* GRBL na plataforma IDE.

A IDE é uma plataforma simples onde só é preciso conectar o microcontrolador desejado através de um cabo USB no computador, selecionar o micro ao qual está usando e em seguida deve compilar o programa que automaticamente é enviado para a placa. Caso seja usado uma placa não inclusa no histórico de placas do Arduino IDE, é possível baixar o modelo da placa para que o programa a reconheça na sua base de dados (ARDUINO, 2021).

Algoritmo ou programa, é a forma de dizer ao computador o que ele precisa fazer, um caminho com instruções definidas. Hoje em dia, eles são programados em linguagem de alto nível e compiladores transformam esse código em linguagem de máquina para o equipamento reconhecer o que lhe está sendo instruído. O ambiente de desenvolvimento Arduino IDE é o responsável por compilar esse programa e permitir uma ampla variação de códigos, criação de funções, variáveis e outras coisas para assim criar diferentes *sketches* para a programação do Arduino Uno (ARDUINO, 2021).

3.2.2 Java

Diferente da linguagem de programação Java Script, o Java é uma plataforma computacional gratuita que permite executar sistemas que sem ele não funcionaria. Para quase todos os aparelhos eletrônicos, é necessário ter o Java instalado e atualizado para a execução correta dos aplicativos que é utilizado e depende dele (SCHILDT, 2019). Neste trabalho é usada uma plataforma *G-code*, UGS, a qual é necessário o uso juntamente ao Java para seu funcionamento, por isso seu uso é mencionado neste trabalho de conclusão de curso.

Assim como todos os aplicativos, o Java atualiza constantemente, sendo essas atualizações fundamentais para solucionar *bugs* de versões anteriores ou simplesmente melhorar a experiência do usuário com a plataforma. Como as diversas atualizações em sites, navegadores de internet, sites bancários, jogos e demais aplicações, o computador, sem o Java atualizado, fica incapacitado de obter uma boa navegação. É também utilizado para melhorar o desempenho e a segurança de todas as aplicações relacionadas (SCHILDT, 2019).

Como o UGS é um programa baseado em Java, é necessário que tenha este instalado, se não, não é possível abrir a plataforma e é necessário que este mantenha-se atualizado para um melhor funcionamento do aplicativo.

3.2.3 Universal G-code Sender

O *Universal G-code Sender* ou simplesmente UGS, é uma plataforma *G-code* baseada em Java, portanto se o usuário não tiver o Java instalado, ele não funciona. O UGS possui recursos para fazer interface com controladores CNC compatível com GRBL. Usando-se o Java, o GRBL permite a configuração do CNC *Shield* para o controle desejado da máquina. Sua função é traduzir o código G em movimento do motor, porém o Arduino não reconhece o algoritmo do código G. Portanto, é preciso, primeiramente, instalar uma extensão do *firmware* GRBL no Arduino IDE. Desta forma, é possível o microcontrolador Uno ler os arquivos GRBL e o *Universal G-code Sender* funciona como um mediador para enviar, através dele, o código G ao Arduino Uno e assim o micro têm todas as funções para qual foi programado (POLASTRINI, 2011; RODRIGUES; DELMONDES, 2016).

3.2.4 Inkscape

O *Inkscape* é um editor de gráficos vetoriais de código aberto usado na criação de *designs* gráficos no formato SVG "*Scalable Vetor Graphics*" - formato de arquivo livre para gráficos vetoriais. Com vários recursos para manipulação de objetos, o *software* permite rodar em Windows, Mac e Linux. É possível importar e exportar diversos formatos de arquivo e permite instalar extensões como por exemplo o *G-code,* que permite o uso do desenho da máquina CNC. Com a instalação do plugin *G-code,* o desenho é convertido para o formato ".ngc" e assim o controlador consegue ler ele corretamente, para assim poder realizar as corretas instruções a quem lhe foi dado (*INKSCAPE*, 2020).

3.2.5 CAMotics

CAMotics é um programa gratuito *open-source* que simula os três eixos *G-code* de uma CNC. Com uma interface simples, é possível visualizar os resultados dos desenhos em 3D, além da possibilidade de iniciar, pausar, reiniciar ou avançar as simulações e ter também a visualização do objeto 3D em diferentes perspectivas. O CAMotics é criado exclusivamente para ter uma pré-visualização do desenho antes de ir para a máquina, evitando assim erros que podem acontecer na hora da impressão, economizando tempo e dinheiro (CAMOTICS, 2021).

4 ORGANIZAÇÃO

A organização do trabalho está dividida em *hardware* e *software* e descreve os processos utilizados em cada uma das partes.

4.1 *Hardware*

A seção 4.1 corresponde aos *hardwares* é dividida em três partes: motores e eixos, estrutura e microcontroladores.

A seção 4.1.1 apresenta motores e eixos e os materiais utilizados para a construção da estrutura do protótipo.

A seção 4.1.2 apresenta a estrutura e os métodos utilizados para a construção do protótipo.

A seção 4.1.3 apresenta o microcontrolador e como é utilizado o CNC *Shield* no protótipo.

4.1.1 Motores e Eixos

Para a construção do projeto, é utilizado *drivers* de CD-ROM e um *driver* de disquete. É possível utilizar alguns componentes como os motores para movimentação linear e a carcaça dos *driver*s de CD para reutilização na estrutura do projeto. As figuras 33 e 34 apresentam respectivamente os dois *drivers* de CD-ROM e o *driver* de disquete usados no projeto.

Figura 33 – *Driver* de CD-ROM.

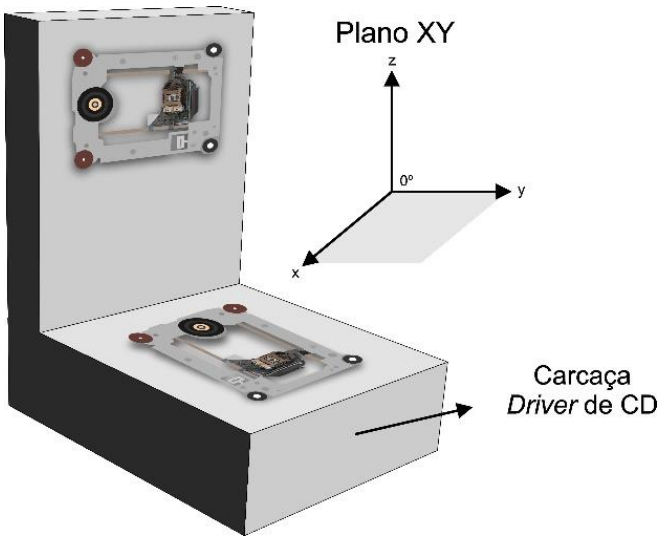
Fonte: Elaborada pela autora.

Figura 34 – *Driver* de disquete.

Fonte: Elaborada pela autora.

Para a movimentação dos eixos X e Y, é utilizado dois *drivers* de CD-ROM, um para o X e outro para o Y indicando-os para movimentação linear na horizontal e vertical respectivamente. E para o eixo Z, utiliza-se do *driver* de disquete, o que permite movimentação linear para cima e para baixo. A figura 35 apresenta o protótipo desenvolvido para o projeto, nele as demonstrações dos eixos X, Y e Z.

Figura 35 – Protótipo da mini CNC artesanal.

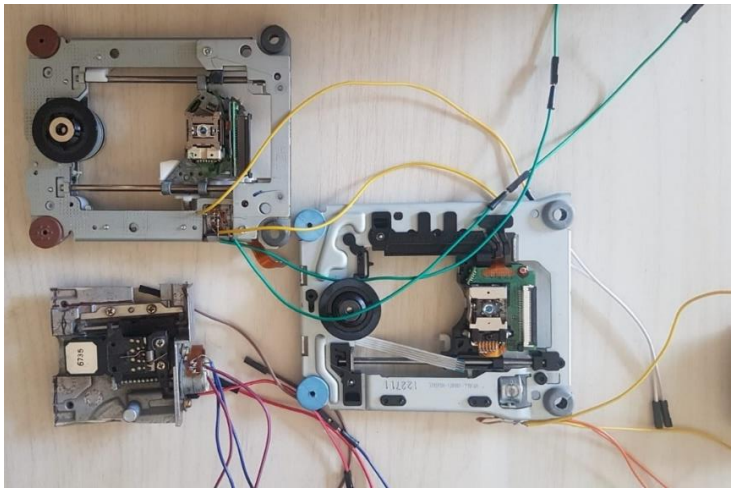


Fonte: Elaborada pela autora.

Os *drivers* são desmontados para se utilizar as partes estruturais interessantes para o projeto. Uma parte totalmente aproveitada para o presente trabalho, são os motores que possuem quatro entradas, separadas aos pares. Com o multímetro, é possível verificar qual bobina é a utilizada, através do teste de continuidade. Esse apita para indicar a bobina de interesse correta ou fica estático caso esteja na errada, essa prática se destina a verificar se há algum problema no dispositivo ou não.

Para cada dupla de entradas do motor, é soldado jumpers macho-macho de mesma cor para melhor discernimento na realização da conexão no CNC *Shield* e conectado jumpers fêmea-fêmea em sua ponta, então assim é realizado para os três motores dos eixos. A figura 36 apresenta as três partes já soldadas com os jumpers.

Figura 36 – Estrutura lineares dos eixos.

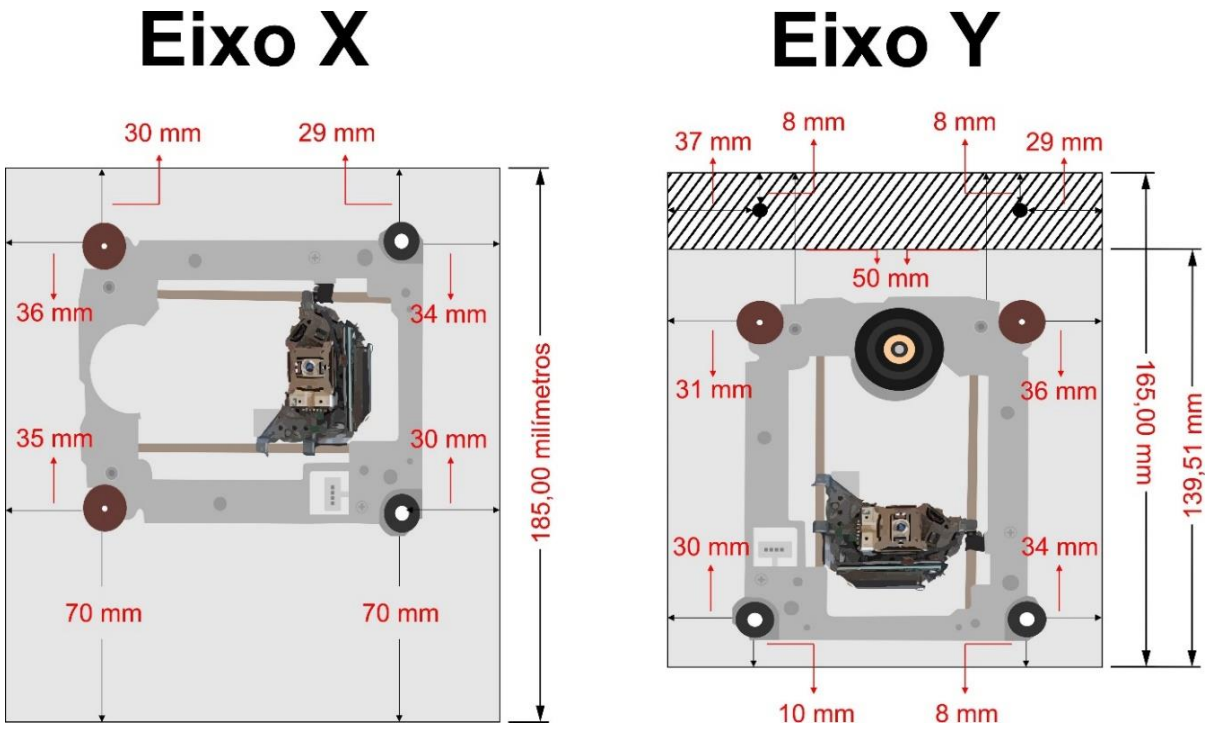


Fonte: Elaborada pela autora.

4.1.2 Estrutura

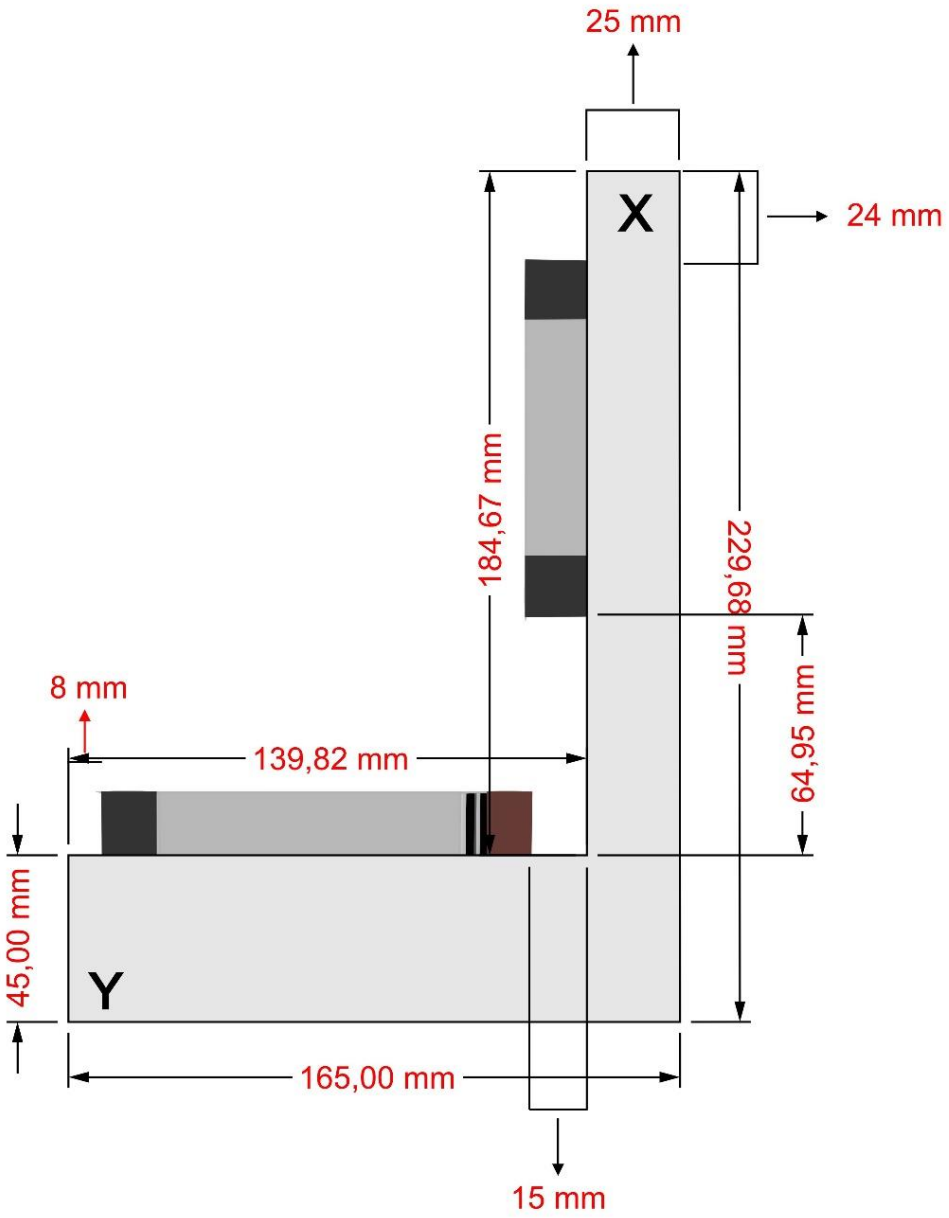
Para montar a estrutura do protótipo, é utilizado da armação de dois *drivers* de CD-ROM, já exemplificado na figura 33. Antes de realizar a montagem da estrutura de fato, realiza-se um projeto para cálculo de distanciamento das peças com base em suas dimensões, visando atingir exatamente em que ponto perfurar a base para que não haja nenhum equívoco ao juntar as peças. As figuras 37 e 38 apresentam um pré-projeto em 2D das dimensões necessárias.

Figura 37 – Distanciamento das placas na estrutura em vista frontal e superior.



Fonte: Elaborada pela autora.

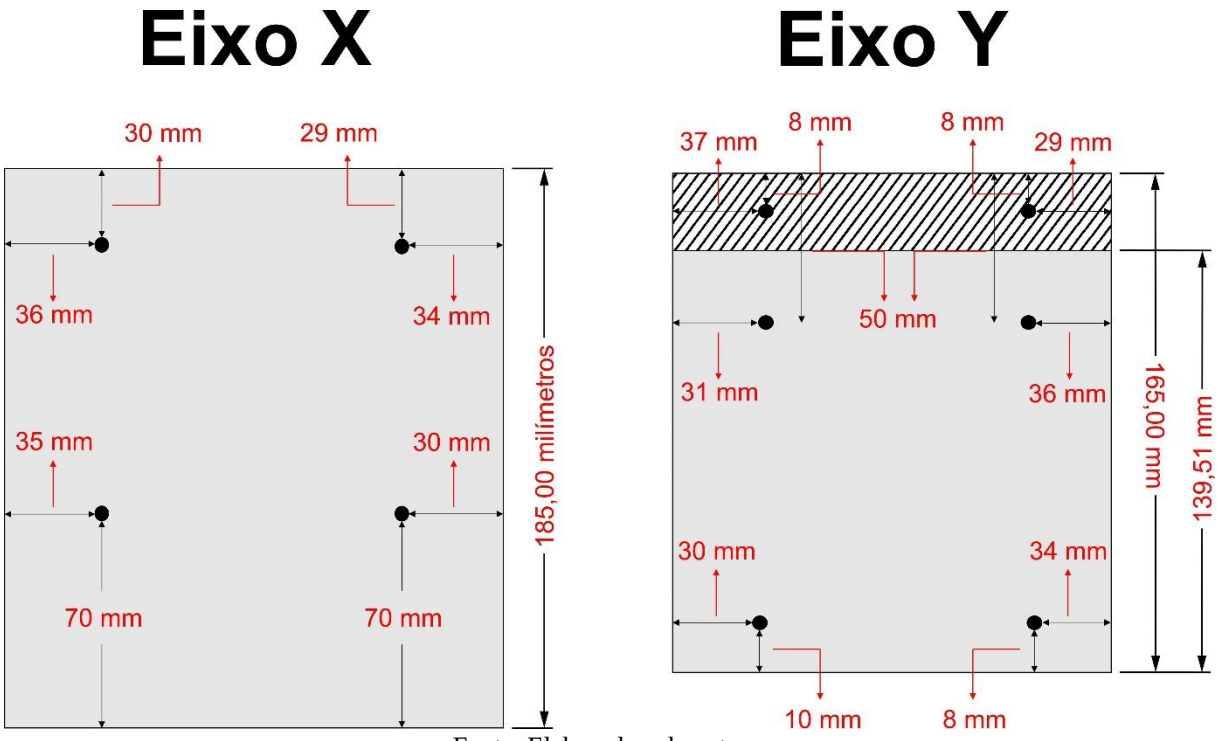
Figura 38 – Dimensões da estrutura do projeto em vista lateral.



Fonte: Elaborada pela autora.

Semelhante a figura 37, desenvolveu-se outra figura somente com esquema de perfuração. A figura 39 apresenta as duas peças a serem perfuradas com as respectivas marcações devidamente distanciadas. As indicações de pequenos círculos em preto significam as perfurações para fixação dos eixos X e Y e os pequenos círculos localizados na parte listrada remetem aos orifícios para fixação da carcaça, são usados dois arrebites para fixação destes.

Figura 39 – Marcações dos furos.



Fonte: Elaborada pela autora.

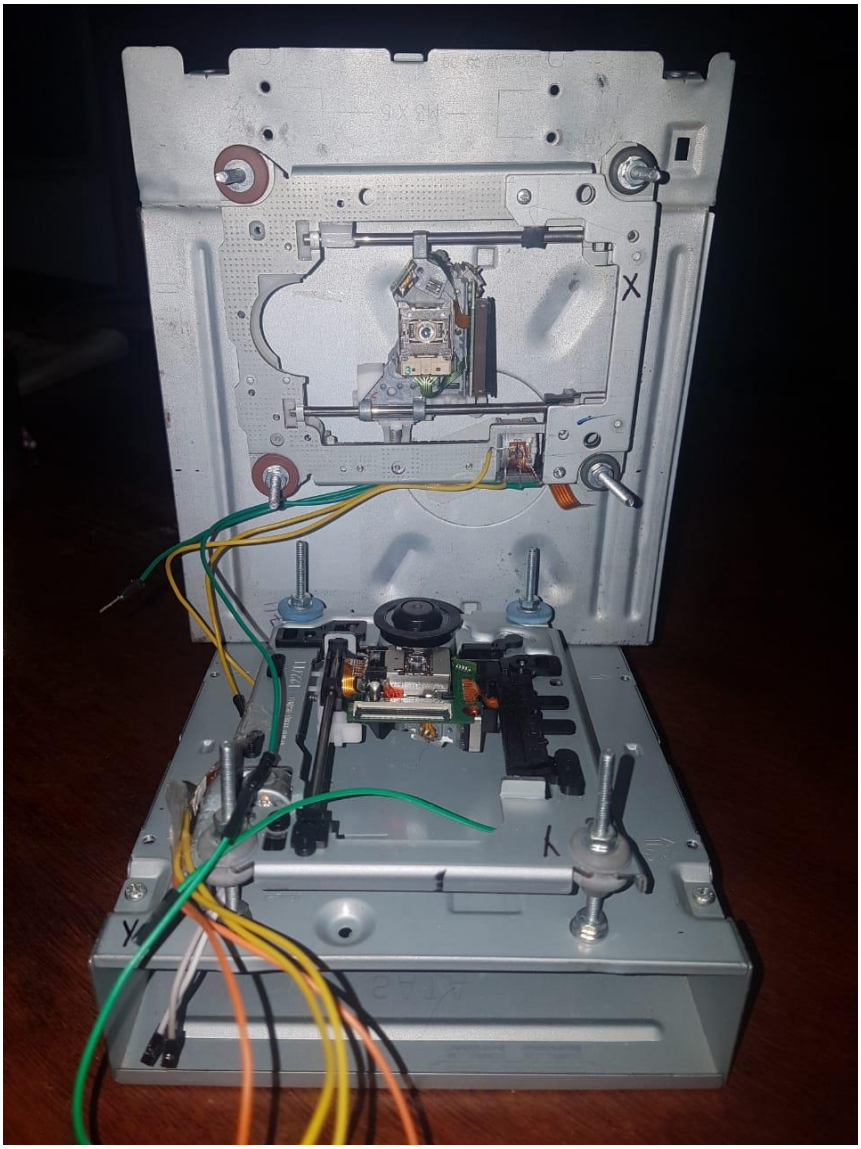
Para a realização dos furos é utilizado uma furadeira com broca de 3mm, para fixar os dois eixos com parafusos de 50mm de comprimento e 2,5mm de diâmetro, adicionando-se arruelas e porcas próprias para os parafusos. A figura 40 apresenta os parafusos, arruelas e porcas usadas no projeto.

Figura 40 – Parafusos, arruelas e porcas.

Fonte: Elaborada pela autora.

Após afixar os parafusos nos furos, é possível acomodar os eixos. É utilizado três porcas para cada orifício, sendo uma para segurar o parafuso na base e deixá-lo firme com a ajuda de uma chave de fenda, outra para servir como referência da altura ao eixo da base e outra para deixar firme a armação. A figura 41 apresenta a estrutura montada do projeto com os eixos X e Y.

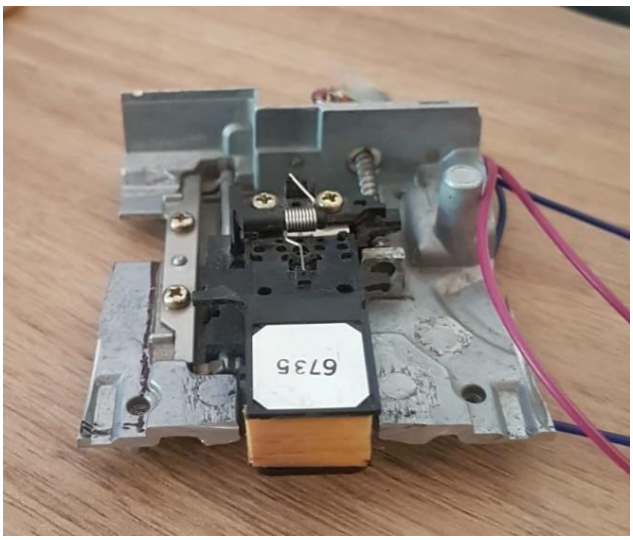
Figura 41 – Estrutura montada.



Fonte: Elaborada pela autora.

É necessário acoplar o eixo Z na estrutura e com o objetivo de adicionar a caneta ao *driver* de disquete, para tal, uma presilha devidamente posicionada é montada para acomodar a caneta. Coloca-se um calço de 10mm x 20mm a fim de permitir que a peça se mantenha reta na ponta do *driver* de disquete. A figura 42 apresenta o *driver* de disquete com o calço.

Figura 42 – *Driver* de disquete com o calço.



Fonte: Elaborada pela autora.

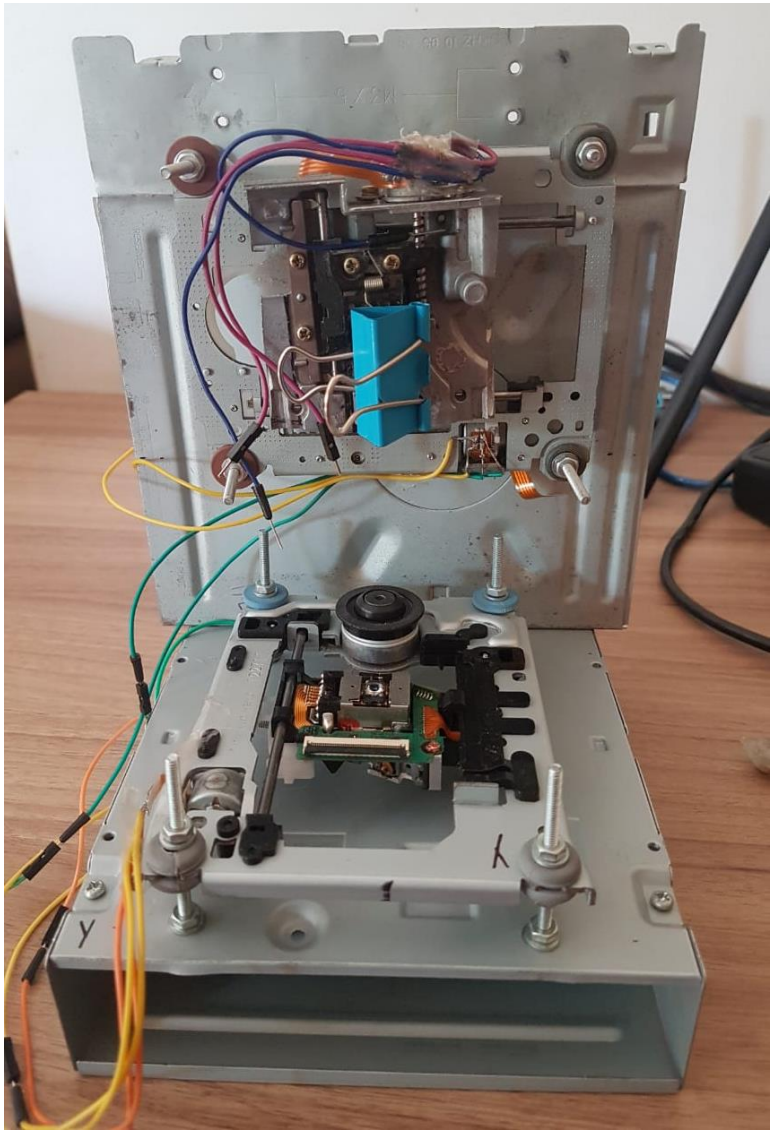
Após a fixação do calço, dois grampos devidamente acoplados, são utilizados para posicionar a caneta de escrita, ou o *end off efect* (atuador de final de movimento). Assim, o *driver* de disquete encontra-se pronto para ser acoplado ao eixo X. A figura 43 apresenta o grampo utilizado no projeto; esse simples grampo possibilita a troca da caneta com facilidade.

Figura 43 – Grampo usado no projeto.

Fonte: Elaborada pela autora.

É necessário afixar o eixo Z em uma posição, que no caso do presente trabalho, após testes, se encontra acima da lente de leitura do *driver* de CD-ROM utilizado como eixo X. A figura 44 apresenta a estrutura do projeto com os três eixos montados.

Figura 44 – Estrutura com os três eixos montados.

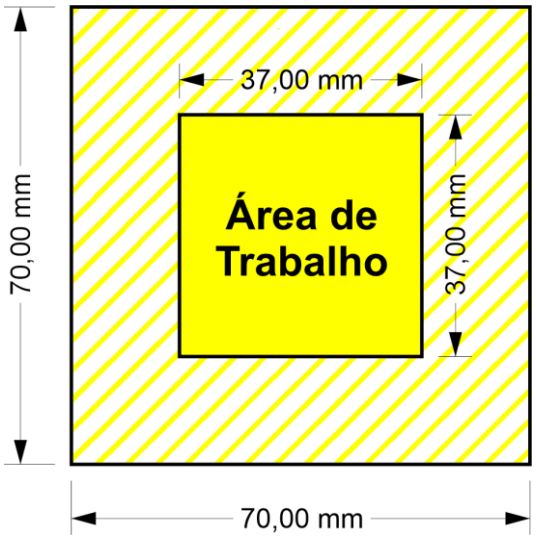


Fonte: Elaborada pela autora.

Após o acoplamento do eixo Z, é necessário acomodar uma caneta em um grampo próprio para prendê-la e fixar a prancha de impressão ao eixo Y. Para isso usa-se uma caneta e para saber o local exato da prancha de impressão, é realizado testes para o cálculo de fim de curso do eixo.

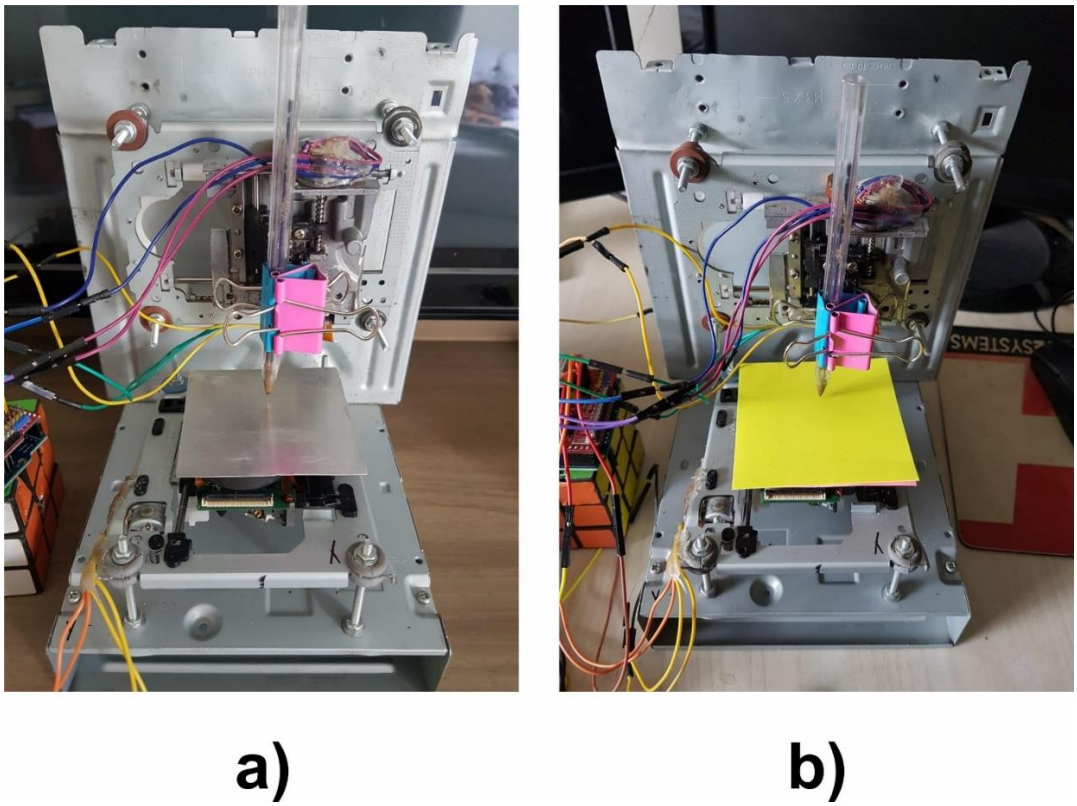
Para o cálculo de fim de curso do eixo, são realizados testes definindo o tamanho da prancha de impressão, usando para tal fim a caneta e os deslocamentos realizados pelos eixos X e Y, e desta forma verificar a área de impressão do projeto. A figura 45 apresenta a área de trabalho do presente projeto. A figura 46 apresenta a montagem com a caneta e a prancha de impressão. A figura 46-a apresenta a estrutura somente com a prancha e a figura 46-b apresenta a estrutura com o papel de impressão.

Figura 45 – Área de trabalho da CNC.



Fonte: Elaborada pela autora.

Figura 46 – Estrutura com a prancha de impressão e papel.

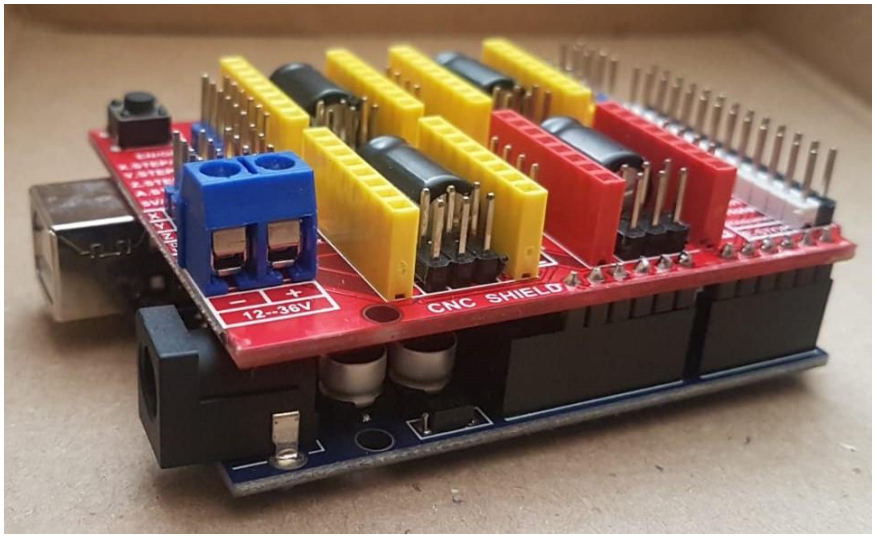


Fonte: Elaborada pela autora.

4.1.3 Microcontrolador

O passo seguinte é preparar a conexão para os microcontroladores. É realizada a expansão com o uso do CNC *Shield*, conectando os pinos do CNC *Shield* no Arduino Uno; os pinos se encaixam perfeitamente, então não ficam dúvidas sobre as posições de ligação. A figura 47 apresenta essa expansão da placa CNC *Shield* juntamente com a placa Arduino.

Figura 47 – Expansão *shield*.



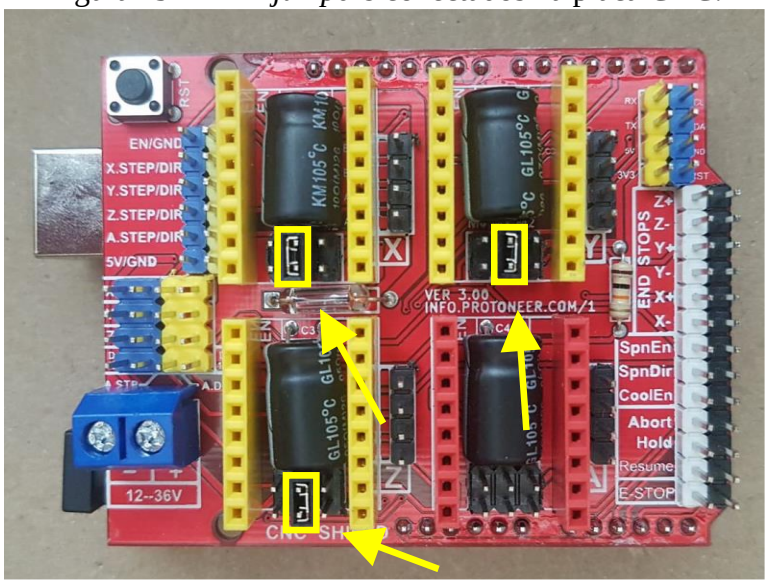
Fonte: Elaborada pela autora.

Posteriormente, são conectados os três minis *jumpers* nas entradas dos *drivers* do *shield*, conectando o segundo pino da primeira linha com o segundo pino da segunda linha, utilizando para isso conectores mini *jumper*. A figura 48 apresenta os minis *jumpers* utilizados nessa configuração e a figura 49 apresenta eles na placa CNC *Shield*.

Figura 48 – Mini *jumpers*.

Fonte: Elaborada pela autora.

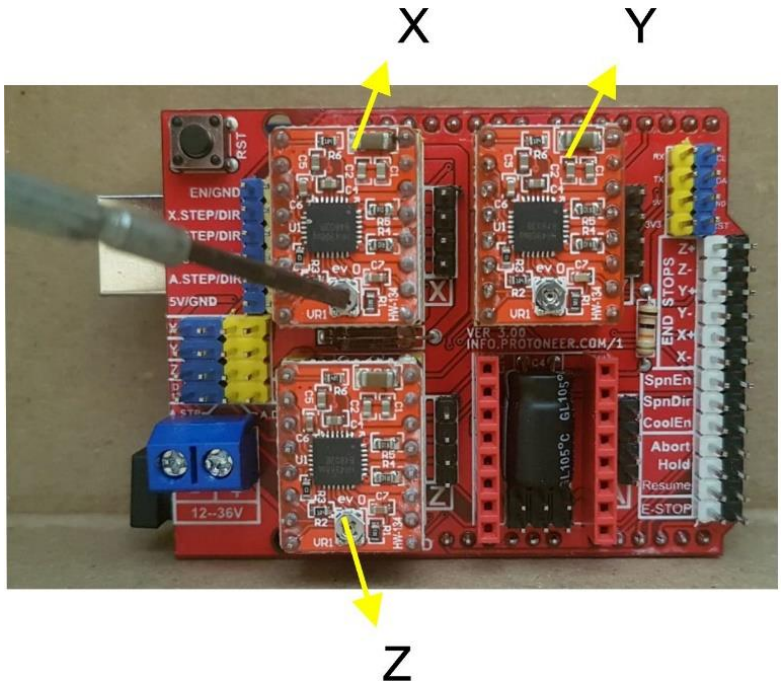
Figura 49 – Mini *jumpers* conectados na placa CNC.



Fonte: Elaborada pela autora.

Na placa CNC *Shield* são conectados os *drivers* do tipo A4988 para as entradas X, Y e Z e na regulagem exata dos *drivers*, é preciso inicialmente girar os *trimpots* de cada um deles para o sentido anti-horário, assim ele se apresenta no valor 0 Volts quando na medição posterior com o multímetro. A figura 50 apresenta a regulagem dos *trimpots* nos *drivers* devidamente acoplados ao CNC *Shield*.

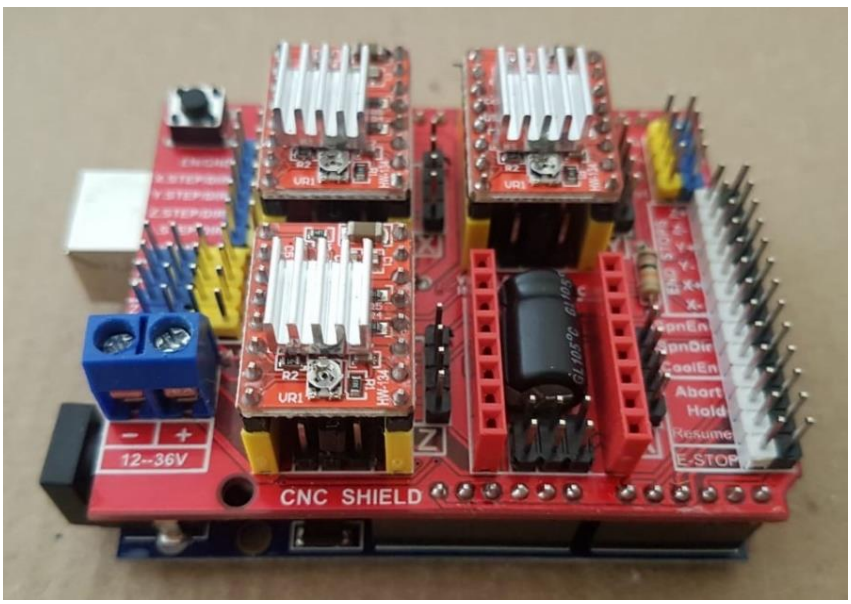
Figura 50 – Placa CNC *Shield* com os três *drivers*.



Fonte: Elaborada pela autora.

Após acoplar os dissipadores de calor nos *drivers* A4988, as placas CNC *Shield* e Arduino Uno encontram-se montados. A figura 51 apresenta a disposição destes vários componentes já montados e interligados.

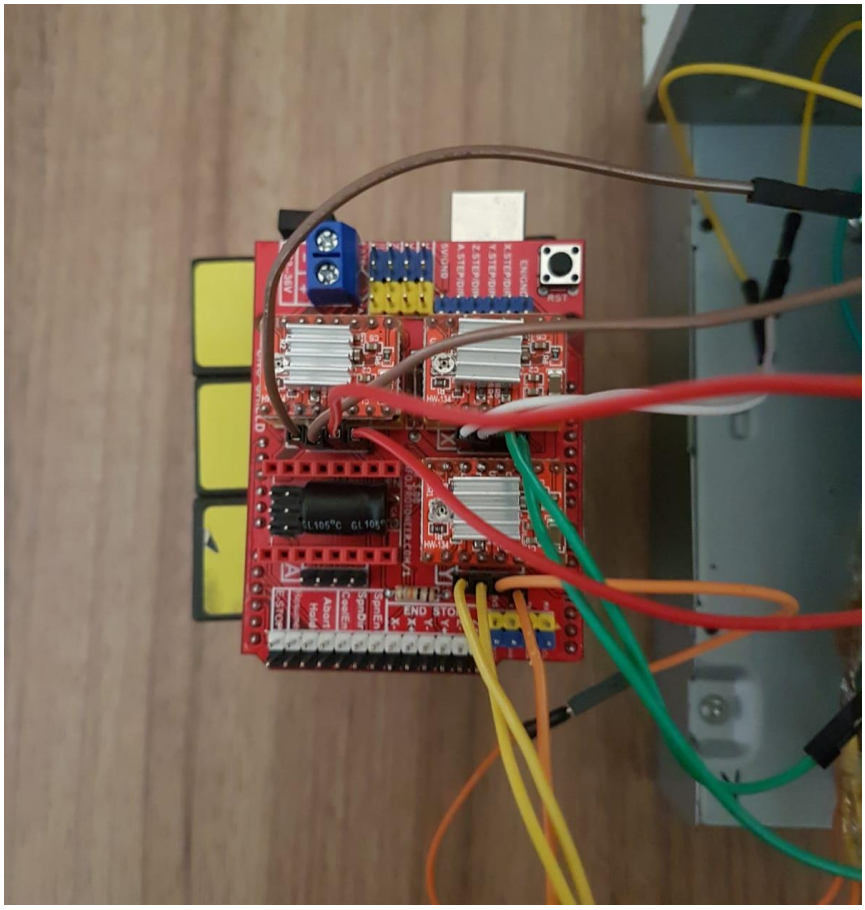
Figura 51 – Arduino Uno com expansão do CNC *Shield*.



Fonte: Elaborada pela autora.

Com a estrutura montada e o *driver* CNC *Shield* no Arduino Uno, é preciso conectar os eixos X, Y e Z devidamente no CNC *Shield*. Cada eixo possui quatro fios e cada fio é ligado nos pinos que se encontram ao lado de cada respectivo *driver*. O eixo X tem seus quatro fios ligados nas pinagens do *driver* X, estendendo-se para os eixos Y e Z. A figura 52 apresenta o *driver* CNC *Shield*, o qual é devidamente conectado aos eixos.

Figura 52 – CNC *Shield* conectada aos eixos.



Fonte: Elaborada pela autora.

4.2 *Software*

A seção 4.2 corresponde aos *softwares* é dividida em três partes: Arduino IDE e UGS, Regulagem da CNC e Desenho *G-code*.

A seção 4.2.1 apresenta os dois *softwares* utilizados para controle da máquina.

A seção 4.2.2 apresenta como é configurado os *drivers* da CNC.

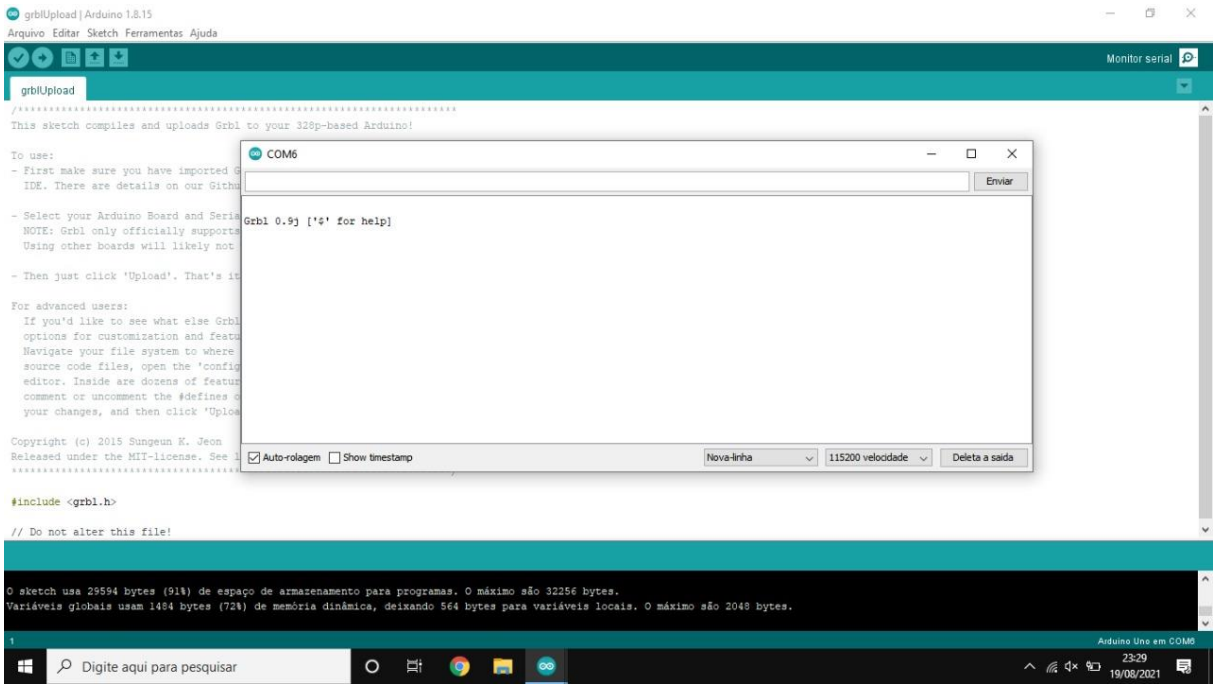
A seção 4.2.3 apresenta qual o método utilizado para transformação do desenho gráfico em código G.

4.2.1 Arduino IDE e UGS

Para que o CNC *Shield* e o Arduino Uno possam trabalhar conjuntamente, é necessário que o Arduino Uno intérprete a linguagem do código G. Para a devida finalidade, é necessário baixar a biblioteca GRBL, copiar todos seus arquivos e *linkar* na *libraries* presentes nos locais de arquivos do Arduino IDE.

Com a extensão GRBL instalada no Arduino IDE, é necessário abrir o exemplo GRBL fornecido pela plataforma, de forma a facilitar e carregar (enviar) no Arduino Uno. Para verificação de envio do GRBL, é possível abrir o monitor serial da plataforma IDE e verificar, com a velocidade 115200 bps, se está imprimindo algum dado; é necessário realizar esse passo somente uma vez. A figura 53 apresenta o carregamento do GRBL no Arduino Uno e mostrado no monitor serial.

Figura 53 – Monitor serial Arduino IDE.

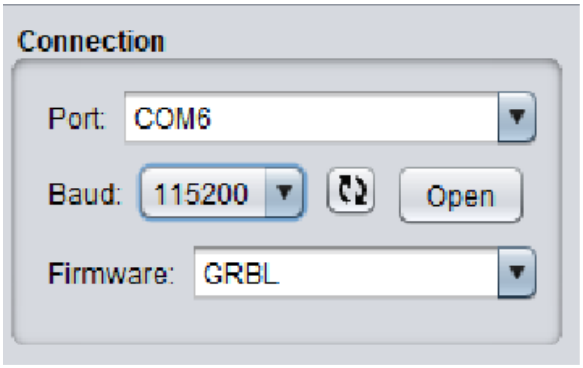


Fonte: Elaborada pela autora.

Com o GRBL carregado no Arduino, estando conectado no computador, é necessário configurar o CNC *Shield* no UGS, onde algumas alterações iniciais são necessárias para configurar o *driver*. Ao abrir o programa, é necessário escolher a porta, taxa de transmissão e *firmware* usados pelo Arduino Uno. Portanto em *port* é escolhido a opção “COM6”, em *baud* a

taxa “115200” e em *firmware* a opção “GRBL”. Com os dados preenchidos, é necessário apertar “*Open”* para abrir o Arduino Uno e possibilitar o uso do UGS. A figura 54 apresenta o local de conexão com o CNC *Shield*.

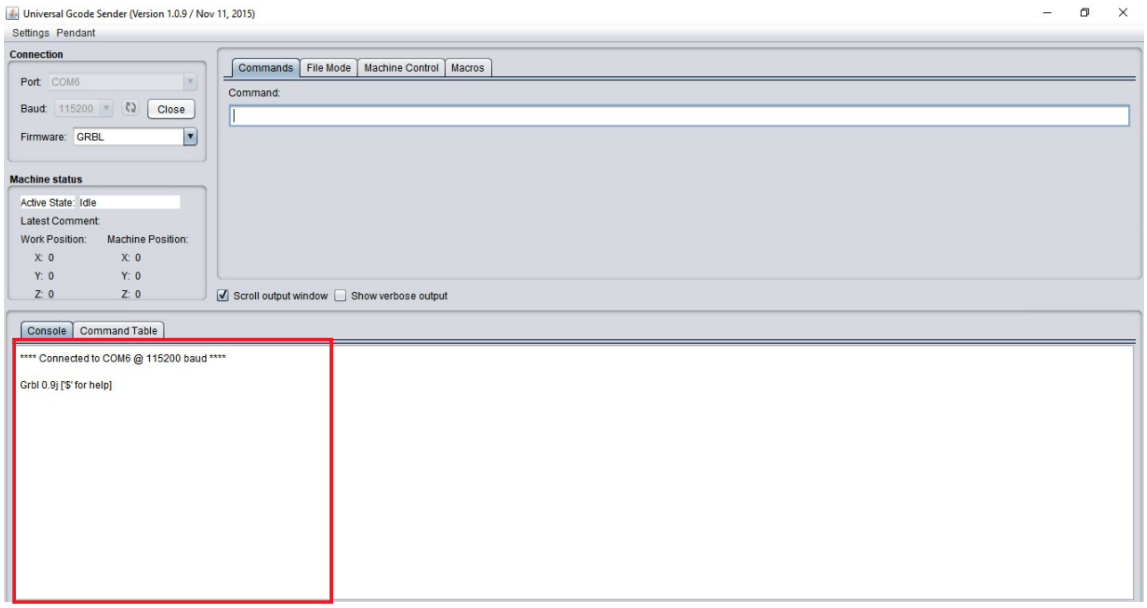
Figura 54 – Conexão UGS



Fonte: Elaborada pela autora.

Ao apertar em abrir, uma mensagem de conexão é informada no *console* do programa juntamente com a mensagem “*Grbl 0.9f ['$' for help]*”. A figura 55 apresenta a mensagem de conexão inicial no UGS.

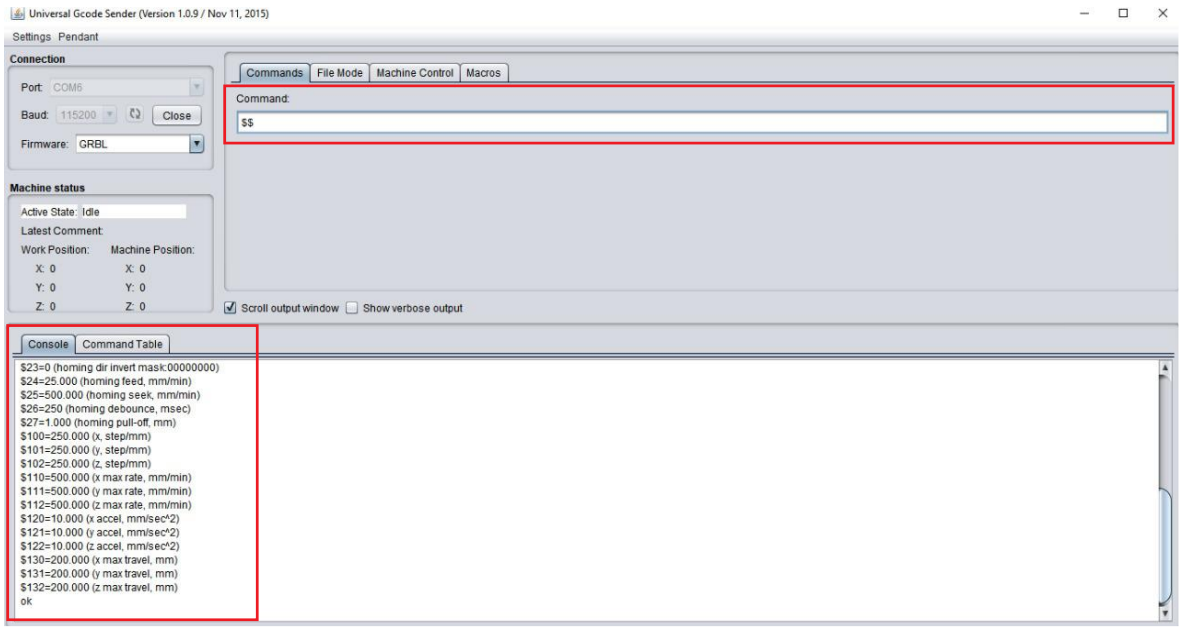
Figura 55 – Conexão inicial no UGS.



Fonte: Elaborada pela autora.

Com o comando ‘$$’ (cifrão), é possível visualizar ou alterar algumas configurações no GRBL. Quando é dado o comando “$$”, o programa mostra uma lista com todos as configurações atuais do CNC *Shield*. A figura 56 apresenta as configurações GRBL.

Figura 56 – Configurações GRBL.



Fonte: Elaborada pela autora.

Para alterar as configurações GRBL, é preciso digitar em “*command*” o comando “$x=valor” e pressionar a tecla “*enter*”. Exemplo: “$110=3000”. A tabela 2 apresenta todas as configurações GRBL carregadas e uma breve descrição de cada configuração.

Tabela 2 – Configurações GRBL.

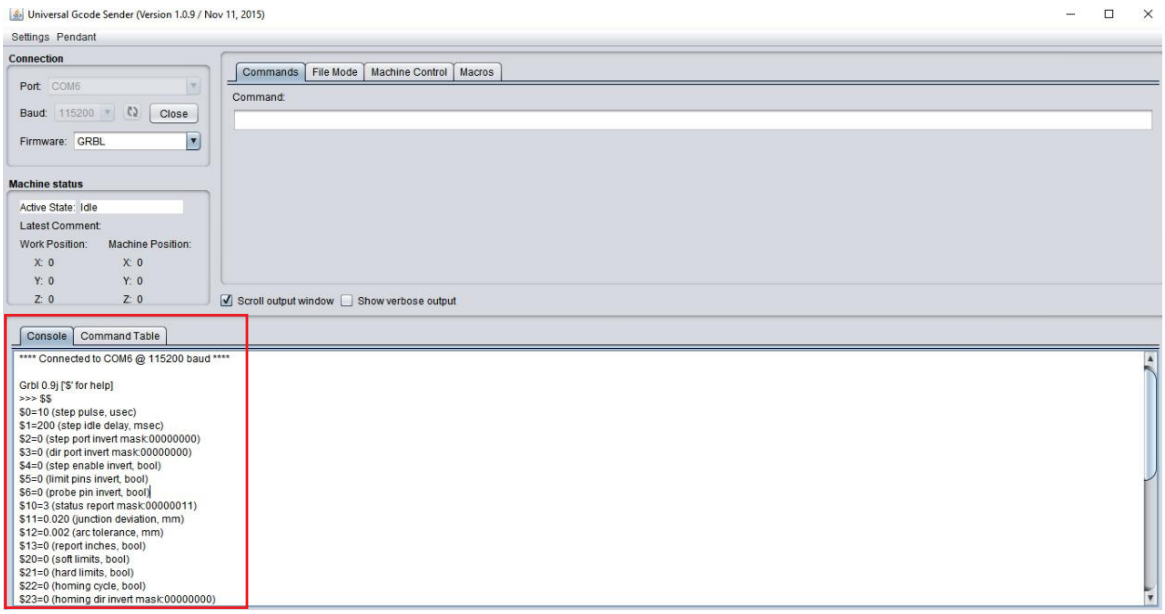
Configurações e valores de amostra	Descrição
$0=10	Pulso de passo, microssegundos
$1=200	Atraso de inatividade da etapa, milissegundos
$2=0	Etapa de inversão da porta, máscara
$3=0	Porta de direção invertida, máscara
$4=0	Etapa habilitar invertido, booleano
$5=0	Pinos de limite invertidos, booleanos
$6=0	Pino da sonda invertido, booleano
$10=3	Relatório de status, máscara
$11=0.020	Desvio de junção, mm
$12=0.002	Tolerância do arco, mm
$13=0	Informe polegadas, booleano
$20=0	Limites suaves, booleano
$21=0	Limites rígidos, booleano

$22=0	Ciclo de *homing*, booleano
$23=0	*Homing dir invert*, máscara
$24=60.000	Alimentação de retorno, mm / min
$25=800.000	Busca de retorno, mm / min
$26=250	*Homing debounce*, milissegundos
$27=20.000	Retirada de *homing*, mm
$100=27.700	X, passos / mm
$101=27.900	Y, passos / mm
$102=43.800	Z, passos / mm
$110=3000.000	Taxa máxima de X, mm / min
$111=3000.000	Taxa máxima de Y, mm / min
$112=1000.000	Taxa máxima de Z, mm / min
$120=1000.000	Aceleração X, mm / seg²
$121=1000.000	Aceleração Y, mm / seg²
$122=200.000	Aceleração Z, mm / seg²
$130=40.000	Viagem máxima X, mm
$131=40.000	Viagem máxima Y, mm
$132=10.000	Viagem máxima Z, mm

Fonte: Elaborada pela autora.

Se a alteração na configuração estiver correta uma mensagem de “Ok” aparece no *prompt*. Quando envia o comando “$$”, o UGS lista as configurações novamente e aquelas alteradas, aparecerão com os novos dados inseridos na lista. A figura 57 apresenta alguns dados da lista de configurações GRBL no UGS.

Figura 57 – Configurações GRBL no UGS.



Fonte: Elaborada pela autora.

4.2.1.1 Configurações GRBL

A tabela 2 contém uma descrição básica das configurações GRBL. Nessa seção, é apresentada uma descrição mais detalhada das configurações GRBL.

4.2.1.1.1 Comando $0 - Pulso de passo, microssegundos ($0=10)

Os motores de passo possuem um tempo mínimo de acionamento que é medido em microssegundos, onde apresenta o valor padrão de 10µs ou a ser avaliado através de testes. É recomendado que esse tempo não seja muito curto, pois o sistema eletromecânico do motor pode não conseguir realizar a tarefa (POLASTRINI, 2016).

4.2.1.1.2 Comando $1 - Atraso de etapa inativa, milissegundos ($1=200)

O atraso de etapa inativa é o tempo em milissegundos que o GRBL atrasa na desativação dos *steppers* toda vez que eles completam um movimento e param ou então, é possível manter eles ativados para manterem a posição (POLASTRINI, 2016).

4.2.1.1.3 Comando $2 - Etapa de inversão de porta, máscara ($2=0)

A etapa de inversão de porta inverte o sinal de pulso de degrau. O sinal padrão começa normalmente em 0V, mas se deseja inverter os eixos X e Z, a configuração enviada para o GRBL é $2 = 5 (POLASTRINI, 2016).

4.2.1.1.4 Comando $3 - Inverter porta de direção, máscara ($3=0)

Essa configuração inverte o sinal de direção dos eixos. No padrão 0, o eixo X se move na direção positiva, quando o sinal de pino de direção é baixado para 0, esse anda na direção negativa, quando o pino de direção é elevado para nível 1, esse anda na direção positiva. O sinal padrão é zero, mas se desejar inverter o eixo y, por exemplo, a configuração enviada para o GRBL é $3 = 2 (POLASTRINI, 2016).

4.2.1.1.5 Comando $4 - Inverter para ativar a etapa, booleano ($4=0)

No padrão do GRBL é usado sinal alto para desabilitar o *stepper* e sinal baixo para habilitar. Se precisar inverter essa configuração do pino de habilitação, é preciso trocar a configuração do GRBL para $4=1 e desativar com $4=0. Pode ser preciso um pulso de energia para a troca (POLASTRINI, 2016).

4.2.1.1.6 Comando $5 - Pino de limite invertido, booleano ($5=0)

Os pinos de limite são por padrão mantidos em nível alto com o resistor *pull-up* interno do Arduino. Quando um pino de limite for baixo, o GRBL interpreta que ele está acionado. Para a troca de comportamento, é preciso carregar a configuração ($5=1) para o GRBL. Pode ser preciso um pulso de energia para a troca (POLASTRINI, 2016).

4.2.1.1.7 Comando $6 - Inversão do pino da sonda, booleano ($6=0)

Semelhante ao funcionamento do comando 5, o pino de sonda é por padrão mantido normalmente em nível alto com o resistor *pull-up* interno do Arduino. Quando o pino da sonda for baixo, o GRBL interpreta que ele está acionado. Para a troca de comportamento, é preciso carregar a configuração ($6=1) para o GRBL. Pode ser preciso um pulso de energia para a troca (POLASTRINI, 2016).

4.2.1.1.8 Comando $10 - Relatório de *status*, máscara ($10=3)

A configuração de relatório de *status* determina quais dados em tempo real do GRBL são reportados ao usuário quando enviado o caracter ‘?’ que representa um comando. O relatório enviado contém os valores de substituição atuais, o estado de execução atual, a taxa de alimentação em tempo real, a posição em tempo real, os estados do buffer, os estados dos pinos e o número da linha do *G-code* atualmente em execução. Para enviar o relatório somente com a posição da máquina, é preciso trocar a configuração para $10=1 (POLASTRINI, 2016).

4.2.1.1.9 Comando $11 - Desvio de junção, mm ($11=0.020)

Desvio de junção determina o quão rápido a máquina pode se mover em um dado caminho *G-code*. Se for uma curva, por exemplo, e a máquina estiver se movendo rápido, essa configuração determina o tanto que a máquina precisa desacelerar para fazer a curva. Quanto

mais alto o valor, mais rápido será o movimento pelos cantos e aumenta o risco de perder passos e posicionamento. Quanto menor o valor, mais lento a máquina faz o percurso, levando a curvas mais cuidadosas (POLASTRINI, 2016).

4.2.1.1.10 Comando $12 - Tolerância de arco, mm ($12=0.002)

O GRBL renderiza arcos, círculos e hélices subdividindo-os em pequenas linhas, de modo que a precisão do tracejado nunca fique abaixo do valor configurado. Valores menores possuem uma maior precisão, porém, podem sobrecarregar o GRBL, ocasionando um desenho com baixa definição por possuir muitas linhas minúsculas. Em contrapartida, valores maiores possuem menor precisão, mas podem acelerar o desempenho do GRBL, já que eles possuem menos linhas de código (POLASTRINI, 2016).

4.2.1.1.11 Comando $13 - Polegadas do relatório, booleano ($13=0)

O GRBL também possui um relatório de posicionamentos em tempo real informando a localização da máquina naquele exato momento e parâmetros para desvios de coordenadas. Por padrão, essa configuração é relatada em milímetros, mas é possível alterar para relatar em polegadas. Para receber o relatório em polegadas, é preciso alterar a configuração com o comando $13=1, se preferir voltar para mm, é preciso retornar para o valor $13=0 (POLASTRINI, 2016).

4.2.1.1.12 Comando $20 - Limites suaves, booleano ($20=0)

A configuração de limites suaves é um recurso de segurança que ajuda a máquina a evitar que se desloque fora das margens limites. Se um código GRBL exceder o espaço limite da máquina, o GRBL enviará um sinal de suspensão de alimentação da máquina, desliga a ferramenta e emite um alarme do sistema para indicar o problema. Para ativar a função, é preciso enviar o comando $20=1, para desativar é preciso enviar $20=0 (POLASTRINI, 2016).

4.2.1.1.13 Comando $21 - Limites rígidos, booleano ($21=0)

Na configuração de limites rígidos, interruptores são alocados perto do final do curso de cada eixo onde há problemas se o eixo se mover muito para este lado. Quando o interruptor for acionado, a máquina interrompe seu movimento, desligando os fusos, entrar em modo alarme

e força a máquina a reiniciar. Para ativar a função, é preciso enviar o comando $21=1, para desativar é preciso enviar $21=0 (POLASTRINI, 2016).

4.2.1.1.14 Comando $22 - ciclo de *homing*, booleano ($22=0)

O ciclo de *homing* é usado para localizar a posição da máquina toda vez que inicia o GRBL. Essa configuração é importante também para quando há uma queda de energia e a máquina desliga. Com o ciclo de *homing* é possível obter sua localização exata. Para usar o ciclo de *homing*, é preciso ter interruptores de limite em posição fixa. Quando o *homing* está habilitado, o GRBL bloqueia todos os comandos *G-code*, para desbloquear, é preciso apertar o botão $X (POLASTRINI, 2016).

4.2.1.1.15 Comando $23 – Máscara de direção de retorno ($23=0)

O GRBL assume por padrão que os interruptores de limite de *homing* estão na direção positiva, movendo assim, os eixos X, Y e Z na direção positiva. Se a máquina possuir uma chave de fim de curso na direção negativa, a configuração da máscara de direção de retorno pode fazer com que os eixos sejam invertidos (POLASTRINI, 2016).

4.2.1.1.16 Comando $24 – Taxa de alimentação de *homing*, mm / min ($24=60.000)

Primeiramente o ciclo de *homing* procura os interruptores de limite em uma taxa de alimentação alta e depois de encontrá-los, a configuração de taxa de alimentação de *homing* se move em uma taxa mais lenta para o local preciso do zero da máquina (POLASTRINI, 2016).

4.2.1.1.17 Comando $25 - Busca de *homing*, mm/min ($25=800.000)

A configuração $25 é a taxa de busca de *homing,* na qual ela leva para tentar encontrar as chaves interruptores de limite (POLASTRINI, 2016).

4.2.1.1.18 Comando $26 - *Homing debounce*, milissegundos ($26=250)

Toda vez que um interruptor é acionado, alguns deles podem gerar ruído mecânico ou elétrico, que rebate os sinais altos e baixos por alguns milissegundos antes de se estabilizar (POLASTRINI, 2016).

4.2.1.1.19 Comando $27 - *Homing pull-off*, mm ($27=20.000)

Essa configuração ajuda a evitar o disparo acidental do limite rígido após o ciclo de *homing* (POLASTRINI, 2016).

4.2.1.1.20 Comando $100, $101 e $102 - [X, Y, Z] passos / mm ($100=27.700; $101=27.900; $102=43.800)

O GRBL precisa saber até onde cada eixo pode se movimentar. Essas configurações definem quantos passos por milímetros, os eixos X, Y e Z se movimentam até alcançar o limite (POLASTRINI, 2016).

4.2.1.1.21 Comando $110, $111 e $112 - [X, Y, Z] Taxa máxima, mm / min ($110=3000.000; $111=3000.000; $112=1000.000)

Essas configurações definem qual a taxa máxima que cada eixo pode se mover. Se alguns dos movimentos GRBL exceder a taxa máxima configurada, a máquina vai desacelerar o movimento para garantir que nenhum dos eixos exceda seus limites de taxa máxima (POLASTRINI, 2016).

4.2.1.1.22 Comando $120, $121, $122 - [X, Y, Z] Aceleração, mm / s^2 ($120=1000.000; $121=1000.000; $122=200.000)

Essas configurações definem os parâmetros de aceleração dos eixos em mm/s^2. Um valor baixo torna o GRBL mais lento, um valor alto, torna ele mais rápido (POLASTRINI, 2016).

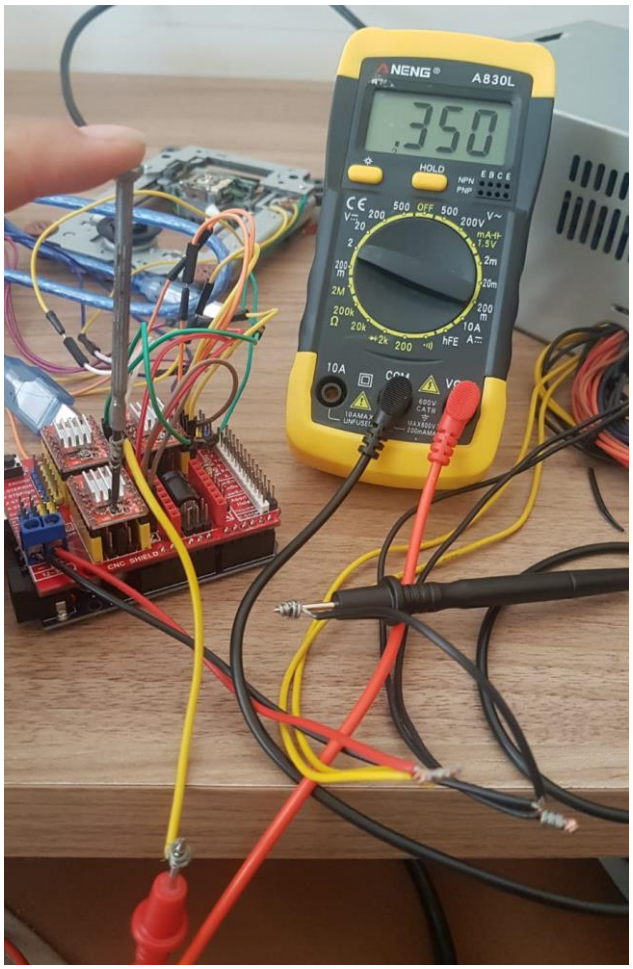
4.2.1.1.23 Comando $130, $131, $132 - [X, Y, Z] Viagem máxima, mm ($130=40.000; $131=40.000; $132=10.000)

Essas configurações definem o curso máximo de limites dos eixos em milímetros. Só funciona se tiver limites suaves e *homing* habilitados, assim o GRBL verifica se excedeu os limites da máquina com um comando de movimento (POLASTRINI, 2016).

4.2.2 Regulagem da CNC

Com o projeto ligado na fonte, é preciso obter a regulagem dos *drivers* A4988. Anteriormente girou-se o *trimpot* do *driver* para 0V; agora é necessário girar para o sentido horário a fim de alcançar o valor de 0,35V em corrente contínua, permitindo que ele energize com a corrente e tensão adequadas. Para essa medição é utilizado um multímetro na faixa tensão contínua. Na saída “COM” do multímetro é posto o negativo da fonte e na saída “VΩmA” é posto no *trimpot* dos *drivers.* A figura 58 apresenta a medição dos *trimpots* com o valor no multímetro.

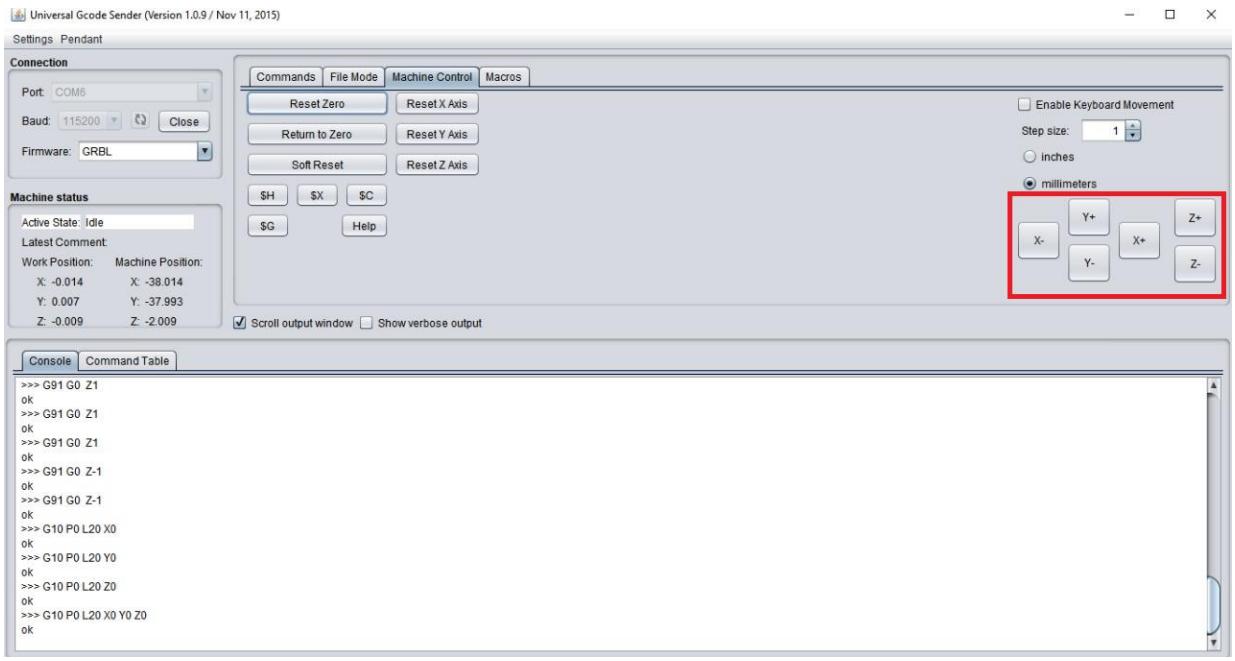
Figura 58 – Medição *trimpots*.



Fonte: Elaborada pela autora.

Depois do projeto energizado, montado os eixos e conectado ao computador, é necessário verificar se as movimentações dos eixos estão caminhando para o sentido correto. Para essa verificação e controle do projeto, é usado o UGS, onde na aba "*Machine Control*" encontra-se 6 botões: X-, X+, Y-, Y+, Z- e Z+. A figura 59 apresenta no UGS a parte de controle da máquina.

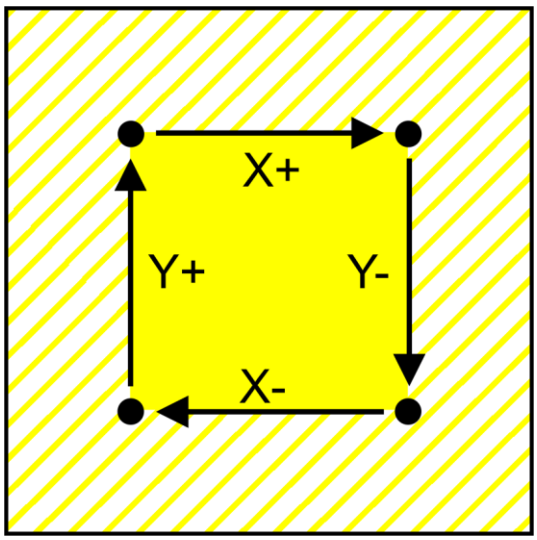
Figura 59 – Controle da máquina no UGS.



Fonte: Elaborada pela autora.

Na verificação dos eixos, ao apertar em "X-" o eixo X movimenta para esquerda e ao apertar em "X+" movimenta para direita. Para o eixo Y, ao apertar em "Y-" o eixo movimenta para trás ou para baixo se for visto por uma vista superior, ao apertar em "Y+" o eixo movimenta para frente ou para cima em uma vista superior. No eixo Z, ao apertar em "Z-" o eixo desce e ao apertar em "Z+" o eixo sobe A figura 60 exemplifica o movimento da caneta pelos eixos X e Y.

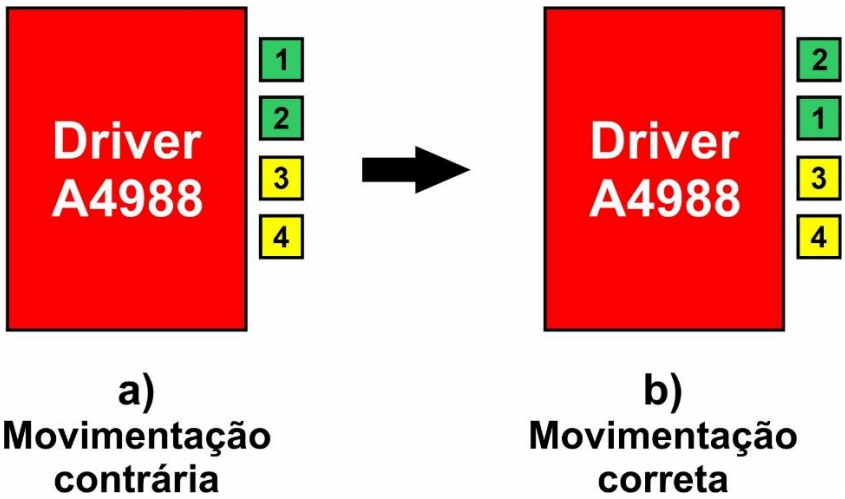
Figura 60 – Movimento eixos X e Y.



Fonte: Elaborada pela autora.

Caso haja algum eixo movimentando para o lado contrário, é possível corrigir trocando de lado os fios de mesma cor, mesma dupla, do *driver* correspondente. A figura 61 apresenta um exemplo de troca de fios em um dos três *drivers* onde cada número é representado por um fio e cada dupla de cor é representado pela dupla do motor. Na figura 61-a, o *driver* está com as conexões trocadas, portanto o eixo está movimentando para o sentido contrário. Na figura 61-b, o *driver* está com a conexão correta e se movimenta para o sentido correto. Percebe-se que na figura 61-a, os fios estavam na ordem “1, 2, 3 e 4” e na figura 61-b, eles passaram para a ordem “2, 1, 3 e 4”.

Figura 61 – Exemplo troca de fios na CNC *Shield*.



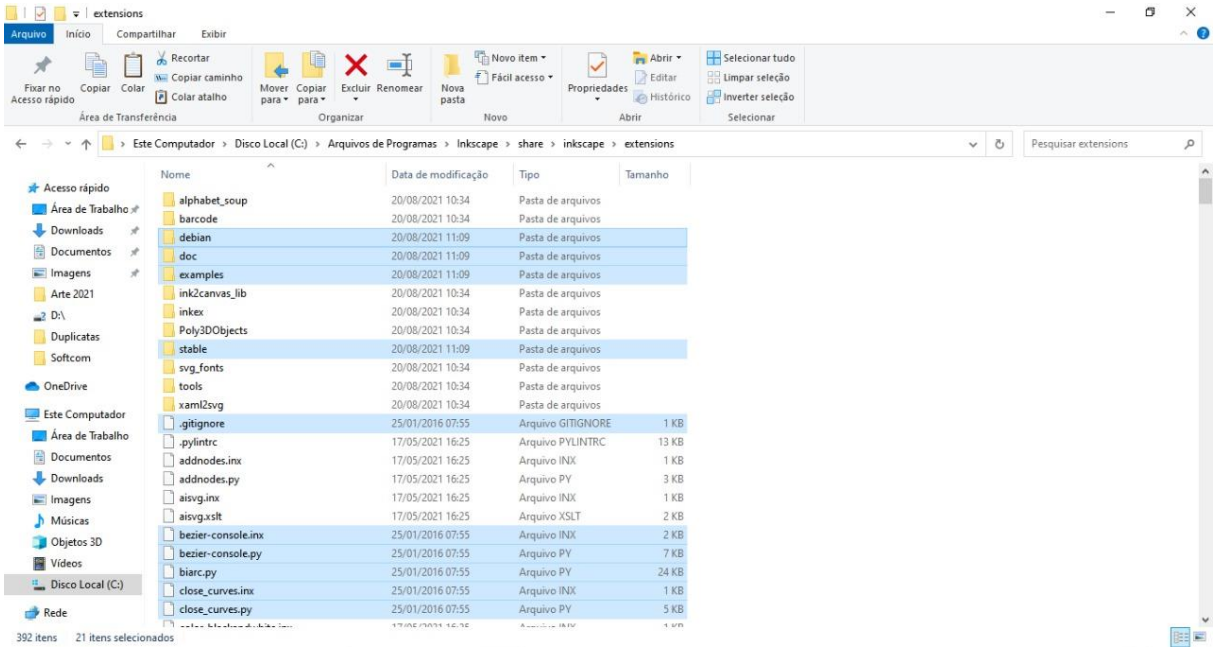
Fonte: Elaborada pela autora.

4.2.3 Desenho G-code

Com o projeto montado e configurado, é necessário carregar o desenho na máquina, porém o desenho tem de estar no formato “.ngc”. Assim. ele estará em código G e nesse propósito o *software Inkscape* é uma boa alternativa para transformar desenhos em código G.

Para usar o *Inkscape,* com o propósito do trabalho, é necessário instalar a extensão *gcodetools*, copiar todos os arquivos da pasta e *linkar* na pasta “*share*” presente no local de arquivo do *Inkscape*. A figura 62 apresenta o local de arquivo do *Inkscape* e os arquivos selecionados são os copiados do *gcodetools*.

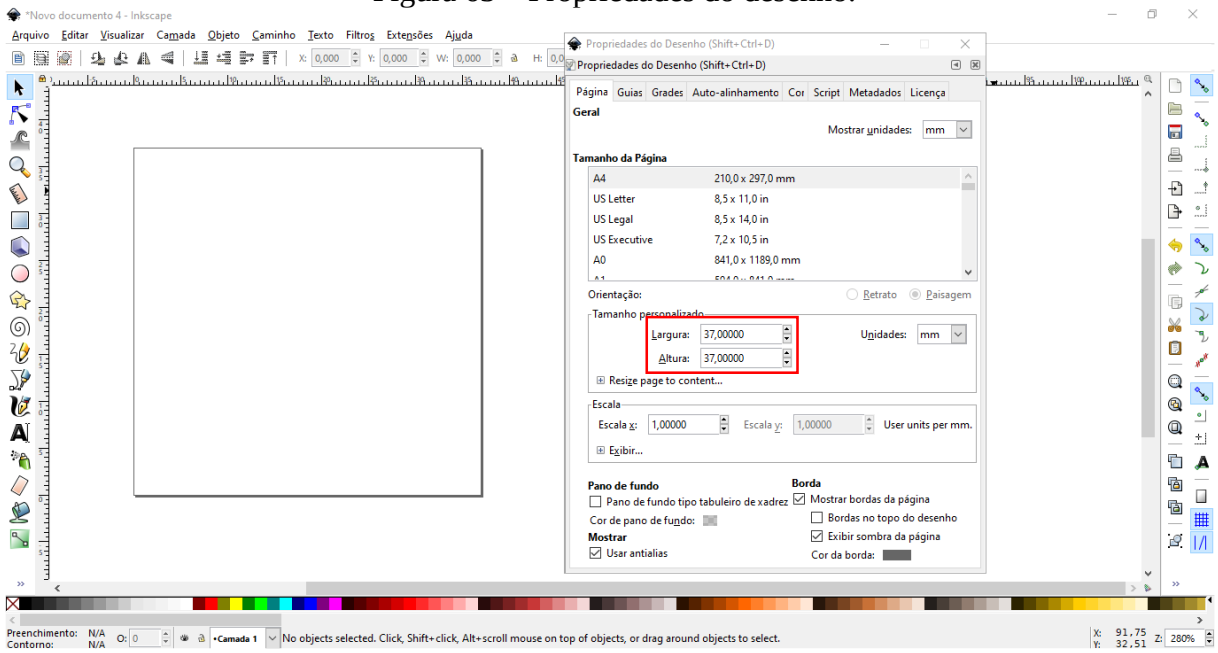
Figura 62 – Local de arquivos *Inkscape*.



Fonte: Elaborada pela autora.

O projeto possui uma área de trabalho de 37mm x 37mm e para indicar o tamanho permitido do desenho, é possível alterar o layout do *Inkscape* para 37mm x 37mm, isso torna mais fácil o trabalho na hora de dimensionar o desenho. A figura 63 apresenta as propriedades do desenho.

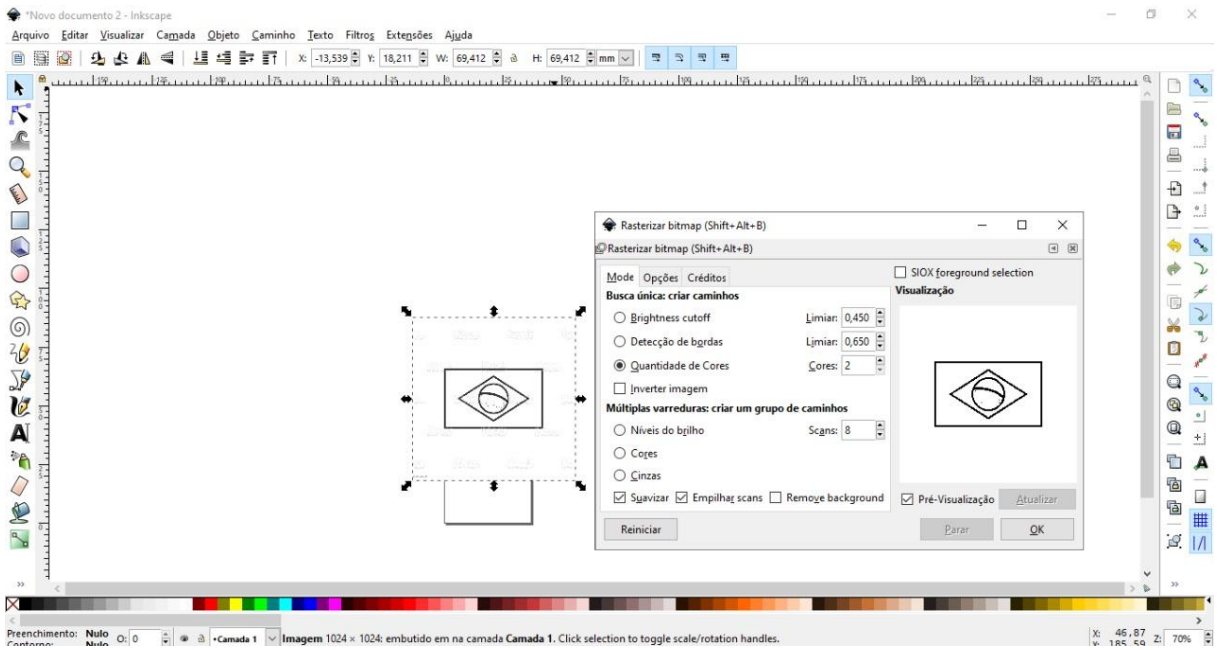
Figura 63 – Propriedades do desenho.



Fonte: Elaborada pela autora.

É possível usar também uma imagem em “JPG”, “JPEG”, “PNG” ou outro formato que é possível baixar na internet, mas antes é necessário vetorizar a figura. O *Inkscape* permite rasterizar uma imagem, em que essa opção se encontra em “Objeto > Rasterizar bitmap”. A figura 64 apresenta a opção Rasterizar bitmap.

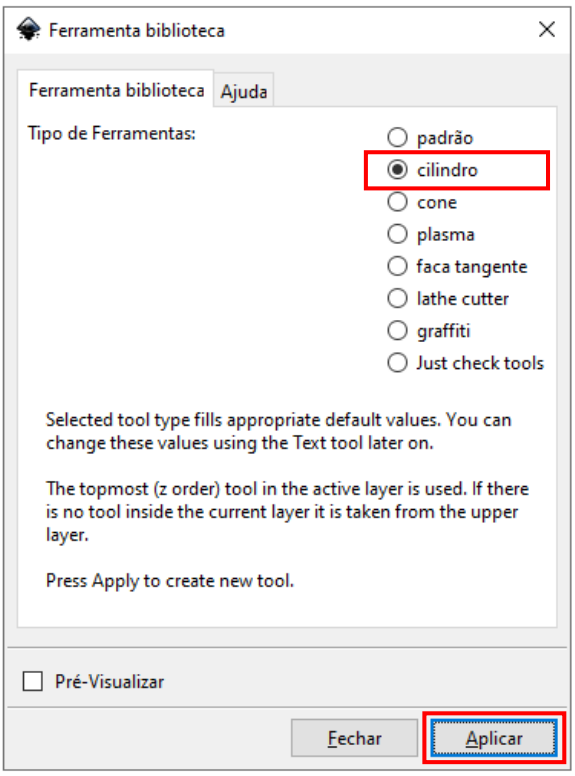
Figura 64 – Rasterizar bitmap.



Fonte: Elaborada pela autora.

Com o desenho vetorizado, é necessário transformá-lo em código G, e o primeiro passo é escolher o tipo de ferramenta a ser usado na máquina. No presente trabalho é usado a ferramenta do tipo cilindro, pois a caneta se assemelha a um cilindro. Para denotar como cilindro, com o desenho selecionado, é necessário acessar o caminho "Extensões > *Gcodetools* > Ferramenta biblioteca", selecionar "cilindro" e depois apertar em "Aplicar". A figura 65 apresenta a ferramenta biblioteca.

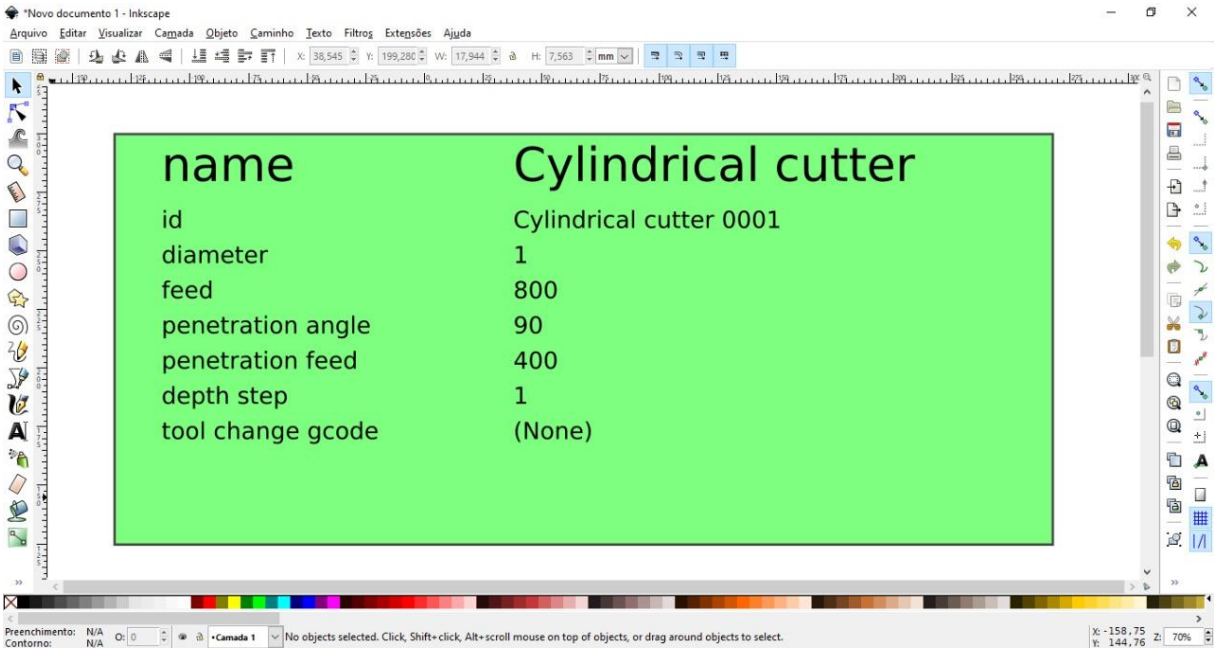
Figura 65 – Ferramenta biblioteca.



Fonte: Elaborada pela autora.

Ao aplicar a ferramenta biblioteca, um retângulo é inserido com as propriedades do desenho, são elas: "*name*", "*id*", "*diameter*", "*feed*", "*penetration angle*", "*penetration feed*", "*depth step*" e "*tool change gcode*". *Name* refere-se ao nome do arquivo, *id* refere-se a identificação do arquivo, *diameter* refere-se ao diâmetro da ponta da caneta, *feed* é a velocidade de impressão, *penetration angle* refere-se ao ângulo de penetração, *penetration feed* refere-se a velocidade de penetração e é a velocidade com que o eixo Z vai subir ou descer, *depth step* significa passos por camada e em "1" permite que a caneta afunde um pouco no papel. *Tool change gcode* refere-se a mudança de ferramenta *G-code* que no caso é o padrão *none*. Todas essas propriedades podem ser alteradas. A figura 66 apresenta as propriedades dos desenhos.

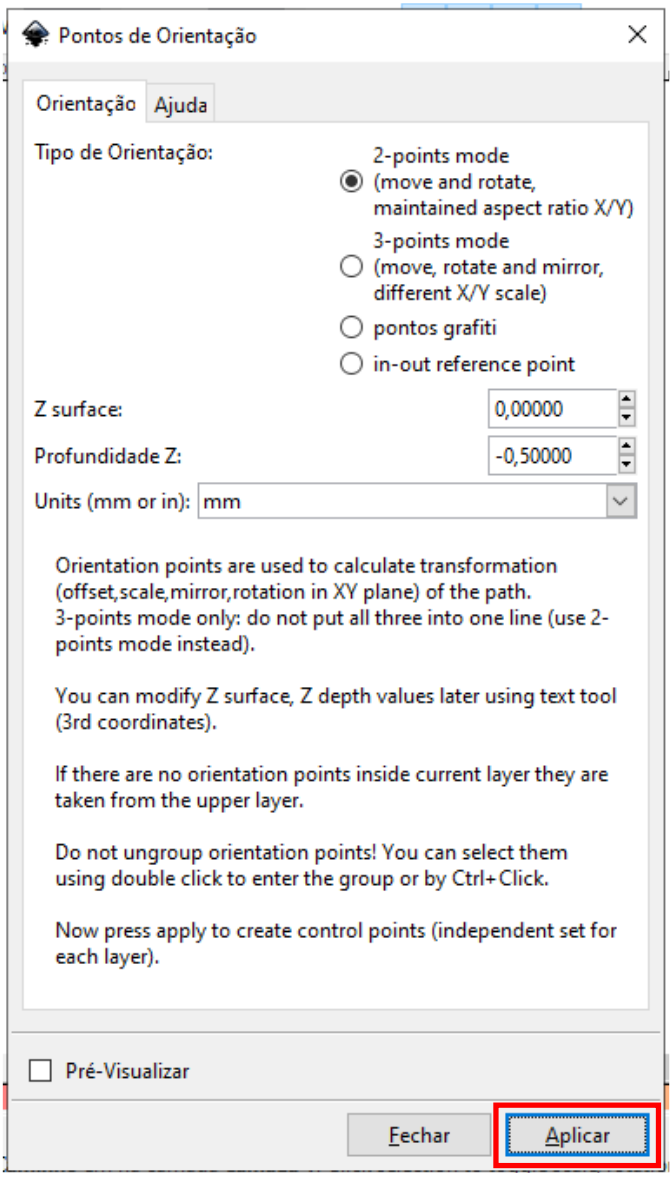
Figura 66 – Propriedades do desenho.



Fonte: Elaborada pela autora.

O segundo passo para transformar o desenho em código G é aplicar os pontos de orientação. Para inserir os pontos, é necessário acessar o caminho “Extensões > *Gcodetools* > Pontos de Orientação” e apertar em “Aplicar”. A figura 67 apresenta as propriedades dos pontos de orientação.

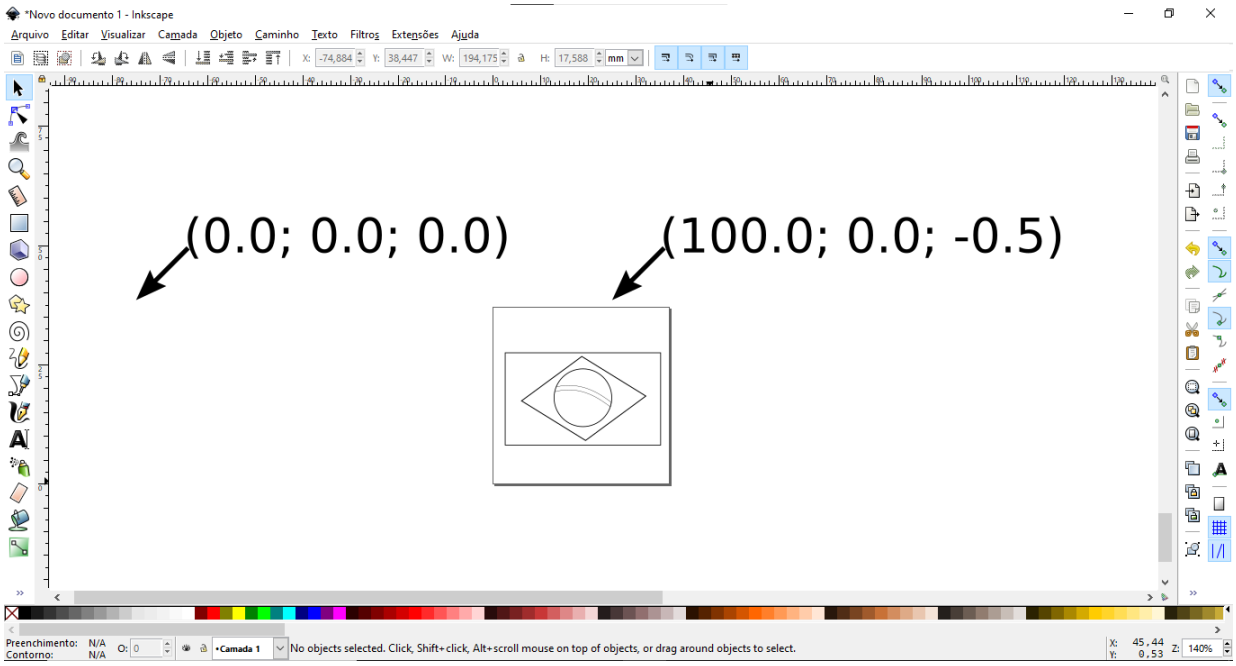
Figura 67 – Propriedades pontos de orientação.



Fonte: Elaborada pela autora.

Ao aplicar os pontos de orientação, dois campos de coordenadas são apresentados na tela, um primeiro com todos em zero, iguais a “(0.0; 0.0; 0.0)” e outro com diferentes números, iguais a “(100.0; 0.0; -0.5)”. A figura 68 apresenta os pontos de orientação do desenho.

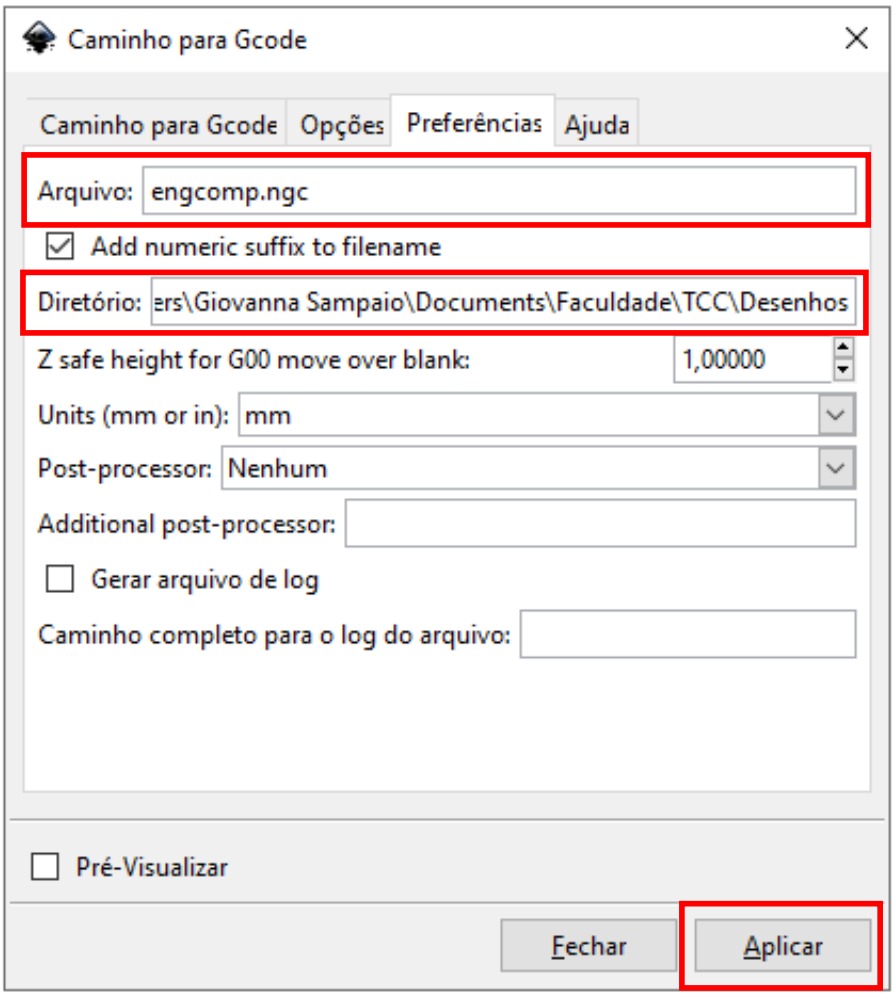
Figura 68 – Pontos de orientação.



Fonte: Elaborada pela autora.

O terceiro passo é salvar o arquivo em código G. Para salvar o desenho em *G-code*, é necessário acessar o caminho "Extensões > *Gcodetools* > Caminho para *Gcode*". Na nova janela, em "Preferência", é colocado em "Diretório:" o caminho escolhido para salvar o arquivo e se preferir, é possível alterar o nome do arquivo presente em "Arquivo:" e depois apertar em "Aplicar". Desta forma, o desenho é salvo no caminho escolhido com o nome informado anteriormente e agora no formato ".ngc". O desenho se encontra em código G e agora é possível carregá-lo na máquina. A figura 69 apresenta as propriedades do Caminho para *Gcode*.

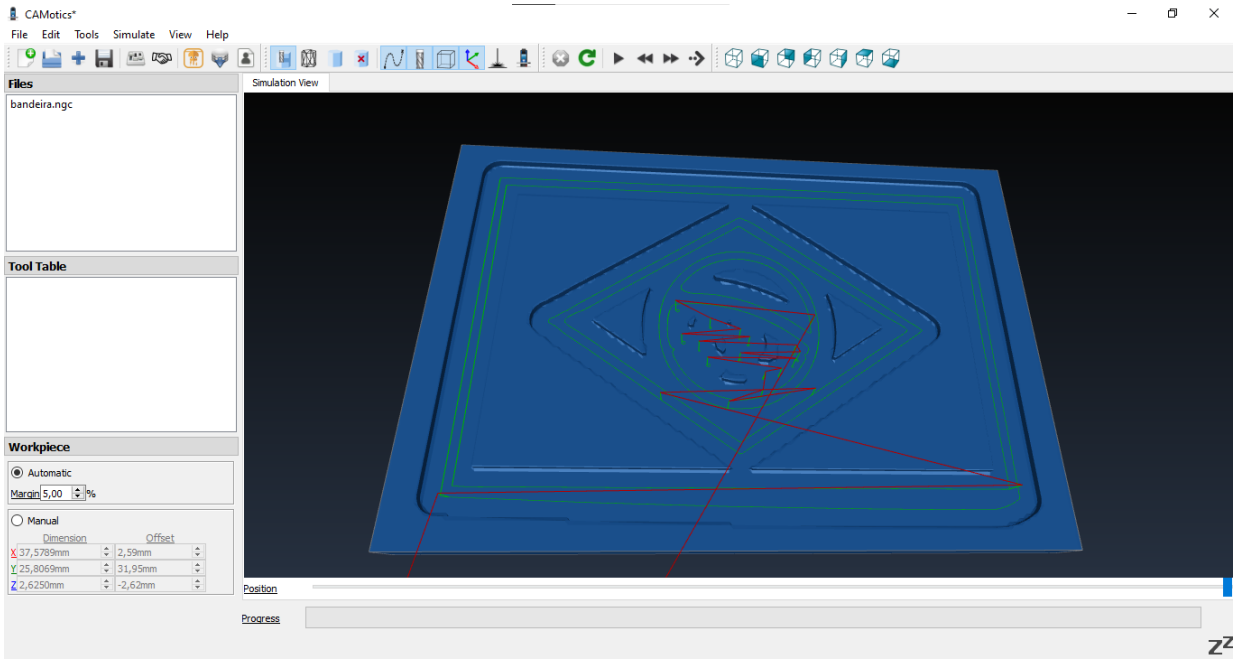
Figura 69 – Caminho para *Gcode*.



Fonte: Elaborada pela autora.

O desenho está salvo no local de arquivo de escolha e agora se encontra pronto para ser carregado na CNC, mas para que não haja desperdício de materiais, visualização final do desenho ou outro propósito, é possível usar o CAMotics para simular o desenho na CNC, se o desenho não carrega no programa, ele não funciona na CNC. A figura 70 apresenta a tela do CAMotics com o desenho carregado.

Figura 70 – Visualização do desenho no CAMotics.

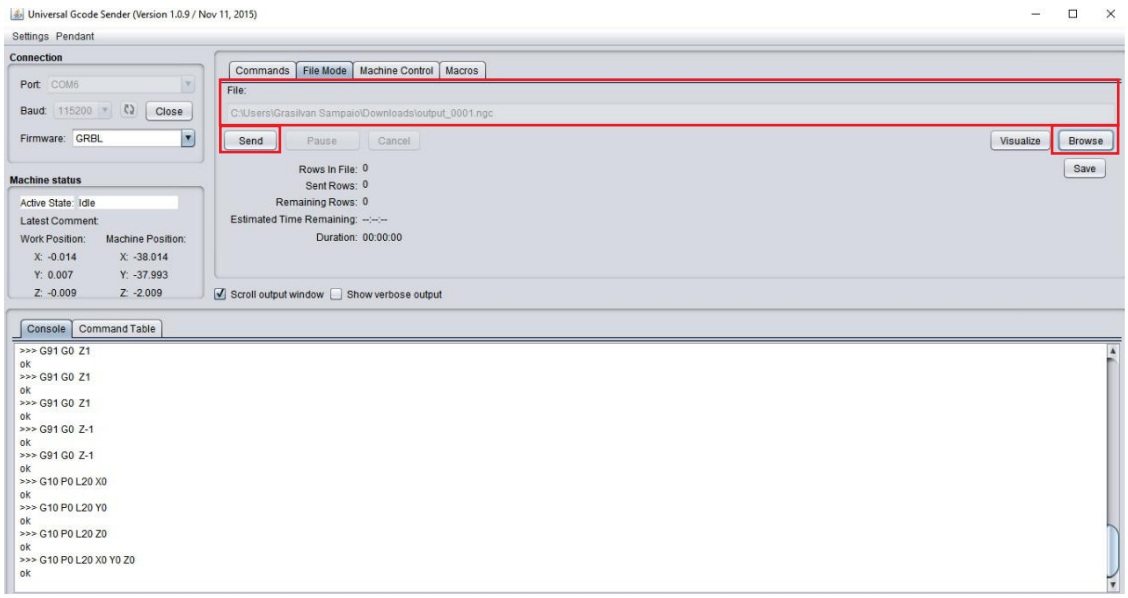


Fonte: Elaborada pela autora.

O projeto se encontra na reta final, faltando apenas enviar o desenho à CNC. Para tal fim, é necessário primeiro ordenar a caneta para o canto superior esquerdo e apertar em *Reset X Axis*, *Reset Y Axis* e *Reset Z Axis* localizado na aba "*Machine Control*" do UGS. Esse é o ponto inicial de todo desenho; seguindo esse procedimento não há perigo de a máquina tentar desenhar fora da área de trabalho permitida.

Para carregar o desenho no UGS, é necessário abrir a aba "*File Mode*" e carregar o arquivo em "*File*". Para isso, ao apertar em "*Browser*" o explorador de arquivos é aberto e é possível navegar pelo computador e selecionar o arquivo que deseja que a CNC desenhe. Com o arquivo carregado, é necessário apertar em "*Send*" para enviar o arquivo à máquina. Assim, a máquina ferramenta inicia a realização dos movimentos necessários para a plotagem do desenho. A figura 71 apresenta a tela de carregamento do desenho.

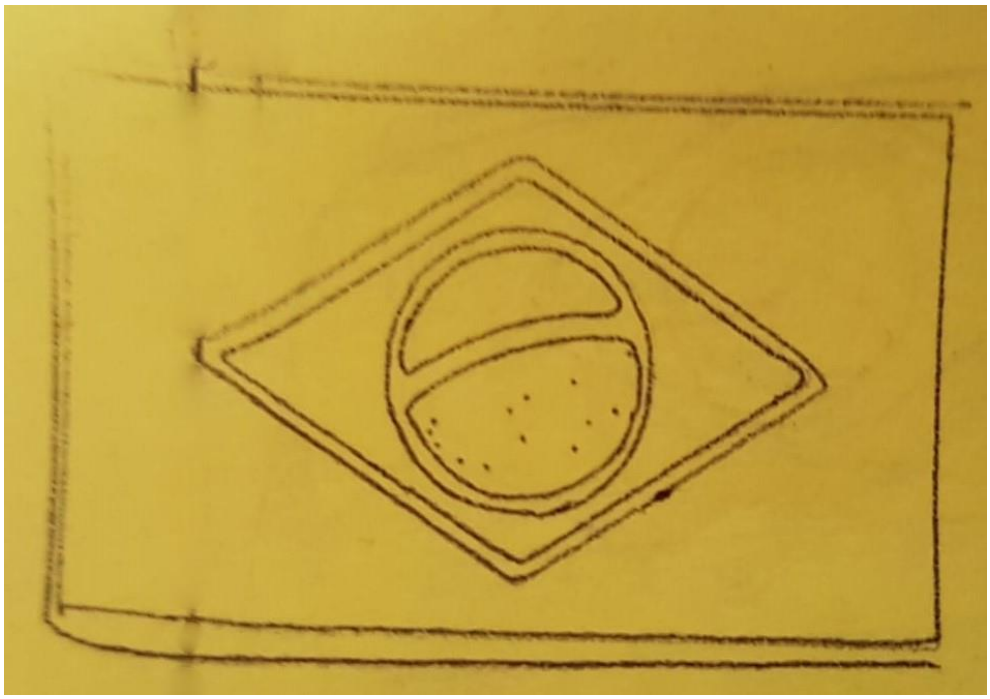
Figura 71 – Tela de carregamento do desenho.



Fonte: Elaborada pela autora.

Ao enviar o desenho, a máquina começa a desenhar conforme instruções oferecidas através do código G, linguagem em que o desenho foi construído, e ao terminar, uma mensagem da duração do desenho é apresentada ao final. A figura 72 apresenta um dos desenhos realizados pela CNC.

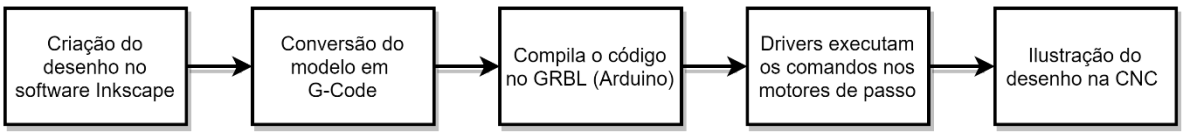
Figura 72 – Desenho final CNC.



Fonte: Elaborada pela autora.

São necessárias algumas etapas para a criação do desenho. De forma a simplificar as etapas utilizadas na CNC, a figura 73 apresenta em um esquema de blocos, basicamente a sequência de etapas para execução na máquina.

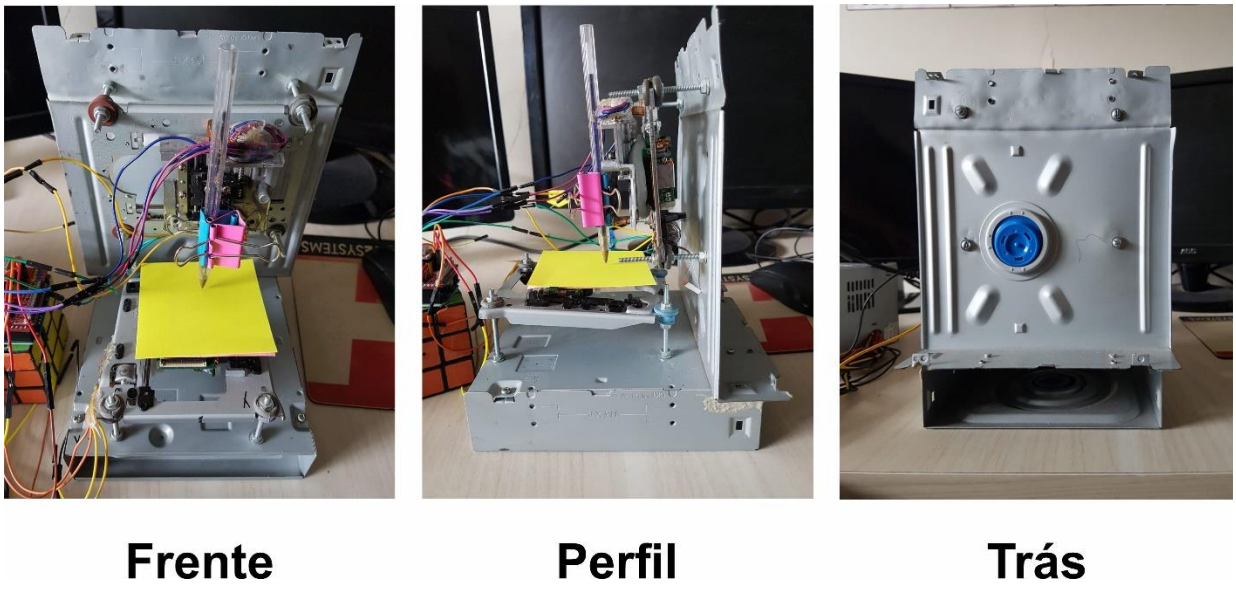
Figura 73 – Etapas da CNC em diagrama de blocos.



Fonte: Elaborada pela autora.

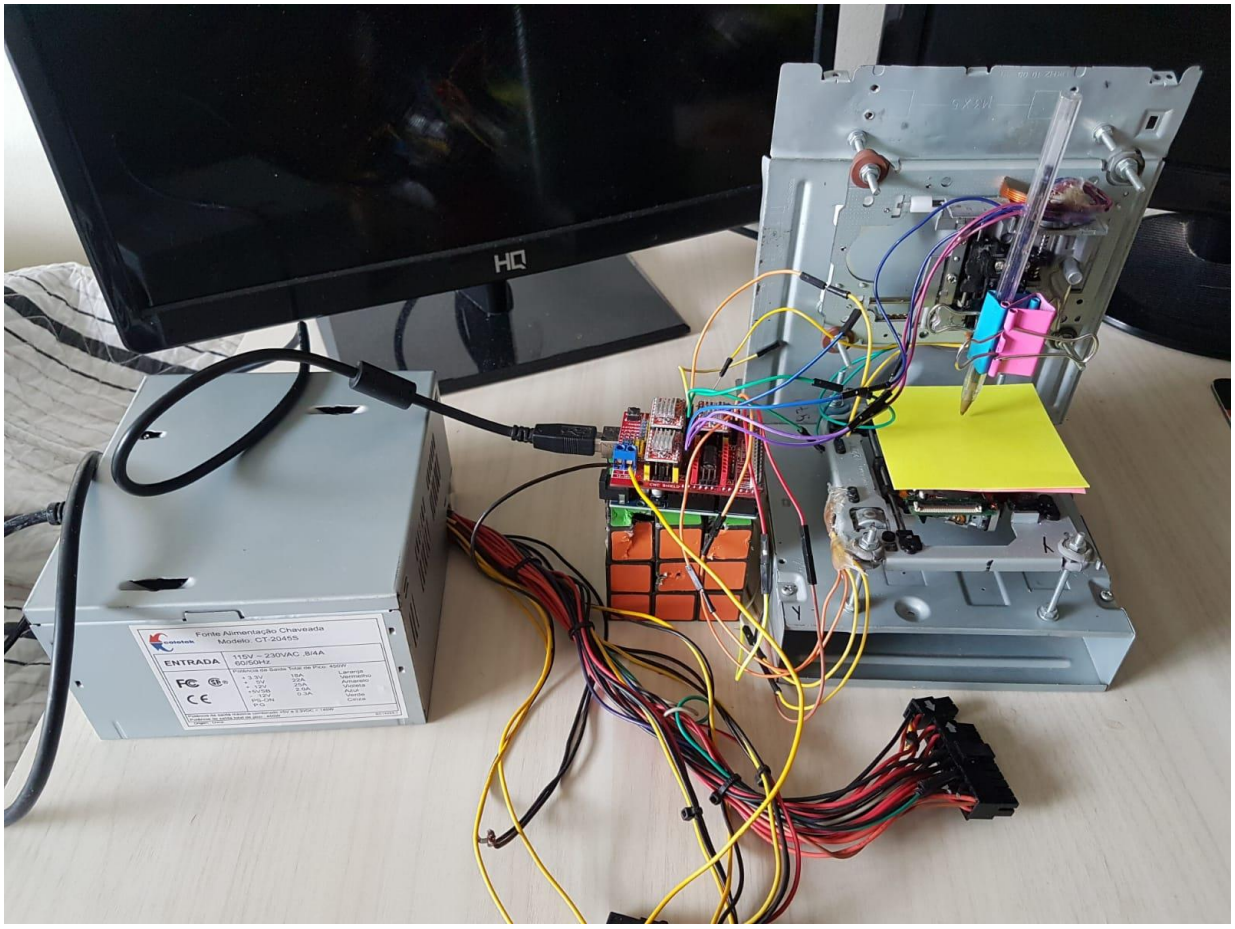
A figura 74 apresenta a máquina em visualização frontal, de perfil e por trás, respectivamente. A figura 75 apresenta a versão final da máquina ferramenta junto com a estrutura, os eixos, as placas e a fonte de energia.

Figura 74 – Máquina CNC com visualização frontal, perfil e trás.



Fonte: Elaborada pela autora.

Figura 75 – *Plotter* CNC.



Fonte: Elaborada pela autora.

5 CONSIDERAÇÕES FINAIS

Este capítulo detalha o desenvolvimento do presente trabalho de conclusão de curso para criação de desenhos gráficos. Para este propósito, desenvolve um amplo estudo de diversas propostas de máquinas ferramentas com o objetivo de criar um projeto que melhor se adeque as condições locais, com o menor custo possível e com produtos acessíveis no mercado nacional. Assim, fabrica-se uma máquina de controle numérico computadorizado artesanal de pequeno porte.

Dada a complexidade e a quantidade de áreas tecnológicas envolvidas, surgiram vários desafios durante a fabricação do protótipo, exigindo a integração do conhecimento obtido durante o curso de Engenharia da Computação.

O intuito de soma dessa pesquisa, tem como ação primordial o direcionamento ao interesse de instituições acadêmicas em relação a simplicidade, ao custeio do projeto e ao aprendizado adquirido durante todo o desenvolvimento do trabalho.

São investigados trabalhos relacionados e estudada a bibliografia corresponde à área de desenvolvimento de máquinas CNC. Em seguida, desenvolve-se a mini CNC artesanal. Posteriormente ao seu desenvolvimento físico, são analisados *softwares* que servem para realizar desenhos gráficos. Neste propósito utiliza-se *softwares* de código aberto que contêm todas as exigências necessárias à máquina ferramenta em questão, o que facilitou em muito a plotagem dos desenhos.

5.1 Testes

No propósito de analisar os resultados obtidos pela máquina ferramenta em pauta, primeiramente avalia-se a funcionalidade do protótipo quanto a sua descrição. Posteriormente, avalia-se todas as peças separadamente realizando ajustes, testa-se o *hardware* e *software* para identificação de possíveis falhas e por fim, realiza-se um novo teste, conectando todas as partes do projeto e fazendo com que a máquina CNC execute uma trajetória definida.

Passando por todas as etapas, primeiramente, depois de obter os *drivers* de CD e disquete, realiza-se a soldagem das pinagens correspondentes do motor de cada eixo. Finalizado os ajustes das peças, inicia-se a montagem dos conjuntos na estrutura, onde realiza-se testes de funcionamento de cada um.

Finalizado a montagem da estrutura, realiza-se a montagem elétrica do conjunto. Os testes elétricos são iniciados com a instalação do Arduino IDE e o carregamento da biblioteca

grbl-master no microcontrolador do Arduino, posteriormente energiza-se as placas, os *drivers* e os motores dos eixos com a fonte de alimentação e configura-se o CNC *Shield* no *Universal G-code Sender* com comandos do próprio programa. Feito isso, realiza-se os testes de funcionamento dos eixos X, Y e Z separadamente. Em teste, é necessário trocar as conexões de um dos eixos de lugar, se ocorrer uma movimentação de forma contrária.

Finalizado o teste elétrico, realiza-se diversos testes na máquina, em um deles são utilizados diversos usos de modelos diferentes de caneta, outro com a caneta acoplada na CNC. É definida a altura do eixo Y, levantando ou abaixando o mesmo, a fim de que a caneta se mantenha alinhada em toda a área de desenho. Essa é uma das etapas mais trabalhosas, pois é preciso que a caneta se mova somente milímetros para cima ou para baixo ou que o eixo Y levante ou abaixe algumas de suas posições até chegar na altura ideal.

Finalizados os testes na máquina, inicia-se a plotagem dos desenhos. Para criação dos desenhos, se utiliza um programa de código aberto no propósito de permitir usar um tradutor de código G. Nessa linha de pensamento, utiliza-se do *Inkscape*, o qual permite adicionar à extensão *gcodetools*, pois com essa ferramenta só é utilizada na criação do desenho de forma gráfica e visual e que depois de alguns ajustes, o desenho se traduz em linguagem de código G.

Finalizado o desenho no *software*, é possível dispor ele na máquina para iniciar a criação física do desenho, porém antes disso, realiza-se testes usando um *software* que permite visualizar arquivos ".ngc", extensão *G-code*, no propósito de economizar tempo ou confusões caso o desenho não esteja na forma especificada do criado no aplicativo de *design* gráfico. Para contornar problemas desse tipo, utiliza-se o CAMotics.

Na primeira tentativa, o desenho não é gerado corretamente, portanto não aparece no simulador, em consequência, a CNC não desenha. Sendo assim, é preciso criar outro desenho. Na segunda tentativa, o desenho é gerado, porém está com tamanho maior que a área de trabalho permitido, precisando a criação de novo desenho e dessa vez com largura e altura dentro do permitido.

Finalizado o desenho, inicia-se os testes de gravura da máquina. O desenho, já no formato ".ngc", é concebido no UGS e depois de enviado para a máquina, esta inicia a trajetória definida. Nesta parte são necessários ajustes na altura da caneta e dos eixos, pois alguns dos lados ilustram tracejados mais marcantes, enquanto outros ilustram tracejados menos significativos pelo fato da máquina ainda estar um pouco desalinhada. Posteriormente o desalinhamento é corrigido, não ocorrendo mais esses problemas.

6 CONCLUSÃO

Para concluir este trabalho pode-se comentar a respeito dos objetivos gerais e específicos estabelecidos que se propôs para a criação de uma mini *plotter* artesanal que se utiliza de um Arduino Uno, um CNC *Shield*, de *softwares* livres e de elementos mecânicos encontrados em sucatas de *hardwares* computacionais. Quanto a isso, o resultado é positivo, o protótipo criado utiliza-se de todos os parâmetros mencionados.

Quanto a justificativa do trabalho, é proposto uma máquina que ilustre desenhos gráficos de forma automática pelo fato deles serem geralmente mais precisos e econômicos no tempo de trabalho, o que demoraria mais se fosse feito de forma manual. Em resposta a isso, o resultado também é positivo, visto que a máquina consegue ilustrar os desenhos criados no *Inkscape*.

Os resultados esperados para o presente trabalho têm resultados positivos, a construção de um controlador numérico computadorizado construído *in home* foi adequadamente realizada.

Os *softwares* escolhidos atendem ao esperado para o projeto e ainda por possuírem uma interface simples, tornam o processo uma tarefa mais simples para qualquer usuário.

O desenvolvimento do presente trabalho de conclusão de curso é de suma importância na aplicação de conceitos adquiridos ao longo do curso de Engenharia de Computação, além de agregar conhecimentos primordiais na área de máquinas de controle numérico computadorizado.

Ao longo do desenvolvimento do projeto, novos desafios se mostram presentes como o aprendizado de uma nova área de estudos na construção de máquinas ferramentas e na busca de precisão e exatidão para a operação da máquina com objetos de menor custo, *drives* de CD e disquete.

7 TRABALHOS FUTUROS

Por se tratar de um protótipo artesanal e ser de baixo custo, mudanças podem ser realizadas na intenção de criar máquinas maiores ou até mesmo de mesmo porte, mas que possam ser comercializadas no mercado. Como trabalhos futuros, sugere-se:

- Construir a CNC em um suporte de madeira substituindo o presente uso de carcaças computacionais;
- Adicionar suporte para fixar o Arduino com a expansão CNC *Shield* no propósito de não deixar as conexões expostas e para obter uma forma de transporte e de manutenção mais simples;
- Adicionar uma mola na parte traseira da caneta para não correr o risco do não alinhamento exato dos eixos com a caneta causar defeito na ilustração desenho;
- Melhorar objeto de fixação da caneta para se adequar a diferentes tamanhos de canetas com um ajuste mais fácil a todos;
- Adicionar sensores fim de curso que detectariam o deslocamento máximo de um eixo.

REFERÊNCIAS

ARAUJO, Warley Monteiro; CAVALCANTE, Maxwell Machado; SILVA, Rogério Oliveira da. **Visão geral sobre microcontroladores e prototipagem com Arduino**. Revista Tecnologias em Projeção, Brasília, v. 10, n. 1, p. 36-46, 2019. Disponível em: http://revista.faculdadeprojecao.edu.br/index.php/Projecao4/article/view/1357. Acesso em: 18 jan. 2021, 14:45h.

ARDUINO. **Arduino Software (IDE)**. 2015. Disponível em: https://www.arduino.cc/en/guide/Environment#toc4. Acesso em: 09 jan. 2020, 15:03h.

ARDUINO. **O que é Arduino?** Disponível em: https://www.arduino.cc/en/guide/introduction. Acesso em: 18 jan. 2021, 14:31h.

AZEVEDO, Américo Luiz de. **Os Primórdios do controle numérico**. 2008. Disponível em: https://www.mundocnc.com.br/historico/. Acesso em: 20 maio 2021.

BANZI, Massimo. **Getting Started with Arduino**. 2. ed. Sebastopol: O'reilly Media, 2011.

BEN-ARI, Mordechai; MONDADA, Francesco. **Elements of Robotics**. Cham: Springer Open, 2018. 308 p. Disponível em: https://link.springer.com/content/pdf/10.1007%2F978-3-319-62533-1.pdf. Acesso em: 22:04 abr. 2021, 00:02h.

BOXALL, John. **ARDUINO WORKSHOP ARDUINO WORKSHOP**: a hands-on introduction with 6 5 projects. San Francisco: No Starch Press, 2013.

CAMOTICS (org.). **CAMotics**: simulação de código aberto e usinagem auxiliada por computador. Simulação de código aberto e usinagem auxiliada por computador. Disponível em: https://camotics.org/. Acesso em: 22 jan. 2021, 11:00h.

CHAKRAVORTY, Dibya. **3D Printer G-code Commands List & Tutorial**. 2020. Disponível em: https://all3dp.com/g-code-tutorial-3d-printer-gcode-commands/. Acesso em: 14 abr. 2021, 01:40h.

CHAPMAN, Stephen J.. **Fundamentos de Máquinas Elétricas**. 5. ed. Porto Alegre: Amgh, 2013. Disponível em: http://www.paftech.com.br/gallery/fundamentos%20de%20maquinas%20eletricas%20chapman.pdf. Acesso em: 09 set. 2021.

CHM (org.). **1801: CARTÕES PERFURADOS CONTROLAM TEAR JACQUARD**: padrões do programa de cartões perfurados de joseph jacquard em um tear de tecelagem. PADRÕES DO PROGRAMA DE CARTÕES PERFURADOS DE JOSEPH JACQUARD EM UM TEAR DE TECELAGEM. 2021. Disponível em: https://www.computerhistory.org/storageengine/punched-cards-control-jacquard-loom/. Acesso em: 20 maio 2021.

COFFLAND, Joseph. **Você pode construir um CNC, mas pode explicá-lo?** 2017. Disponível em: https://buildbotics.com/blog/you-can-build-a-cnc-but-can-you-explain-it/. Acesso em: 20 mar. 2021, 22:31h.

CORTÉS, Fernando Reyes. **Robótica**: control de robots manipuladores. Col. del Valle: Alfaomega Grupo Editor, 2011. 592 p.

CUNHA, Renato. **O tear Jacquard não só revolucionou a indústria têxtil mas foi o primeiro computador do mundo**. 2017. Disponível em: https://www.stylourbano.com.br/o-tear-jacquard-nao-so-revolucionou-a-industria-textil-mas-foi-o-primeiro-computador-do-mundo/. Acesso em: 20 maio 2021.

DEANS, Marti. **G-Code for CNC Programming (2020 Update)**. 2018. Disponível em: https://www.autodesk.com/products/fusion-360/blog/cnc-programming-fundamentals-g-code/. Acesso em: 14 abr. 2021, 11:31h.

FITZPATRICK, M. **Introdução aos processos de usinagem.** Porto Alegre: AMGH, 2013. Disponível em: https://statics-submarino.b2w.io/sherlock/books/firstChapter/115145429.pdf. Acesso em 25 abr. 2021, 23:42h.

FLORENCIO, Heitor Medeiros. **Sistemas Embarcados: Microcontroladores**. Rio Grande do Norte: Ufrn, 2017. Color. Disponível em: https://docplayer.com.br/44367040-Sistemas-embarcados.html. Acesso em: 22 set. 2021.

FORD, Edward. **Make: Getting Started with CNC**. San Fracisco: Maker Media, 2016. 167 p.

GOBI, Nathan. **DESENVOLVIMENTO DE PROTÓTIPO DE MÁQUINA CNC DE BAIXO CUSTO PARA PROCESSOS DE CORTE E GRAVAÇÃO EM MICRO E PEQUENAS EMPRESAS**. 2018. 86 f. TCC (Graduação) - Curso de Engenharia Elétrica, Univates, Lajeado, 2018.

HARDWARE, Maker (org). **CNC *Shield* Guide**. Perth: Maker Group Global, 2018, 12 p. Disponível em: https://www.makerstore.com.au/download/publications/CNC-*Shield*-Guide-v1.0.pdf. Acesso em: 12 maio 2021, 22:34h.

HOBSBAWM, Eric J. **A Era das Revoluções 1789-1848**. Rio de Janeiro: Paz e Terra, 2014.

INKSCAPE (org.). **Visão geral do Inkscape**. 2020. Disponível em: https://inkscape.org/pt-br/sobre/. Acesso em: 22 jan. 2021, 11:30h.

LEE, Jan. **Pioneiros da Computação**: john t. parsons. John T. Parsons. 2021. Disponível em: https://history.computer.org/pioneers/parsons.html. Acesso em: 20 mar. 2021, 22:40h.

LIGHTED WAYS TECH. **The Security Vulnerabilities of Embedded Operating Systems**. 2020. Disponível em: https://www.lightedways.com/2020/06/the-security-vulnerabilities-of.html. Acesso em: 22 set. 2021.

LIMA, Thiago. **AGC – O primeiro grande Sistema Embarcado**. 2014. Disponível em: https://www.embarcados.com.br/agc-primeiro-grande-sistema-embarcado/. Acesso em: 25 fev. 2021, 18:15h.

MAMEDE FILHO, **João Instalações elétricas industriais: de acordo com a norma brasileira NBR 5419:2015** / João Mamede Filho. 9. ed. Rio de Janeiro: LTC, 2017.

MARCICANO, João Paulo P. **Introdução ao Controle Numérico**. 2020. Disponível em: http://sites.poli.usp.br/d/pmr2202/arquivos/aulas/cnc.pdf. Acesso em: 08 abr. 2021, 01:10h.

MECÂNICA INDUSTRIAL (org.). **O que é máquina de controle numérico**. 2017. Disponível em: https://www.mecanicaindustrial.com.br/o-que-e-maquina-de-controle-numerico/. Acesso em: 24 mar. 2021, 00:05h.

METZ, Devan. Arduino CNC *Shield*: Guia do comprador: escudos arduino cnc. ESCUDOS. 2020.

MOLLE, Harald; ADAMS, Josh; WARREN, John-David. **Arduino Robotics**. New York: Apress, 2011.

POLASTRINI, Fernando Henrique. **DESENVOLVIMENTO DE UMA MÁQUINA CNC DE BAIXO CUSTO COM SOFTWARE E HARDWARE ABERTOS**. 2016. 101 f. TCC (Graduação) - Curso de Engenharia Elétrica, Instituto Federal Minas Gerais, Formiga, 2016.

POZZEBOM, Rafaela. **O que são sistemas embarcados?** 2014. Disponível em: https://www.oficinadanet.com.br/post/13538-o-que-sao-sistemas-embarcados. Acesso em: 24 fev. 2021, 13:40.

RODRIGUES, Lucas; DELMONDES, Rômulo. **CNC INTERCAMBIÁVEL APLICADA A CONFECÇÃO DE PLACAS DE CIRCUITO IMPRESSO**. 2016. 129 f. TCC (Graduação) - Curso de Engenharia Elétrica, Pontifícia Universidade Católica de Goiás, Goiânia, 2016.

SCHILDT, Herbert. **Java The Complete Reference**. 11. ed. New York: Mc Graw Hill Education, 2019. Disponível em: https://pdf.zlibcdn.com/dtoken/d7021a8499b3e7edca26f9e86027d1da/Java_The_Complete_Reference,_Eleventh_Edition_by__3697965_(z-lib.org).pdf. Acesso em: 08 nov. 2021.

SMARTEC (org.). **Uma breve história da automação industrial**. 2018. Disponível em: https://www.smartec-automacao.com.br/blog/15532-2/. Acesso em: 05 abr. 2021, 21:05.

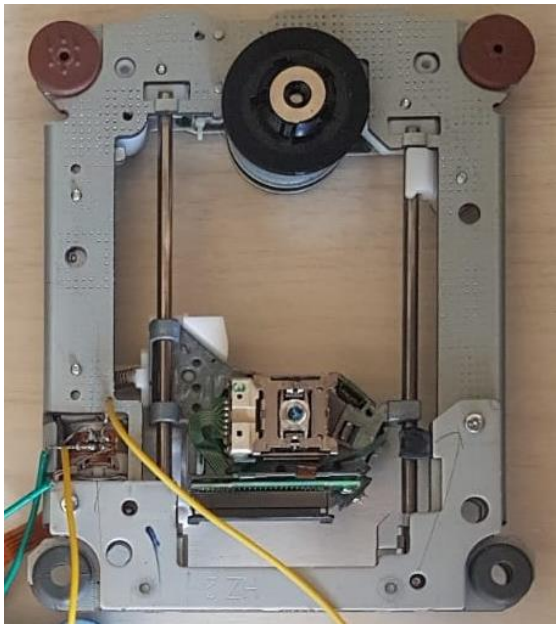
SMID, Peter. **CNC Programming Handbook**: a comprehensive guide to practical cnc programming. 2. ed. New York: Industrial Press, 2003. 529 p. Disponível em: https://3lib.net/book/1074203/fa98be?dsource=recommend. Acesso em: 13 maio 2021, 17:14h.

TECNOLOGIA, Cnc (org.). **Tecnologia CNC**: o que e cnc?. O que e CNC?. 2014. Disponível em: http://cnctecnologia.no.comunidades.net/index.php. Acesso em: 15 abr. 2021, 01:15h.

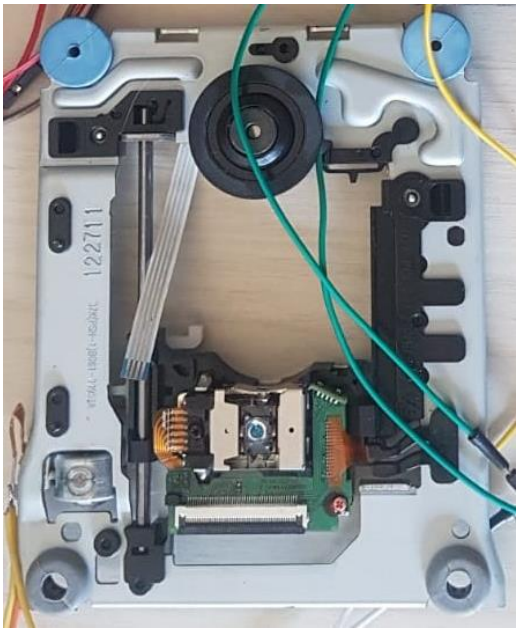
TREJO, Moacir del. **Cartão perfurado**. 2016. Disponível em: http://sites.unoeste.br/museu/cartao-perfurado/. Acesso em: 20 maio 2021.

WOLF, Marilyn. **Computers as Components**: principles of embedded computing system design. 3. ed. Waltham: Elsevier, 2012. 508 p.

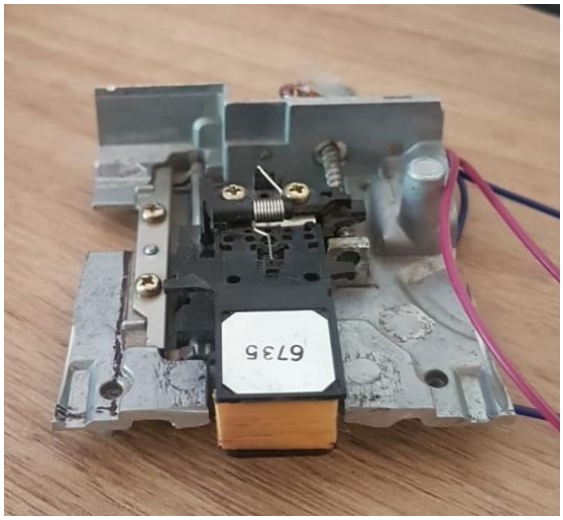
ANEXO A – FOTOS DOS CONJUNTOS UTILIZADOS



Anexo A1 – Eixo X.



Anexo A3 – Eixo Y.



Anexo A2 – Eixo Z.

Anexo A4 – Grampo.

Anexo A5 – Parafusos, Arruelas e Porcas.

Anexo A8 – Prancha de Impressão.

Anexo A6 – *Driver*s CD-ROM.

Anexo A9 – *Driver* de Disquete.

Anexo A7 – *Drivers* A4988.

Anexo A10 – Dissipadores de Calor.

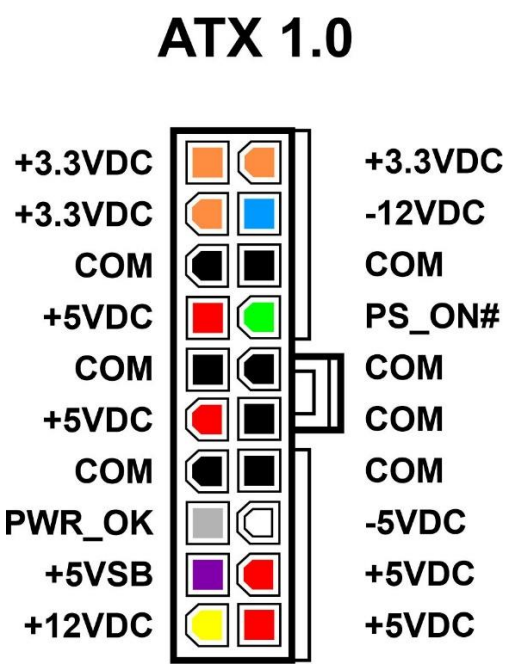
Anexo A11 – Fonte de Energia.

Anexo A12 – Mini Jumpers.

ANEXO B – FONTE DE ENERGIA

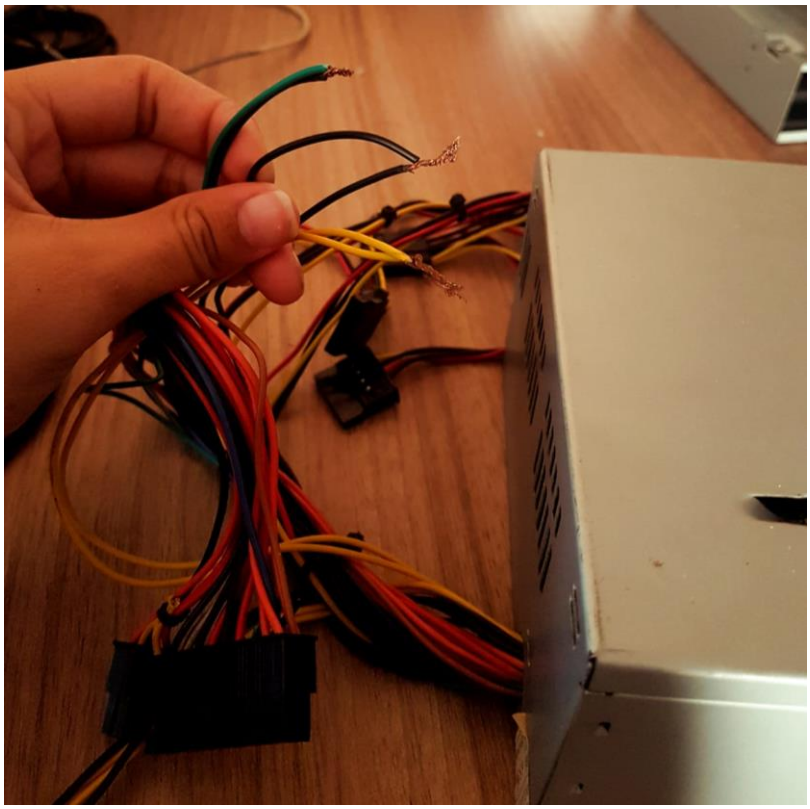
Para alimentar o circuito é necessário a conexão em uma fonte de energia, sendo a melhor opção a fonte de bancada, que permite controlar a tensão de entrada no projeto. Uma boa alternativa para caso não haja uma fonte de bancada, como é o caso do presente trabalho, é uma fonte de energia, como as fontes ATX usadas em computador.

A fonte de computador possui diferentes cores de fios e cada um com um propósito para melhor discernimento no uso do equipamento. Os fios pretos possuem tensão de 0 Volts, sendo ele o terra (GND), os fios laranjas possuem tensão de +3.3V, os fios vermelhos +5V, os fios amarelos 12V, os fios azuis -12V, o fio verde em contato com um fio terra liga a fonte, o fio cinza liga o LED para indicar o funcionamento da fonte e o fio roxo liga o LED para indicar *Stand-by* da fonte. O anexo B1 apresenta as pinagens dos conectores da fonte ATX 1.0.



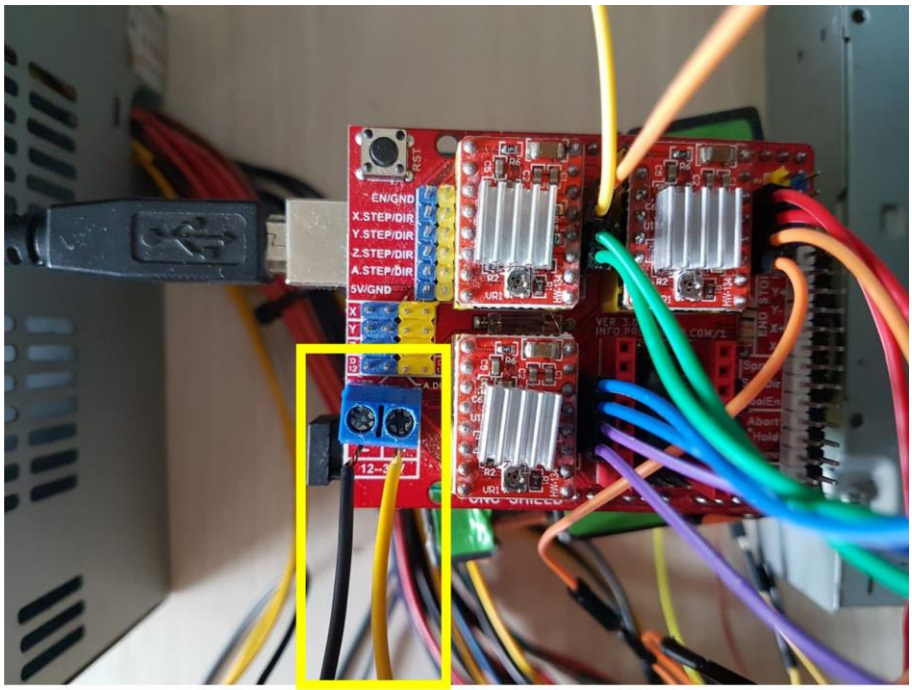
Anexo B1 – Conectores da Fonte ATX 1.0.

Para que seja possível o uso da fonte de computador no projeto, é necessário juntar 1 fio preto com 1 fio verde, isso faz com que a fonte ligue direto quando ligá-la na tomada. Para a conexão no CNC *Shield*, é necessário ligar 1 fio preto com outro fio preto e 1 fio amarelo com outro fio amarelo, assim, tem a conexão positiva e negativa necessárias no *driver*. O anexo B2 apresenta a fonte de alimentação usada no presente trabalho.



Anexo B2 – Fonte de alimentação.

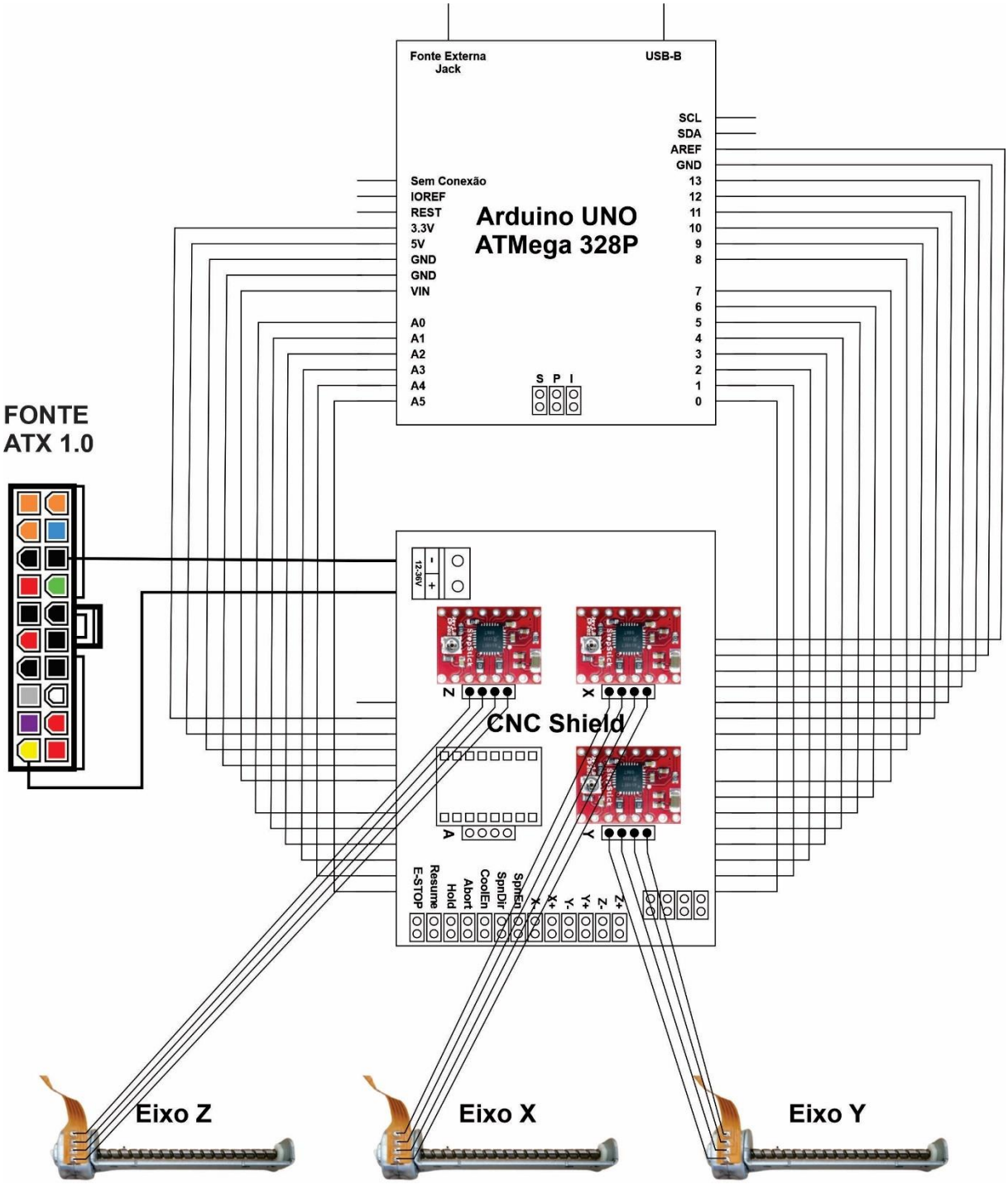
No CNC *Shield*, na parte inferior, há um sinal de menos (-), onde é plugado o fio preto da fonte e um sinal de mais (+), onde é plugado o fio amarelo. Para o presente trabalho, a junção dos dois fios amarelos utiliza-se no positivo do CNC *Shield* e os dois fios pretos pluga-se no negativo para conexão no *driver*. O anexo B3 apresenta o terminal de ligação presente no CNC *Shield*.



Anexo B3 – Ligação CNC *Shield* com a fonte de energia.

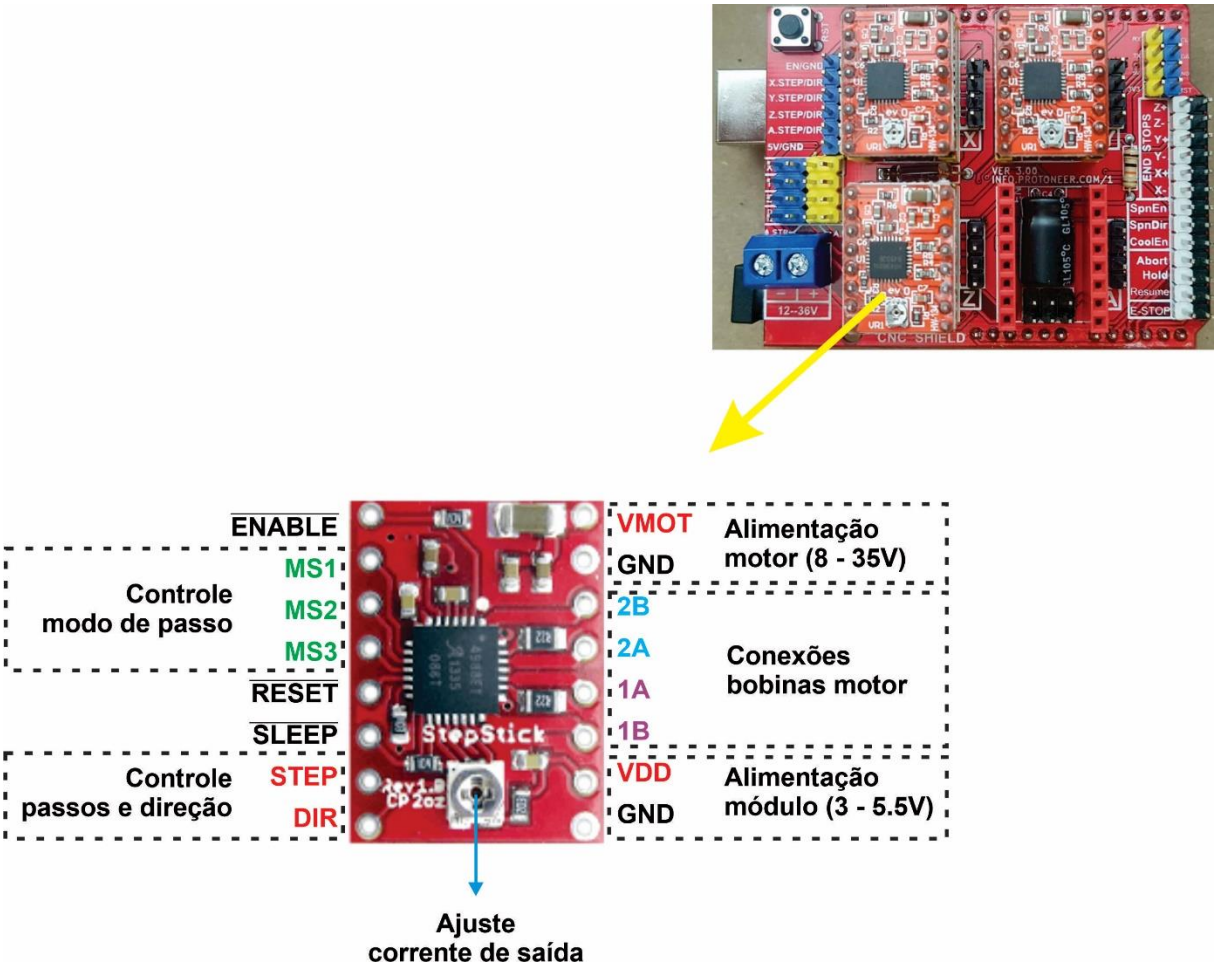
ANEXO C – ESQUEMATIZAÇÃO

Nesse anexo será apresentado uma esquematização em blocos das ligações nas placas no presente trabalho de conclusão de curso.



Anexo C1 – Esquematização placas.

Driver A4988 para o Arduino Uno.



Anexo C2 – Pinagens do *driver* A4988.