

# Electromechanical Conversion of PCB Milling Machine into a 3-Axis CNC

By:

Bashar Bawatna

MASTER OF SCIENCE IN ENGINEERING PHYSICS- MECHANICAL  
ENGINEERING

UNIVERSITY OF CENTRAL OKLAHOMA

EDMOND, OKLAHOMA

MAY 2023

MASTER THESIS SUBMITTED TO THE FACULTY OF THE GRADUATE  
COLLEGE OF UNIVERSITY OF CENTRAL OKLAHOMA

IN PARTIAL FULFILLMENT OF REQUIREMENTS OF DEGREE OF  
MASTER OF SCIENCE

MAY 2023

Electromechanical Conversion of PCB Milling Machine into a 3-Axis CNC

---

Electromechanical Conversion of PCB Milling Machine into a 3-Axis CNC

---

Thesis Title

**Bashar Bawatna**

---

Author's Name

**5/3/2023**

---

Date

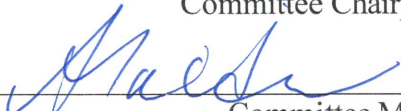
Jackson College of Graduate Studies at the University of Central Oklahoma

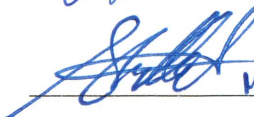
A THESIS APPROVED FOR

PARTIAL FULLFILLMENT OF REQUIREMENTS FOR MASTER OF SCIENCE IN  
ENGINEERING PHYSICS-MECHANICAL ENGINEERING

By

  
\_\_\_\_\_  
Committee Chairperson

  
\_\_\_\_\_  
Committee Member

 **Muhsen Khundakar**  
\_\_\_\_\_  
Committee Member

\_\_\_\_\_  
Committee Member

## Acknowledgements

I am forever grateful for having made it this far in my academic career. I would like to thank all the people that have supported me over the years in ensuring I finish my degree.

I would like to thank my family for always supporting me in not only my degree but in everything I do. A special thank you to my parents who always motivate me to improve not only academically but as a person. They always push me to be the best version of myself and I will forever be grateful for having them in my life. I would also like to thank my brothers who always encourage me to do my best and be successful.

I would like to thank Dr. Moussa for his continued support in this project and helping me improve as an engineer. I have learned so much from him and his great attitude has made this project less stressful than it should be. Without his encouragement and support this project would not have been possible.

I also want to thank UCO and the UCO Engineering Department for allowing me to become a Masters student at the university, for allowing me to work on the decommissioned PCB Machine and for their financial support in realizing this project.

Thank you to everyone who supported me, this experience has been a wonderful memory that I will never forget.

Name: Bashar Bawatna

Date of Degree: May 2023

Title of Study: Electromechanical Conversion of PCB Milling Machine into a 3-Axis CNC

Major Field: Mechanical Engineering

**Abstract:** In the fast-growing technological world that we live in today, out-of-date embedded technologies are now replaced with compact, more powerful and energy efficient electronic hardware that ensures safer use, flexible productivity, ease of maintenance and more autonomy. In many cases the associated mechanical hardware and components are perfectly functional, nonetheless incompatible, and oftentimes costlier to upgrade. This was the case for the decommissioned LPKF ProtoMatS100 circuit board plotter (i.e., PCB printer) housed in the electronics lab at the University of Central Oklahoma. All attempts to restore the device were unsuccessful due to the higher cost of repair hence the machine was left unused. The subject of the current research is the transformation of the decommissioned PCB printer to a fully operational 3-Axis CNC milling router for machining small mechanical devices. To achieve this objective, we devised and implemented an in-house system control hardware and software that employs the original power distributor, newer stepper and DC motor drives, a spindle, an AVR Microcontroller, and a custom-built user-friendly control interface. The upgrades are currently on schedule and the project completion date is set for the spring 2023. The reinvented CNC milling router is expected to machine soft materials with accuracy comparable to modern CNC milling machines.

# Table of Contents

Chapter	Page
Chapter 1 .....	1
1.1 Introduction.....	1
1.2 Background Information.....	1
1.3 Project Objectives.....	2
1.4 Thesis Organization.....	3
Chapter 1.....	3
Chapter 2.....	3
Chapter 3.....	3
Chapter 4.....	3
Chapter 5.....	4
Chapter 6.....	4
1.5 Literature Review .....	4
1.5.1 Introduction .....	4
1.5.2 History of CNC.....	4
1.5.3 Modern CNC Machines in Application .....	5
1.5.4 CNC Components .....	5
1.5.5 Cutting Operations.....	6
1.5.6 Cutting Tools.....	8
1.5.7 CNC Control Software.....	9
1.5.8 Human Machine Interface.....	10
1.5.9 GRBL .....	10
1.5.10 Universal G-Code Sender (UGS).....	10
1.5.11 CNC Control Algorithms.....	11
1.5.12 CAD/CAM.....	13
1.5.13 Existing DIY CNC Machines with GRBL .....	14
1.5.14 Promat S100 PCB Machine .....	15
Chapter 2: Electronic Design .....	16
Introduction.....	16
2.1 Electronic Control .....	16
2.2 Arduino Microcontroller .....	16
2.3 Motor Control .....	17

2.4 Limit Switches.....	18
2.5 Power Supply.....	19
2.6 CNC Machine Wiring .....	20
Chapter 3: Mechanical Design.....	21
Introduction.....	21
3.1 Model Overview.....	21
3.2 Design Considerations .....	22
3.3 Component Selection .....	22
3.3.1 Spindle.....	22
3.3.2 Spindle Mount .....	23
3.3.3 Workpiece Vise .....	24
3.3.4 Vise Parallels.....	25
3.3.5 Motor Shaft Extender.....	25
3.3.6 Flexible Couplings.....	25
Chapter 4: Controls Development .....	26
Introduction.....	26
4.1 CNC Software.....	26
4.2 GRBL Features .....	26
4.2.1 Standard Commands.....	26
4.2.2 Motion Commands.....	27
4.2.3 Jogging Mode.....	27
4.2.4 Homing .....	27
4.2.5 Alarm & Error Codes.....	28
4.3 C# Windows Application .....	28
Chapter 5: CNC Milling Router Completion & Testing .....	30
Introduction.....	30
5.1 Completed CNC Milling Router .....	30
Budget Analysis .....	31
5.2 Testing Overview.....	31
5.3 Test Materials & Tools .....	32
5.3.1 Machinable wax.....	32
5.3.2 Aluminum 6061.....	32
5.3.2 Cutting Tools.....	32

5.3.3 Measurement Device .....	33
5.4 Test Operations & Results .....	33
5.4.1 Face Milling .....	33
5.4.2 Pocketing .....	34
5.4.3 Irregular Shapes .....	35
5.5 Aluminum Face Milling .....	36
5.6 Accuracy Results .....	39
Chapter 6: Conclusion & Future Work.....	41
6.1 Conclusion .....	41
6.2 Future Work .....	41
References .....	42
Appendix .....	45
CNC Milling G-Code .....	45
G-Code .....	45
M-Code.....	46
Error Codes .....	47
Alarm Codes .....	48
Operation Manual .....	49
Safety Precautions .....	49
Powering & Connecting CNC Milling Router to UGS .....	50
Manually Operating CNC Milling Router using UGS .....	51
Uploading G-code Programs.....	52
Router .....	52
Visual Studio Code .....	53

## List of Figures

Figure	Page
Figure 1.1: Promat S100 PCB Machine .....	2
Figure 1.2: Driving Mechanisms in Modern CNC Machines .....	5
Figure 1.3: CAM Toolpath Operations.....	7
Figure 1.4: Common CNC Cutting Tools .....	9
Figure 1.5: UGS HMI.....	11
Figure 1.6: CAM Simulation using Solidworks .....	14
Figure 1.7: Promat S100 PCB Machine .....	15
Figure 2.1: Arduino Uno Microcontroller with GRBL Pinout.....	17
Figure 2.2: (a):3DM580S and (b) DQ542ma Stepper Motor Driver .....	18
Figure 2.3: Original PCB Machine Power Supply with Cooling Fan .....	19
Figure 2.4: Connections Diagram for CNC Electronics.....	20
Figure 3.1: Solidworks Model of (a)PCB Machine and (b)Y-axis Rails .....	22
Figure 3.2: RoutER11 from OpenBuilds.....	23
Figure 3.3: 65mm Aluminum Spindle Mount.....	24
Figure 3.4: Genmitsu Workpiece Vise.....	24
Figure 3.5: Vise Parallel Set.....	25
Figure 4.1: GRBL G-Code Commands.....	27
Figure 4.2: Custom-Built HMI .....	29
Figure 5.1: (a)Completed CNC Machine and (b)CNC Milling Router with Covers .....	30
Figure 5.2: Digital Caliper used for Measuring.....	33
Figure 5.3: Face Milling Toolpath using Solidworks CAM .....	34
Figure 5.4: Pocketing Toolpath using Solidworks CAM .....	35
Figure 5.5: Pocketing Dimensions.....	35
Figure 5.6: Measurements of Pockets .....	35
Figure 5.7: (a)Solidworks Model of Irregular Pockets and (b)Machined Parts.....	36
Figure 5.8: Face Milling without Coolant (a) Block 1 (b) Block 2 .....	37
Figure 5.9: Face Milling with Coolant (a) Block 1 (b) Block 2 .....	37
Figure 5.10: Face Milling Roughness without Coolant (a) Block 1 (b) Block 2.....	37
Figure 5.11: Face Milling Roughness with Coolant (a) Block 1 (b) Block 2 .....	37
Figure 5.12: Surface Plot of Blocks without Coolant (a) Block 1 (b) Block 2 .....	38
Figure 5.13: Surface Plot of Blocks with Coolant (a) Block 1 (b) Block 2 .....	38



## List of Tables

Table	Page
Table 5.1: Budget Analysis .....	31
Table 5.2: Aluminum Face Milling Roughness Results .....	38
Table 5.3: Machinable Wax Face Milling Results .....	39
Table 5.4: Square Pocket Test #1 Results.....	39
Table 5.5: Square Pocket Test #2 Results.....	39
Table 5.6: Circular Pocket Test #1 Results.....	40
Table 5.7: Circular Pocket Test #1 Results.....	40

## Abbreviations

CNC	Computer Numerical Control
CAD	Computer Aided Design
CAM	Computer Aided Manufacturing
DIY	Do-It Yourself
HMI	Human Machine Interface
HSS	High Speed Steel
NC	Numerical Control
PCB	Printed Circuit Board
RPM	Revolution Per Minute
UGS	Universal G-Code Sender

# Chapter 1

## 1.1 Introduction

In the fast-growing technological world that we live in today, out-of-date embedded technologies are now replaced with compact, more powerful and energy efficient electronic hardware that ensures safer use, flexible productivity, ease of maintenance and more autonomy. CNC machining has grown tremendously thanks to these innovations. CNC machines are an integral part of manufacturing, they are capable of machining parts of various complexity and material (i.e., wood, plastics, and metal) with speed, precision, and accuracy. As technology has improved, so have CNC machines with control systems becoming more compact while also improving performance and productivity. This compact design has not only led to smaller yet more powerful machines, but also ones that cost less leading to the expansion of CNC machining use from larger industries such as aerospace and defense to smaller companies that produce custom parts for a variety of sectors.

There has even been a growing trend of home projects using complex control systems that were limited to industrial use in the past due to high costs, with CNC machining being a perfect example. Today there are many affordable CNC software packages and even open-source software that allows many users to collaborate on making universal standard software.

## 1.2 Background Information

Electrical hardware in machines can fail for a variety of reasons while the associated mechanical hardware and components are perfectly functional, nonetheless incompatible, and oftentimes costlier to upgrade. This was the case for the decommissioned LPKF ProtoMatS100 circuit board plotter (i.e., PCB printer) housed in the electronics lab at the University of Central Oklahoma. All attempts to restore the device were unsuccessful due to the higher cost of repair hence the machine was left unused and soon after in line for surplus. The subject of the current research is the transformation of the decommissioned PCB printer to a fully operational 3-axis CNC milling router for machining small mechanical devices.

## 1.3 Project Objectives

The project focuses on the reinvention of the Promat S100 PCB plotter in Figure A to a functioning 3-axis CNC router capable of machining small parts of simple and complex geometries out of soft materials such as Machinable wax and Aluminum. To achieve this objective, a plan was set out to target the various elements needed for this transformation, which are briefly summarized as follow:

- a) The design and implementation of a compact control unit that can adapt to the current machine layout as well as motion transmission mechanisms and power distribution.
- b) The design and implementation of the accompanying firmware that controls all aspects of machining operations that the CNC mill will be tasked to do.
- c) The design and implementation of a friendly user interface that permits manual and automatic motion control of the XYZ axis with standard CNC programming language.
- d) The electromechanical transformation of the old pressure punching head to a CNC spindle with convenient Z machining clearance.
- e) The implementation of safety system in the form of limit switches for homing and to prevent machine crashing.



*Figure 1.1: Promat S100 PCB Machine*

## 1.4 Thesis Organization

The objectives outlined in the previous section are individually discussed throughout the rest of this thesis according to the following schedule:

### Chapter 1

Serves as an introduction and focus on the motivation and purpose of this project. It will go over the current uses of CNC and how it has grown from being used exclusively in the manufacturing industry to academic research and even personal use. It will then cover the aims and objectives of the project will be described to accomplish the goal of converting the PCB machine into a CNC milling router. The chapter will then focus on the literature review of the project. It will discuss the decommissioned PCB machine that will be transformed into a CNC router. It will then discuss the development & advancements of CNC technology, and how the use of CNC has changed the manufacturing industry. Finally, the design of current CNC machines will be explained in detail, including the mechanical aspects, the electronic design, and the software and control related components.

### Chapter 2

The second chapter will focus on the electronic changes that will be made to the PCB machine to allow it to work as a CNC milling router. The changes made to the microcontroller and motor drives will be discussed as well as the reuse current electronic components such as the stepper motors and power supply units will be discussed including electronic additions such as limit switches and a cooling fan for the power supply.

### Chapter 3

The third chapter will focus on the mechanical changes made to the PCB machine. A Solidworks model of the PCB machine was created to assess the design considerations that are to be made. The components that will be replaced and added as well as the reused existing components were discussed in detail.

### Chapter 4

The fourth chapter will focus on the software that will be used to control the CNC machine. It will focus on the G-code programming language that is used to send motion commands to the CNC router. It will also focus on the applications used to control and send the G-code commands to the

CNC milling router. A custom-made application for CNC milling router and how communication to the microcontroller is achieved will also be talked about.

## Chapter 5

The fifth chapter will focus on testing and use of the CNC milling router. Parts will be made using machinable wax and aluminum using the CNC milling router and accuracy will be compared to using a commercial CNC milling router.

## Chapter 6

The sixth and final chapter will be the conclusion of the thesis. This will include the closing statements as well as potential future work on the CNC milling router.

# 1.5 Literature Review

## 1.5.1 Introduction

CNC machining has become a staple of modern manufacturing, constantly improving, and used in many applications. With the advancements in computer hardware, the ability of CNC machines to produce complex parts has become more efficient while also reducing the associated costs of building and operating these machines. In addition to computer hardware improving, software has also improved allowing for improved part designs and machining operations. Over time, CNC machines have grown to be used in many applications outside of manufacturing, with their use becoming prevalent in research, academia, and even at home personal use.

## 1.5.2 History of CNC

Numerical Control (NC) was first developed for the aerospace industry due to the demand for complex parts with high precision and quick manufacturing and can be traced back to late 1940s and early 1950s, developed jointly by engineer John Parson, Massachusetts Institute of Technology (MIT), and the US Air Force (Smid, 2008). Initially, NC machines were hardwired and used a fixed logic sequence that cannot be changed at the machine, using a punch card system to input the logic until the advancement in computer technology led to the use of microprocessors and software to control NC machines, allowing users to quickly change commands and routines at the machine (Suh et al., 2008). These changes resulted in the creation of Computer Numerical Control

(CNC). Nowadays, there is little difference between NC and CNC in everyday talk and are generally used to refer to the same thing.

### 1.5.3 Modern CNC Machines in Application

Over time, the technology for CNC continued to improve and became an important part of the manufacturing and metal working industries. The development of smaller and more compact microprocessors led to smaller and more efficient CNC machines that are now used in many different industries. There are a variety of different CNC machines such as lathes, mills, and drilling machines but all are controlled using a computer and move using stepper or closed-loop servo motors. Initially, CNC machining was limited to larger industries due to the cost associated with acquiring and maintaining the machines but as the technology for computers has grown, they have become more widespread for smaller machining shops and even for hobbyist who create parts for at home applications. CNC machining has a massive advantage over regular machining as it leads to a reduction in time to create parts with higher accuracy and repeatability for more advanced features and complex shapes (Smid, 2008).

### 1.5.4 CNC Components

CNC are driven using closed-loop servo motors or stepper motors in smaller machines, attached to a ball screw connected using a coupling or power transmission gear with feedback using an encoder as shown below in Figure 1.1 (Suh et al., 2008). A ball screw is used to convert the motor movement into a linear movement to allow for back-and-forth movement of the workpiece. In addition to the ball screw, linear movement guides are also added to improve accuracy and

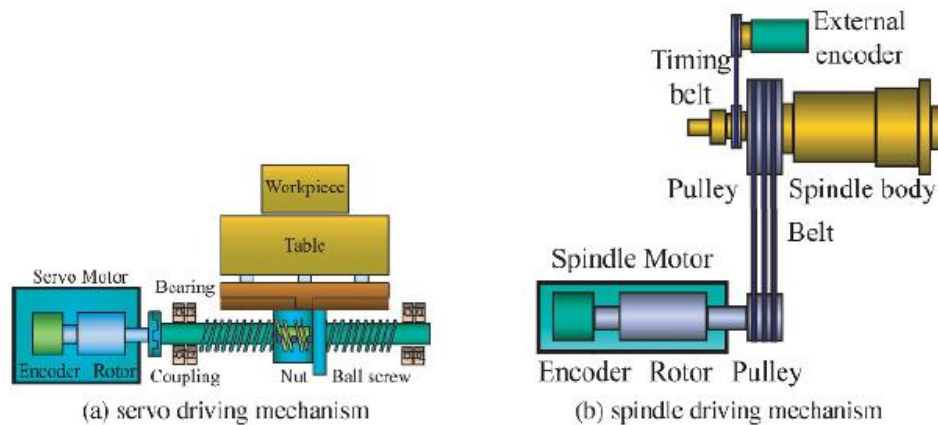


Figure 1.2: Driving Mechanisms in Modern CNC Machines – (Suh et al., 2008)

smoothness of movement. Couplings are added between the motor and ball screw to connect the two devices. Alignment of couplings and the motor shaft is important for accuracy, and flexible couplings are generally used to compensate for any offsets that may be present. Cutting of the workpiece is performed by the spindle motor, which generally requires high speed and torque to get adequate cutting power. Induction motors are commonly used for the spindles over DC motors due to superior qualities in size, weight, efficiency, maximum speed, and maintenance (Suh et al., 2008). Another important aspect of the spindle is being able to maintain constant rotational velocity while cutting, which can be achieved with an electronic speed sensor to maintain constant rpm.

From the electronic point of view, the machine will require a microcontroller, motor drivers, encoders, and limit switches. The microcontroller will be the brains of the machine, where the software will be uploaded, and the memory of the program commands stored. The outputs of the CNC will be sent to the motor drivers, which will be used to send and read the position using velocity and torque control. Encoders are added to read the position of the machine as well as the speed of the axis movement. The most common types of encoders used in CNC machines are optical and magnetic encoders which are generally attached to the shaft of the motors. Limit switches are important for setting the movement limits of the machine, stopping the machine from moving past its intended positional limits and preventing any crashes that could damage the machine.

### 1.5.5 Cutting Operations

There are many different ways that CNC machines can remove material from a workpiece depending on the finish required. Choosing the right operation is an important part of machining parts that can affect the finish on a part. Some popular cutting operations include:

1. Facing: Face milling is the process of removing material from a part to create a smooth, flat surface. This process is also used in the beginning of a CNC operation to ensure a flat finish before other operations are used. An example is shown in Figure 1.2(a).
2. Drilling: Drilling is the process of removing material to create holes with a specific radius and depth in parts. This type of holing operation can only be as big as the drill bit used. An example is shown in Figure 1.2(b).



3. Bore: Boring is the process of removing material to create a large circular radius. Similar to drilling in that it is used to create circular pockets but different in that the size is not constrained to the size of a drill bit. An example is shown in Figure 1.2(c).
4. Pocketing: Pocketing is the process of removing material to create a pocket, slot, or hole with a specific depth and a smooth finish in a part. An example is shown in Figure 1.2(d).
5. Contouring: Contouring is the process of cutting complex shapes along a predefined path. This process is used to create different shapes including arcs, fillets, and other curved surfaces. This is usually done with a smaller tool and allows for a smoother finish. An example is shown in Figure 1.2(e).

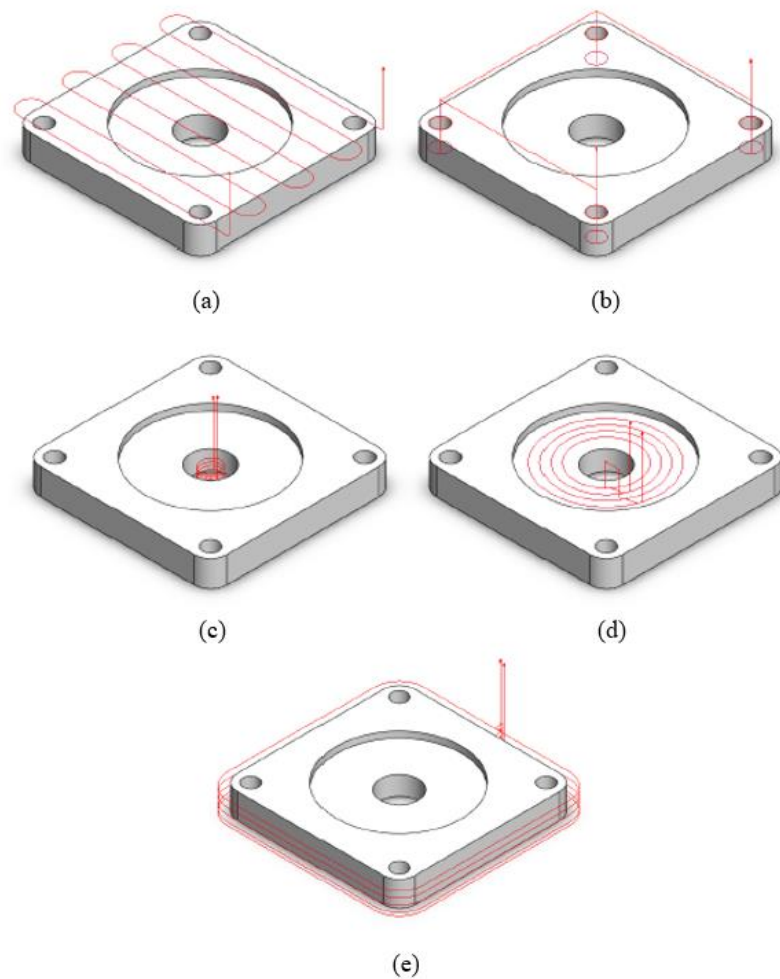


Figure 1.3: CAM Toolpath Operations – (Spilling, 2006)

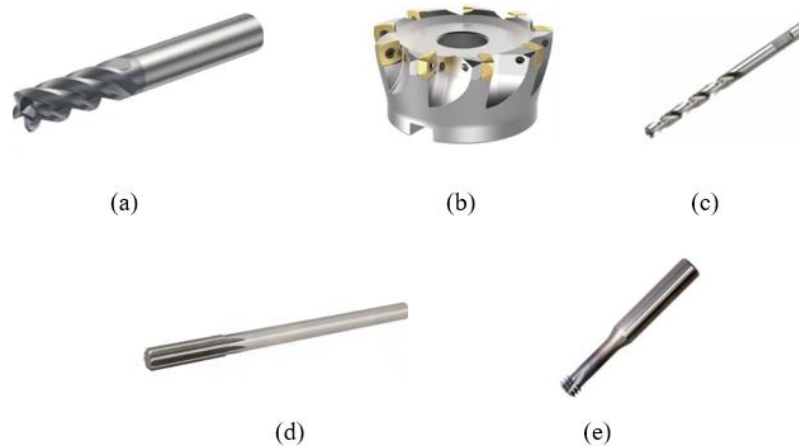
## 1.5.6 Cutting Tools

An important part of any CNC machine is the cutting tools used to remove materials from a work piece. The type of cutting tool used is important for determining the accuracy, efficiency, and finish of the work done on a work piece. Cutting tools are also made of different materials such as high-speed steel, carbide, and ceramics and depends on the material being worked, as well the efficiency and finish required. There are many different types of cutting tools used to remove material with different functions and finishes such as:

1. End Mills: End mills are the most widely used tool in CNC milling, used to create pockets, contours, and slots in a work piece. They come in a variety of different sizes, lengths, and with different flute counts that affect the efficiency of removing material from a work piece. The most common flute count for end mills is two and four, with smaller flute count used for softer materials such as Aluminum and larger flute counts for harder materials such as Steel. The larger the flute counts on an end mill, the more material it can remove in one revolution. The bottom of an end mill can also come in different designs, such as flat, ball nose, and square which can provide a different finish depending on the operation required. An example is shown in Figure 1.3(a).
2. Face Mill: Face mills are cutting tools used to create smooth, flat surfaces. Generally, face mills are larger than other cutting tools, allowing them to remove a larger amount of material at a faster rate. Since they remove a larger amount of material quickly, they are a popular choice for roughing operation. These cutting tools are generally used in the beginning of a CNC operation to ensure a smooth and even flat finish before any other cuts are created. An example is shown in Figure 1.3(b).
3. Drill Bits: Drill bits are also a common tool used in CNC milling, but unlike end mills, they are primarily used for creating holes in the work piece. They generally come with a pointed end and twisted design that allows the material removed from the workpiece to be discharged from the area of work (Smith, 2010). Other types of drill bits are countersink bits that create a conical recession that can be used for keeping a bolt or screw flush. An example is shown in Figure 1.3(c).
4. Reamers: Reamers are used to enlarge and smooth holes in a part. Generally, a smaller drill bit than the radius required is used to create the initial hole and reamers are used

afterwards to expand the hole to the correct size with a smooth finish. An example is shown in Figure 1.3(d).

5. Threaded Mill: Threaded mills are similar to drill bits but are used after a holing operation to create a threaded profile for bolts. An example is shown in Figure 1.3(e).



*Figure 1.4: Common CNC Cutting Tools*

### 1.5.7 CNC Control Software

The standard language for CNC is RS-274, more commonly known as G-code and sometimes G&M-code because many words in the language use the letters G & M and was initially designed when memory in computers was limited, leading to the language being compact (Lynch, 2016). The lines of code in a G-code program are referred to as blocks and are written in a specific format that defines the type of operation and expected movement. The end of a block is defined using a semicolon and comments can be added by enclosing them in parenthesis (Smid, 2004). Codes that begin with G are preparatory commands that are used to preset certain modes of operation such as G01, G02, G03, and G04 for the type of motion or G20 for English units and G21 for metric units while M codes are miscellaneous codes that turn certain operations on such as M30 being the program end or M08 and M09 to turn on and off a coolant pump motor (Smid, 2004). In addition to G&M commands, there are also other commands such as S for spindle speed and F for feedrate. A list of the G & M codes and their operations for CNC milling are shown in the Appendix.

### 1.5.8 Human Machine Interface

In addition to the G-code language, CNC machines require a way for the code to be uploaded and this can come in the form of software applications. HMIs are used to upload program commands, display status of the machine, and edit settings of the machine. HMI can come in the form of a display screen and buttons that is part of the machine or as a computer application that sends commands through software. There are many different software packages, both free open-source and paid for that can be used to control different CNC machines with the most widely used being the Mach3 (Overby, 2010). The most popular free, open-source G-code sender is the Universal G-code Sender (UGS).

### 1.5.9 GRBL

GRBL has been developed to have a variety of features that allow for a wide range of CNC controls. Written in the C programming language, it is optimized to use as many features as possible of the G-code language within the limited space of the Arduino. Some of the features of GRBL include real time motion control, acceleration management, feed-rate planning, variable spindle speed output, soft limits, homing, probing and emergency stops. It is designed to be able to take in commands by the G-code language and therefore works with most CAM software. GRBL can be used with a variety of different machines, including CNC machines, 3-D printers, laser cutters, and milling machines (grbl, n.d.). Development of a 3-axis CNC milling router using GRBL has shown that it is possible to create a highly accurate and precise CNC machine using the software (S.S. Sarguroh et al. 2016).

### 1.5.10 Universal G-Code Sender (UGS)

Universal G-Code Sender (UGS) Platform is a free and open-source G-code sender that is able to work with a variety of different CNC machines such milling machines, 3D printers, and laser cutters using the G-code language. A screenshot of the program UI is shown below in Figure 1.4. Created using java, UGS platform has a community of developers and users who are constantly updating and improving the software. It allows the user to communicate to their machines through a variety of communication protocols including serial, USB, and Bluetooth. It also allows users to set up and operate their CNC machines using an intuitive user interface that can monitor the machine in real time and visualize the movement of the program and can run on many different

computers supporting Windows, MacOS, and Linux. In addition to supporting GRBL, the UGS Platform also supports other CNC controllers such as TinyG, g2core, and Smoothieware ((Universal Gcode Sender, n.d.). This software program will mainly be used to compare to the custom-built application that will be the intended program for operating the CNC mill.

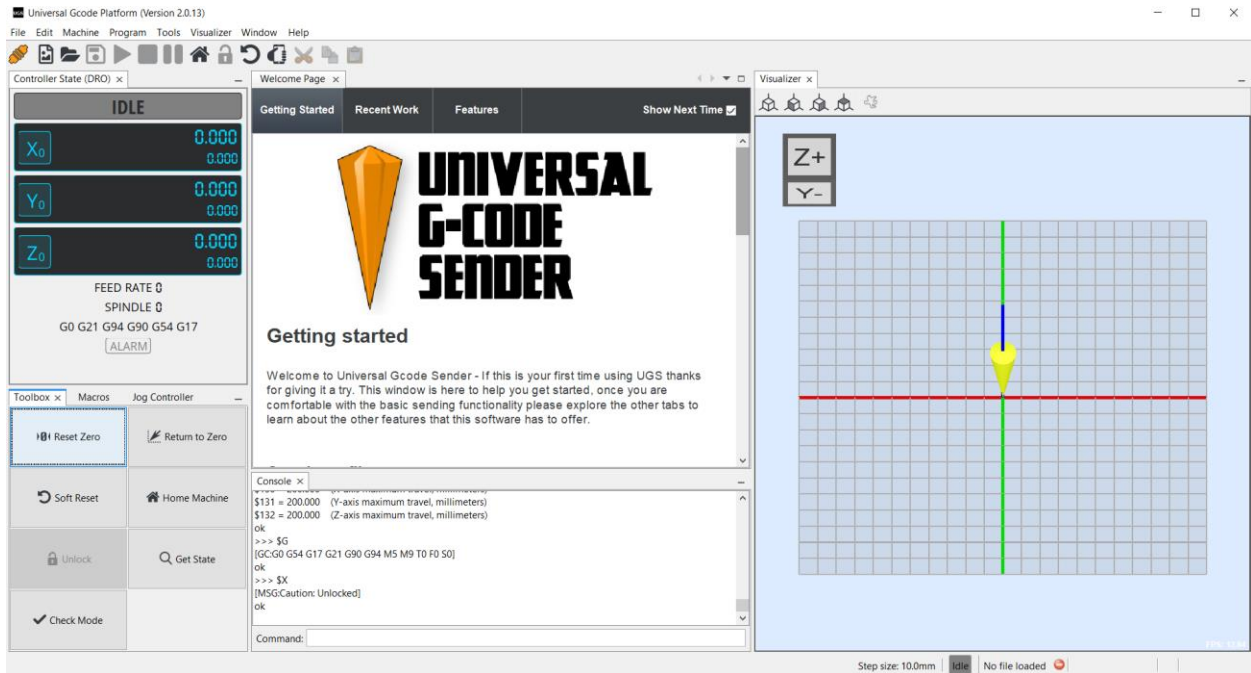


Figure 1.5: UGS HMI

## 1.5.11 CNC Control Algorithms

There are a variety of ways to move the CNC machine to its intended position, either for cutting purposes or just to change the position of the machine. To move the CNC work tool along specific paths, interpolation functions such as linear interpolation, circular interpolation, helical interpolation, and spline interpolation are used (Suh et al., 2008). These interpolation functions are used to create complex geometries during the cutting process. In addition to the movement functions, a coordinate system is required to ensure that the work tool is where it is needed to be during operation.

### 1. Coordinate Systems

It is important to have a defined coordinate system for CNC machining as this will allow the user to know where the work tool is. There are two main coordinate systems that are used in CNC

machining, the machine coordinate system, and the work piece coordinate system (Suh et al., 2008). The machine coordinate system is a point in the machine that can be set manually by the user or by using the homing function of the machine and is referred to as machine home. The workpiece coordinate system is a point on the workpiece that can be set and used to start the machining process. The workpiece coordinate system is also convenient in that it allows you to make changes to design by having a reference point that can be returned to.

## 2. Rapid Positioning

Not all movements in a CNC machine are meant for cutting, whereas sometimes positioning of the work tool as quickly as possible is necessary. This is done using the rapid positioning function and is defined with the G00 command using G-code. Rapid positioning allows the work tool to move to a target area as quickly as possible in a non-circular or helical movement and can move in one or more axis simultaneously (Smid, 2008). This allows the CNC machine to quickly reposition the work tool during the cutting process, reducing the amount of time needed to machine a part. Formulas relating to the time (T, seconds), rapid positioning rate (R, mm/sec, or in/sec) and length of motion(L, mm or in) are shown below (Smid, 2008).

## 3. Linear Interpolation

Similar to rapid positioning, linear interpolation moves the axis in straight line motions but with a specific feed rate during a cutting process. Using the G01 command, linear interpolation is activated and a feedrate can be set using the F command. Linear interpolation is mainly used for contouring and profiling of the work piece. These commands are modal, meaning that once set they don't need to be set again unless a change in motion or feedrate is required. Linear interpolation works by dividing the line of movement into smaller line segments between a start point and end point and is a crucial feature that improves accuracy and smoothness of operation (Smid, 2008). There are three types of motion that can be achieved using linear interpolation, vertical motion & horizontal motion using one axis movement, and angular motion using multiple axis movements.

## 4. Circular Interpolation

Circular interpolation is used to move the CNC machine along a circle and is operated using the G02 (Clockwise) and G03 (Counterclockwise) commands and is mainly used for contouring and profiling in the work piece. This mode is used to create circles, arcs, inside and outside radius,

spherical and conic shapes, radial recesses, grooves, corner breaks, and helical cutting (Smid, 2008). Similar to linear interpolation, this method breaks down the circle into small line segments along a curved path.

#### 5. Helical Interpolation

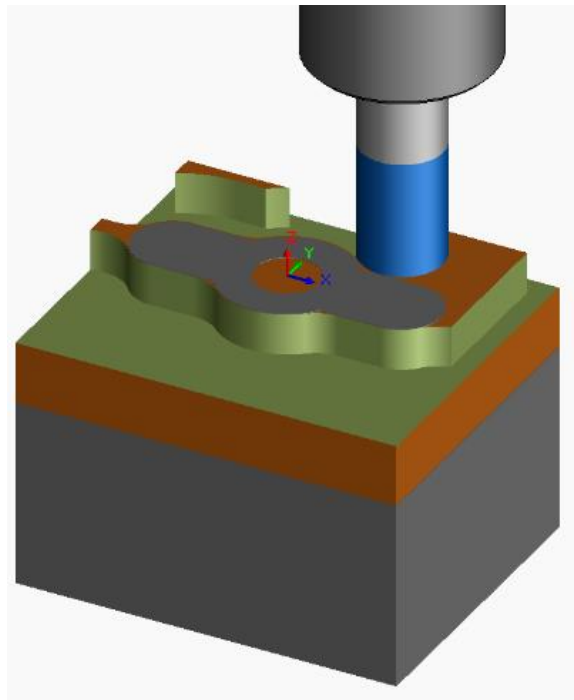
Similar to circular interpolation, helical interpolation creates circles but uses simultaneous motion in three axis to create helical motion. It is essentially circular interpolation in two axis with linear interpolation in the third axis. This motion is mostly used to create a threading profile in parts.

#### 6. Spline Interpolation

Spline interpolation is used to create curved or arced profiles. Spline interpolation uses a start and end point and separates the curved line into small segments that allow for a smooth curved profile along a path.

### 1.5.12 CAD/CAM

In modern CNC machines, parts are designed using Computer Aided Design (CAD) software that allows for complex parts with advanced features to be created. CAD software allows designers to create complex designs for parts at a faster rate than traditional drafting methods. Using Computer Aided Manufacturing (CAM) the tool path and program routines can be created for maximum efficiency and speed. The routines created by CAM software are created in the G-code language and can be processed for different CNC machines. CAM software has been developed to where many parameters such as tool paths, feed rates, and tool selection can be automatically generated (Evans, 2018). There are a variety of different CAD/CAM software that can be used to create the parts and routines for CNC such as Solidworks, Fusion360, and AutoCAD. An example of a CAD/CAM operation using Solidworks is shown below in Figure 1.5.



*Figure 1.6: CAM Simulation using Solidworks*

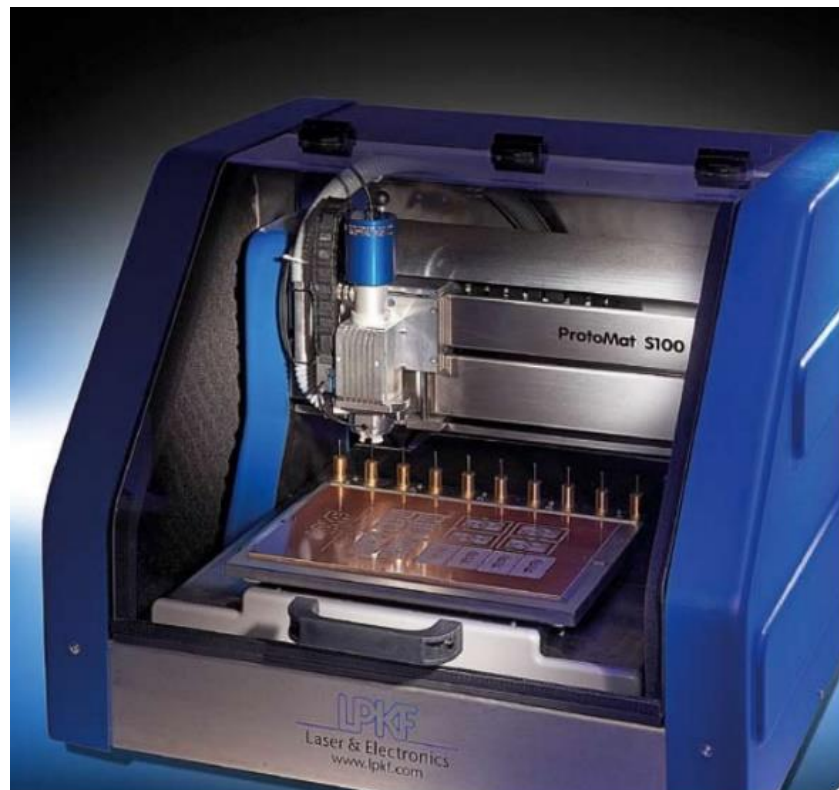
### 1.5.13 Existing DIY CNC Machines with GRBL

There are many CNC machines that have been created outside of manufacturing settings, whether for at home applications or for research to determine whether the development of CNC machines is feasible with current technological advancements. Parajuli et al. were able to develop a CNC machine for drawing and writing purposes with GRBL and were able to get very accurate results with a variety of shapes (2021). A self-improving CNC was developed by Spilling that can fabricate its own part and improve itself, and also used the GRBL library with an Arduino similar to this project (2016). Sundar Pandian and S. Raj Pandian developed a CNC machine that is used in their mechanical engineering lab by students also uses the GRBL and provided a better learning environment due to the open-source nature of the software (2014). Sonawane et al. successfully developed a 3-Axis CNC machine using GRBL and also used UGS similar to how is planned in this project (2017). Jayachandraiah et al. developed a CNC machine for much lower cost than conventional CNC machines and utilized the GRBL library, while minimizing any machining errors (2014). Overall, there have been many successful developments of CNC machines using GRBL that had accuracy and precision comparable to commercial CNC machines.



### 1.5.14 Promat S100 PCB Machine

The Promat S100 PCB machine (Figure 1.6) was used by the UCO engineering department to create circuit boards. Designed and built by Promat International, the machine can be used to create PCBs for testing purposes or for production. It is capable of drilling holes with accuracy and precision and is also capable of routing for tracing lines on circuit boards using a high-speed spindle with speeds up to 60,000 rpm. Initially used for electrical engineering testing purposes, the machine no longer functioned, with it being unable to connect to the computer software. After taking the machine apart and inspecting the electrical hardware, there was a diode that had burnt out on the power supply board. This malfunction likely caused damage to the control board and caused it to no longer connect or be recognized with the associated software. The machine control board and motor drivers were all unserviceable due to being proprietary equipment and therefore no references for the boards were available. The PCB machine was mechanically functional, with the only problems coming from the electrical hardware. The subject of the research will be to convert the PCB machine into a fully functional CNC milling router capable of machining soft metals such as aluminum into complex shapes.



*Figure 1.7: Promat S100 PCB Machine*

## Chapter 2: Electronic Design

### Introduction

Advancement in computer technology has allowed for more efficient, reliable, and cheaper electronics that can be used in a variety of applications. One area that has seen these improvements is in the field of controls where microcontrollers have become more compact and affordable with open-source software that complements their use. This has the advantage over proprietary hardware and software in that they are more easily serviceable with many developers contributing to their continued improvements.

### 2.1 Electronic Control

The Electronics on the original PCB machine were no longer operational. All attempts to power and connect the original machine failed and repairing the original controls was not an option due to the proprietary nature of the components. The controls were also outdated with newer electronic options available that are more efficient with the advantage of having open-source material available for reference.

### 2.2 Arduino Microcontroller

An Arduino Uno microcontroller will be used as the control board for the CNC machine. Arduino is an open-source platform that is popular among hobby, student, and even professional use for its versatility and ease of use. It uses the C++ programming language and has its own Integrated Development Environment (IDE) to write, compile and upload code to the microcontrollers. The Arduino Uno comes with fourteen digital input/output pins, six of which produce PWM signals, six analog input pins, 5V and 3.3V power supply, a 16 MHz timing crystal, 5V power regulator, USB input, and DC power input jack. There are a variety of microcontrollers that Arduino has developed, including the Uno, Mega, and Nano. The Arduino Uno was chosen due to its size, cost, and amount of necessary pins needed to control the planned CNC router. In addition to its own IDE and variety of microcontrollers, the Arduino platform has a variety of open-source software that has a multitude of contributors such as the GRBL library, which will be used to send commands to the CNC machine using the G-code language. The GRBL library has been

customized to work with the Arduino Uno and will be connected to the motor drives and limit switches as shown below in Figure 2.1.

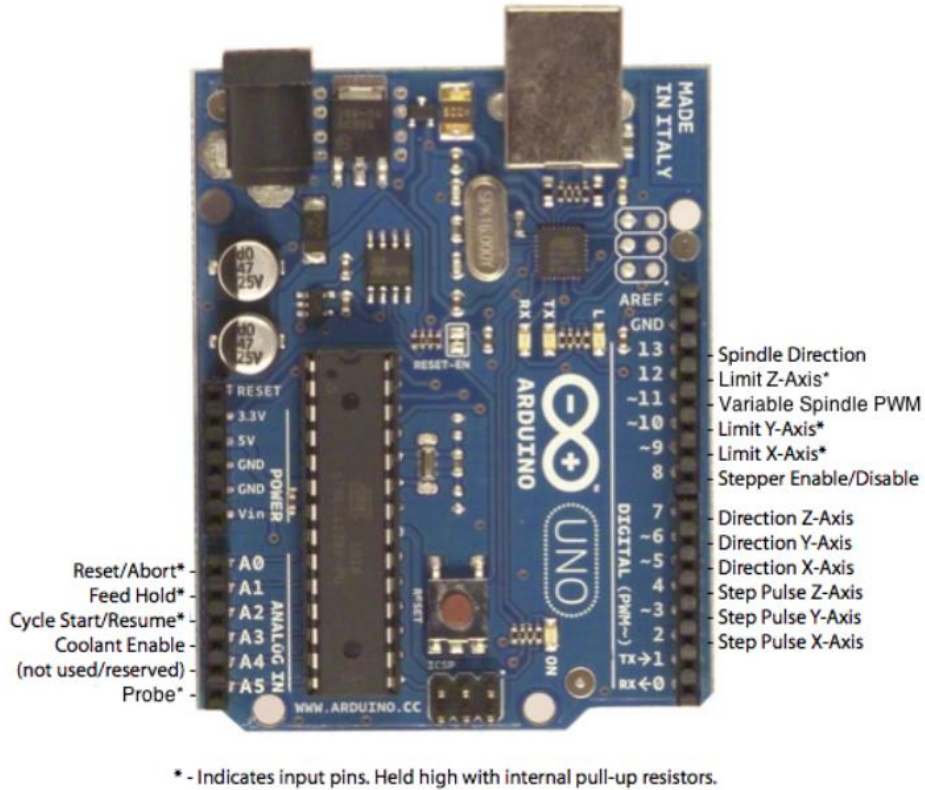


Figure 2.1: Arduino Uno Microcontroller with GRBL Pinout

## 2.3 Motor Control

The original stepper motors of the PCB machine will be used as they are fully operational with no signs of failure. Testing of the stepper motors showed that they work as intended and documentation for the steppers was found online from the original manufacturer. The X & Y axis of the machine used the VRDM366/50LHA00 3-phase stepper motors by Berger Lahr while the Z axis used the C9641-9212KGM 2-phase Vexta stepper motor by Oriental Motor Company. The motor drivers for these steppers were all designed by the original PCB machine manufacturer and had no documentation on their use. The motor drivers will be replaced with 3DM580S 3-Phase stepper motor driver by Cloud Ray for the X & Y-axis stepper motors and the DQ542ma by Wantai Motor for the Z-axis stepper motor as shown below in Figure 2.2. An advantage of using stepper

motors instead of closed-loop servo motors for the axis control is that there will be no need to use encoders to read the position of the motors. This will instead be done by counting the “step” that the motors will make. A disadvantage to this is that stepper motors can have a “misstep” and position might be slightly off than intended. This can be easily corrected though using the homing sequence that will be used during the CNC machines operations.

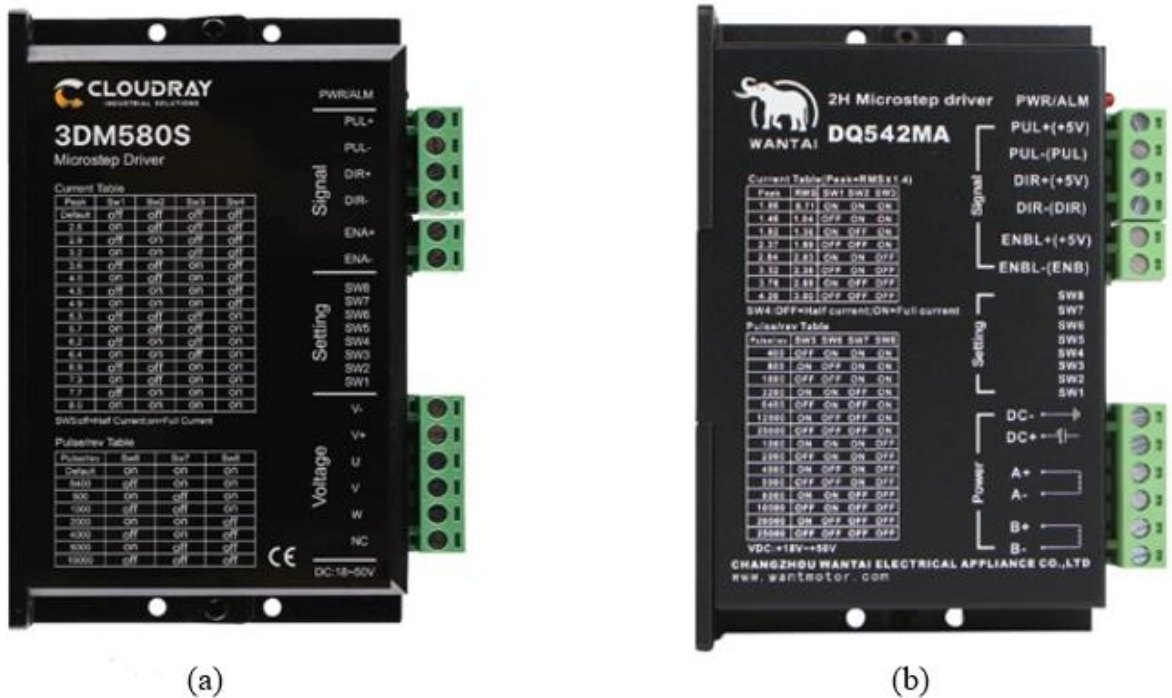


Figure 2.2: (a) 3DM580S and (b) DQ542ma Stepper Motor Drivers

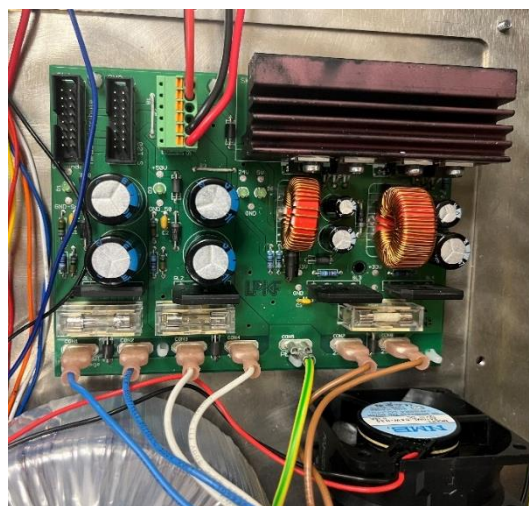
## 2.4 Limit Switches

To prevent the machine from moving past its limits and possibly crashing and causing damage, two limit switches will be added to each axis of the machine. Limit switches work can be set into two different modes, Normally Open (NO) and Normally Closed (NC), where NO is when the switches do not allow current to flow and the switch is off, only activating when the switch is pressed and allowing the voltage to go from 0 to 5V. NC is when the switch is powered at 5V and set to 0V when it is triggered. In modern CNC machines, the most common configuration is the NC position as it allows for better error detection. The limit switches will be wired in the NC

configuration where the voltage is set to 5V and goes to 0V when the limit switches are activated. In addition to the limit switches, a Resistor-Capacitor low pass filter circuit will be added in conjunction with the limit switches to prevent any unwanted noise from affecting the signal. The circuit uses a 4.7 K $\Omega$  resistor and a 100 nF capacitor. To further protect the limit switches from unwanted electrical noise, an EMI shielding cable will be used for the X-axis limit switch wiring.

## 2.5 Power Supply

The original PCB machine power supply circuit (Figure 2.3) will be reused as the only damage noticeable on the supply was a diode that burnt out. The cause of the diode burning out was to prevent any damage from occurring on the power supply. The diode that failed was a ST Unidirectional TVS 1N5908 diode and was replaced with the same model. The power supply provided three output voltages of 5v, 24V, and 50V with the only damage the broken diode causing was that the 50V output was supplying 64V instead of the marked 50V. Replacement of the diode returned the output back to its intended voltage although the 50V output was not used as it was initially used for the original PCB spindle. The 24V output was used for the three Stepper motors used to control the axis movements. The 5V output was used to power the control boards as well as the limit switches. In addition to the power supply circuit, the original PCB machine also included a toroidal transformer that converted the 120VAC power supply from the wall plug into 60 VAC, 40



*Figure 2.3: Original Reused PCB Machine Power Supply with Cooling Fan*

VAC, and 25 VAC reductions. These were attached to the power supply circuit and converted into their DC voltage outputs. The power supply also produced heat during operations and a cooling

fan was installed to help reduce the amount of heat output. The fan added was a 12V 0.16A NMB brushless motor cooling fan with model number 2410ML-04W-B30. The fan drew power from the 24V power supply output that was reduced to 12V using a LM2596 step down DC-DC Buck converter.

### 2.6 CNC Machine Wiring

The motor drives and limit switches were connected to the Arduino Uno and stepper motors as shown below in Figure 2.4. This follows the Arduino Uno GRBL wiring diagram shown in Figure 2.1. The limit switches were not directly connected to Arduino Uno, with the output pins of the Uno connecting to an RC Filter.

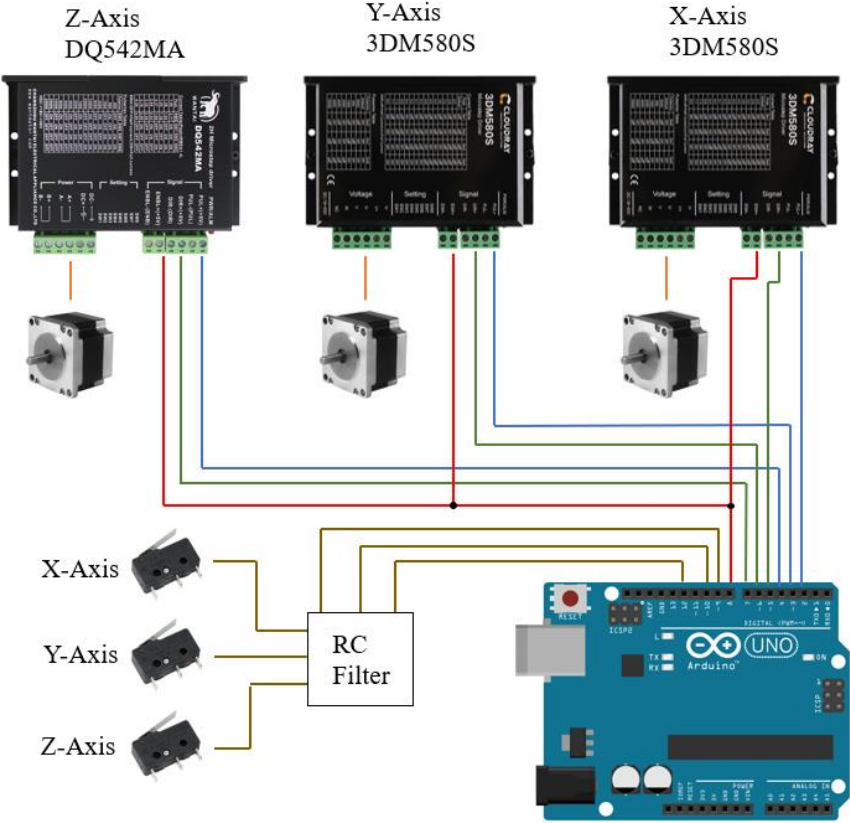


Figure 2.4: Connections Diagram for CNC Electronics

## Chapter 3: Mechanical Design

### Introduction

The mechanical hardware for the PCB machine was mostly functional with small changes needed to convert it into a CNC milling router. The main changes came in the Z-axis where the original machine was designed for PCB drilling and tracing and had to be changed to allow milling. Components were selected to enable the CNC milling router to be able to work on soft metals such as aluminum.

### 3.1 Model Overview

The Solidworks model of the PCB machine is shown below in Figure 3.1(a). The majority of the machine is made up of a steel frame that allows for a rigid support. The X & Y axis movements are controlled using identical stepper motors directly connected to lead screws using a shaft coupler through the center of axis plates. This design is the common design for smaller CNC machines and will not be changed except for replacing the rigid shaft couplers with flexible shaft couplers that will compensate for any misalignments. The part holder plate is connected to the Y axis using 4 bearing supports connected to two support rods which provides strong support against vibrations. The connection of the X axis is similar to the Y axis but with a shorter lead screw and support rods. Figure 3.1(b) shows how the stepper motors are connected to the rods for the Y-axis. The stepper motor used for the Z axis movement was smaller than the stepper motors used for the X & Y axis but was connected to gearbox to increase its load. The movement of the Z axis had to be changed from a primarily punching motion to a controlled continuous motion to allow for accurate CNC machining. In addition to this the Z-axis support will be lifted up about 1.5 inches to allow for a greater height of work to be performed. The initial design was meant for thinner sheets of copper that can be traced for PCBs and is insufficient for larger CNC work to be performed. The spindle used in the original PCB was the Z33-D1100.11 S1 High Frequency Spindle by Jager. This is suitable for PCB board punching and tracing but will not work for a CNC and is to be replaced by a higher torque spindle.

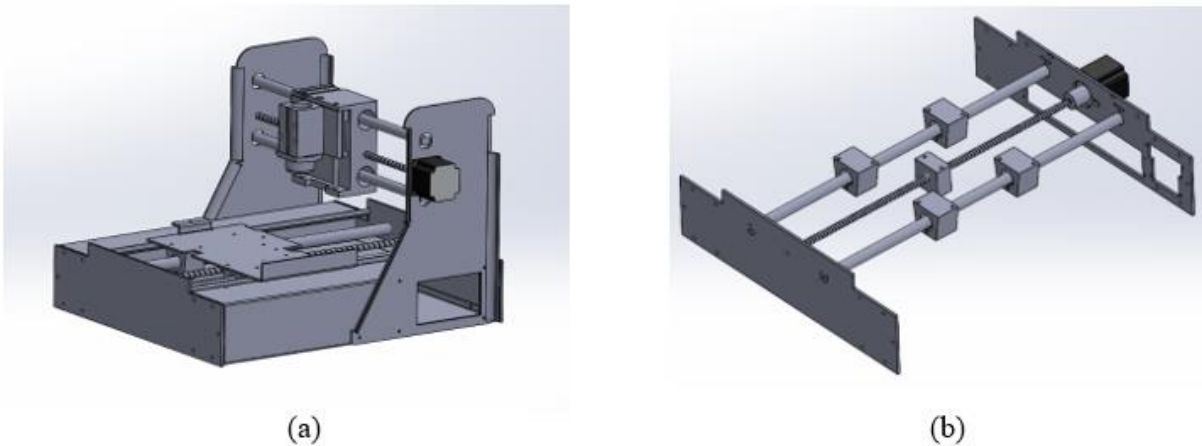


Figure 3.1: Solidworks models of (a) PCB machine and (b) Y-axis Rails

## 3.2 Design Considerations

Considering that the PCB machine was sitting unused due to being unserviceable, the first design consideration will be that the CNC machine is that the proprietary hardware and software is replaced with open-source and interchangeable hardware and software. This will be achieved by using an Arduino Uno as the microcontroller with the open-source library, GRBL. The second design consideration will be that the CNC machine is capable of working on soft metals such as aluminum. This will be achieved by replacing the current high accuracy yet weaker spindle with a bigger and more powerful spindle and changing the Z-axis support to be able to support the weight of the new spindle. Another change to the Z-axis that is to be considered is the change in height, the PCB machine worked on thin materials and had to be increased to allow for larger materials to be machined. This will be achieved by increasing the height at which the Z-axis support is bolted on to the frame of the machine.

## 3.3 Component Selection

### 3.3.1 Spindle

The first major mechanical change considered will be the spindle. The spindle is responsible for cutting the workpiece and one with high speed and high torque is required to be able to work on aluminum. There are two options for cutting the workpiece, a spindle, or a router. Spindles generally have higher torque, faster speeds with better control and can be controlled using software



but require a Variable Frequency Drive to control. Routers on the other tend to be used in smaller CNC applications and require less power to run. The router chosen for the CNC machines is the RoutER11 from OpenBuilds shown below in Figure 3.2. The router has a 1hp motor with a 10K to 32K RPM settings using a 6-dial switch and powered using a 110V outlet. This router will have the necessary power to work on soft metals and features an electronic speed controller to ensure that cutting speed is not affected. The diameter of the router body is 65mm with a collet size of ER11.

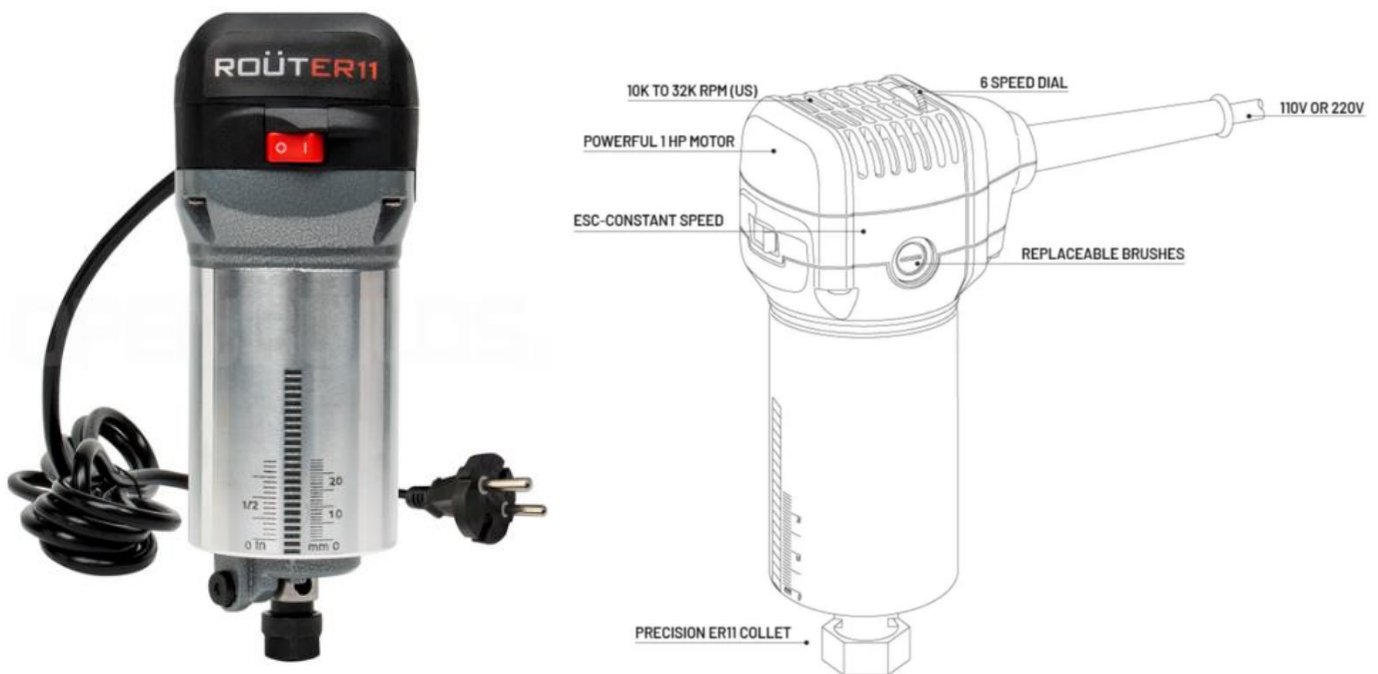


Figure 3.2: RoutER11 from OpenBuilds

### 3.3.2 Spindle Mount

A 65mm Aluminum spindle mount was used to attach the spindle to the Z-axis plate. Four holes were drilled into the mounting bracket to attach to the Z-axis plate. Shown in Figure 3.3.



*Figure 3.3: 65mm Aluminum Spindle Mount*

### 3.3.3 Workpiece Vise

In order to properly machine a part, it must be held down securely, and this is where the vise comes in. The vise chosen is a Genmitsu Aluminum bench vise clamp manufactured by Sainsmart shown below in Figure 3.4. This vise was chosen due to its flatter form allowing it to cleanly fit the CNC machine without affecting the range of the Z-axis. It has dimensions of 260mm(L) x 140mm(W) x 40mm(H) with a clamping range of 0-115mm. It also has a right-angle slider to ensure that the workpiece is squared with the vise. Holes were drilled and threaded in the original PCB Aluminum top to hold the vise in place.



*Figure 3.4: Genmitsu Workpiece Vise*

### 3.3.4 Vise Parallels

A set of 1/8 x 4-inch vise parallels(Figure 3.5) will be used to support the workpiece to the appropriate height and keep the workpiece level. They also serve to stop the workpiece from being pushed down by the spindle as it is being machined.



*Figure 3.5: Vise Parallel Set*

### 3.3.5 Motor Shaft Extender

The X and Y-axis stepper motors couplers are shorter than required to be able to hold the ball screws in their proper position and this will be corrected using motor shaft extenders. The extenders will be made out of steel rods that are 8mm in diameter. A 6mm hole will be drilled into one side with a hole put on top and tapped to be able to hold a set screw.

### 3.3.6 Flexible Couplings

To compensate for any misalignments between the motor shaft and the ball screw, the original rigid couplings used in the PCB machine will be replaced with flexible couplings. The flexible couplings used will be made of aluminum and will be able to attach to the 6mm ball screw shaft and to the 8mm motor shaft extender.

## Chapter 4: Controls Development

### Introduction

There are many ways that CNC machines can be controlled, whether with buttons and knobs, or using software. For the CNC milling router, software will be used to control the machine's movement through the Arduino GRBL library and sent through a G-code sender using serial communication. To make the machine easily serviceable, an open-source sender application will be used as well as a custom-built application for the machine will be used.

### 4.1 CNC Software

The language used for the CNC machine will be G-code. G-code is used in many control systems as the standard for CNC and will be used using the GRBL library for the Arduino. GRBL is a free open-source software that allows for full motion control of CNC machines. GRBL has been constantly updated by different contributors and the version used for this CNC router will be Version 1.1. A diagram of the electronic connections for GRBL library on an Arduino uno is shown below. The GRBL library allows for control of 3 stepper motors, a spindle, with feedback to 3 limit sensors. The remaining inputs will not be used and will instead be sent to the microcontroller through software commands. The GRBL library accepts commands as strings sent through the serial port, and outputs data as a string either as an *ok* or *error* message. This will be taken into consideration for the software to send G-code commands.

### 4.2 GRBL Features

#### 4.2.1 Standard Commands

GRBL accepts motion commands using the standard G-code language using G and M commands. GRBL is constantly being updated to support as many commands as possible using the standard G-code language. A list of the supported G-code commands supported by GRBL is shown below in Figure 4.1.

Support G-Code Commands	
<b>G0, G1:</b> Linear Motions	<b>G54, G55, G56, G57, G58, G59:</b> Work Coordinate Systems
<b>G2, G3:</b> Arc and Helical Motions	<b>G61:</b> Path Control Modes
<b>G4:</b> Dwell	<b>G80:</b> Motion Mode Cancel
<b>G10 L2, G10 L20:</b> Set Work Coordinate Offsets	<b>G90, G91:</b> Distance Modes
<b>G17, G18, G19:</b> Plane Selection	<b>G91.1:</b> Arc IJK Distance Modes
<b>G20, G21:</b> Units	<b>G92:</b> Coordinate Offset
<b>G28, G30:</b> Go to Pre-Defined Position	<b>G92.1:</b> Clear Coordinate System Offsets
<b>G28.1, G30.1:</b> Set Pre-Defined Position	<b>G93, G94:</b> Feedrate Modes
<b>G38.2:</b> Probing	<b>M0, M2, M30:</b> Program Pause and End
<b>G38.3, G38.4, G38.5:</b> Probing	<b>M3, M4, M5:</b> Spindle Control
<b>G40:</b> Cutter Radius Compensation Modes OFF (Only)	<b>M7, M8, M9:</b> Coolant Control
<b>G43.1, G49:</b> Dynamic Tool Length Offsets	<b>M56 :</b> Parking Motion Override Control
<b>G53:</b> Move in Absolute Coordinates	

Figure 4.1: GRBL G-Code Commands

## 4.2.2 Motion Commands

The machine can be moved by using the G-code commands shown above followed by the axis that is to be moved. For example, G0 X50 Y-50 will move the machine in a linear motion without activating the cutter 50 mm in the positive X direction and 50 mm in the negative Y direction.

## 4.2.3 Jogging Mode

Jogging mode is a machine operation that allows the user to manually control the machine using commands, joysticks, or other manual devices. This mode is used to move the machine to locations that allow the user to set workpiece zero coordinates, check alignments, and change tools used in machining processes. The user can move the machine in one or more axes while also having control of the speed in this mode.

## 4.2.4 Homing

Homing is used to find the machine home position. The homing cycle of the machine can be activated by sending the serial command \$H to the Arduino. It begins by moving the Z-axis until a limit switch is triggered and then moves the X and Y-axis until they trigger their limit switches and then the machine sets the position as its home (0,0,0) position. Homing is an important safety

feature for finding the limit of machine movement preventing unwanted movement and crashing. Homing is also important for being able to return to a known and repeatable position. Generally, the homing cycle is run when the machine is first operated to ensure that the machine is in the correct position.

#### 4.2.5 Alarm & Error Codes

Should any errors occur during machine operation, GRBL has a list of errors and alarm codes that can stop the machine and print out codes that describe the problem. The codes are printed out with either error or alarm and a number to show the type of problem the machine is facing. For example, if an unsupported command is read, *error 1* will be printed or if a limit switch is triggered, *Alarm 1* will be printed out. A list of alarm and error numbers is shown below in the Appendix.

### 4.3 C# Windows Application

A custom G-code sender application was created using Visual Studio(VS) C# to interface with the GRBL library using serial communication. The application is created using VS windows form app that is able to communicate with the Arduino using a built-in serial port library that sends and receives information in the form of strings. Common G-code commands are programmed to be sent using buttons as well as an entry mode to allow for manual control of the CNC machine. In addition to this, it is possible to open text files and send commands to the Arduino one line at a time, allowing for a full G-code program to be sent. While not as comprehensive as the UGS Platform, this application will be built to make it easier to interface with the CNC mill for its intended operations without the many options built in for the variety of machines UGS is intended for. A screenshot of UI is shown below in Figure 4.2.

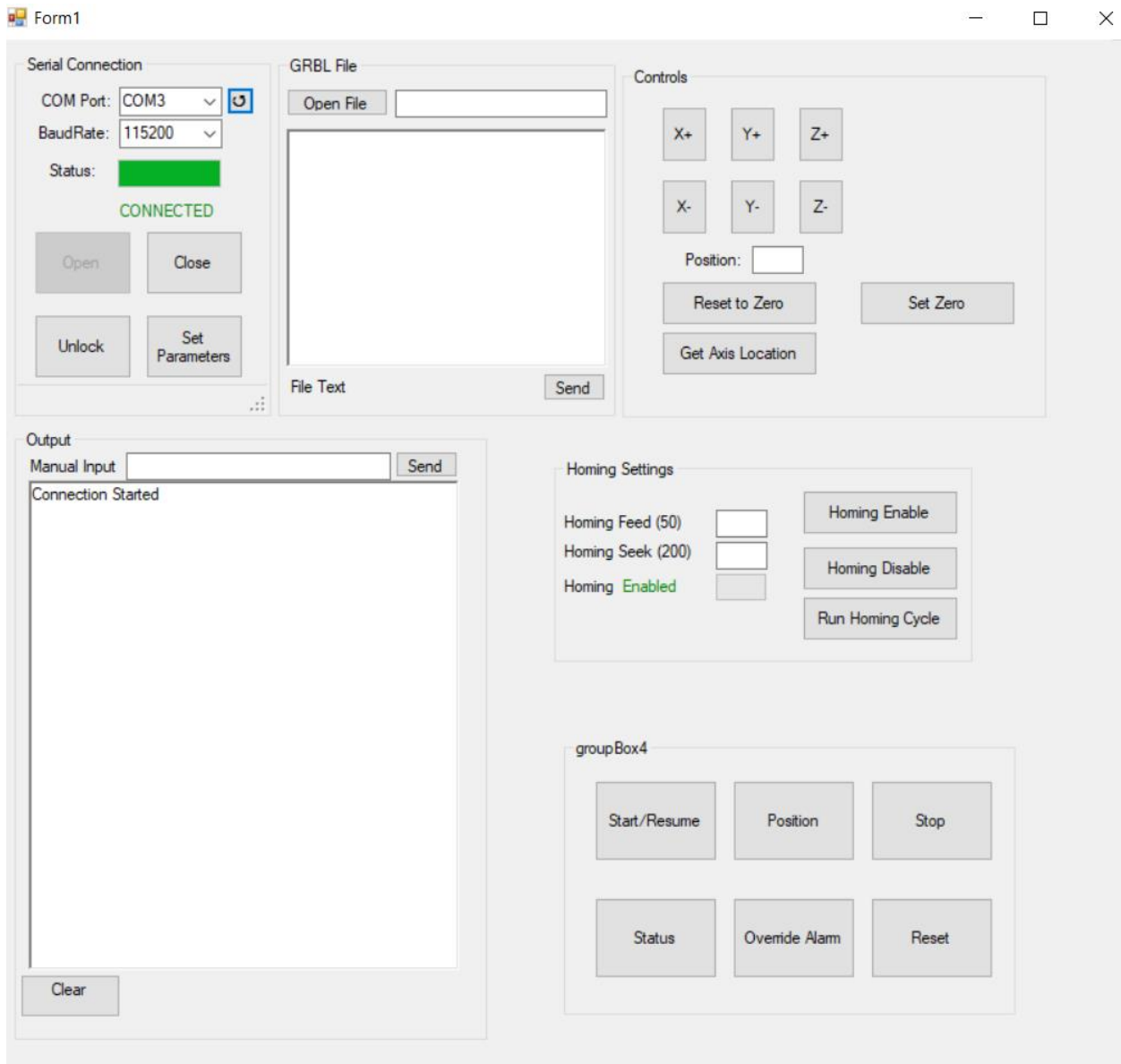


Figure 4.2: Custom-Built HMI

# Chapter 5: CNC Milling Router Completion & Testing

## Introduction

After converting the PCB machine into a CNC milling router, testing is needed to ensure that the machine operates as intended. A variety of shapes and complex designs will be created using machinable wax in order to ensure that the machine is working with the correct dimensions. Machinable wax is a common material used for testing purposes as it is a low cost, easy to machine alternative to metals and plastics.

### 5.1 Completed CNC Milling Router

After finalizing all the electrical connections and assembling the mechanical components, the CNC milling router was completed. All the electrical connections were placed as neatly as possible under the work area and are not exposed to work operations and the rails and lead screws used for the axis motions were also greased for smoother movement. In addition to the changes, the original coverings used for PCB machine were placed back on the CNC milling router to cover the work area and prevent any material being cut from flying around and making a mess in the workspace. The completed CNC milling router is shown below in Figure 5.1.



Figure 5.1: (a) Completed CNC Milling Router and (b) CNC Milling Router with Covers



## Budget Analysis

Table 5.1: Budget Analysis

PART	PRICE	QUANTITY	TOTAL COST
ROUTER11	\$149.99	1	\$149.99
3DM580S	\$43.99	2	\$87.98
DQ542MA	\$51.99	1	\$51.99
GENMITSU VISE	\$69.99	1	\$69.99
PARALLEL SET	\$47.99	1	\$47.99
ARDUINO UNO	\$28.50	1	\$28.50
SPINDLE MOUNT	\$27.88	1	\$27.88
FLEXIBLE COUPLING	\$9.99	3	\$29.97
		Total Cost:	\$494.29

The total cost to convert the PCB Machine into a CNC Milling Router was \$494.29. This includes all the parts that were bought to be used with the machine. In addition to these parts, parts and materials that were already available at UCO were used, such as the motor shaft extensions, brackets to support the limit switches and 3D printing materials. Overall, the total cost to convert the PCB machine into a CNC machine was much lower than one that is bought after taking advantage of the PCB machine that would have been discarded.

## 5.2 Testing Overview

The CNC milling router will initially be tested using the UGS software due to its reliability with working on multiple CNC platforms. Afterwards, the custom-built G-code sender application will be used to ensure that the program is also functional. For the initial tests, the router will be set to a position and moved back and forth to ensure that it moves the intended distance and is able to return to its initial position with accuracy. The homing function will be used to ensure that the limit switches work as expected to prevent any crashes. In addition to the homing sequence, a multimeter was used on the limit switches to ensure that there was no interference with the electrical signal. The different operations of a CNC milling router such as face milling, pocketing, and contouring will be tested with the different cutting tools to ensure that it is capable of standard operations. Common shapes such as circles and squares will be created and measured to ensure . Every part will be face milled before the other operations to ensure an even working area.

Machinable wax will be used due to its easy to machine properties to ensure all settings of the machine are as intended and to confirm that the dimensions are accurate. The designs that will be created will be designed using Solidworks and the G-code commands will be created using Solidworks CAM.

## 5.3 Test Materials & Tools

### 5.3.1 Machinable wax

Machinable wax blocks will be initially used to test the CNC milling router. Machinable wax is useful for testing purposes because it is soft and easy to machine and is a low-cost alternative to metals and plastics. The easy to machine property also ensures that feed rate and spindle speed does not have a major effect on the finish of the material. For Machinable wax, a spindle speed and feed rate of ~10,000 rpm and 136 in/min was used.

### 5.3.2 Aluminum 6061

Aluminum 6061 will be used to test the CNC milling router's ability to work on soft metals. Aluminum 6061 is a popular metal used in manufacturing due to its good strength to weight ratio and resistance to corrosion. Due to the limited tools available, this will only be tested using the face milling operation. For Aluminum face milling, a spindle speed of ~7,000 rpm and feed rate of 235 in/min was used.

### 5.3.2 Cutting Tools

To ensure that the CNC milling router is capable of common CNC operations, a face mill cutter and two end mills will be used. The Face mill is made of carbide tipped steel and can be used for aluminum, but the end mills used are made of High-Speed Steel (HSS) and is more than sufficient for machinable wax but will not be used for aluminum due to carbide cutting tools being the preferred material for machining aluminum. The following cutting tools for the following operations will be used:

1. Face Mill: 3 flute Face Mill with 0.95'' diameter, 0.27'' Cutter length, and 1.66'' total length. This tool will be used for facing and is shown in Figure 5.2(a).
2. End Mill #1: 1/8'' HSS Endmill with 0.125'' cutter diameter, 0.45'' cutter length and 2.3'' total length. This tool will be used for rough milling.

3. End Mill #2: 1/16'' HSS Endmill with 0.0625'' cutter diameter, 0.225 cutter length, and 2.275'' total length. This tool will be used for contour milling.

### 5.3.3 Measurement Device

The final parts were measured using a digital caliper with  $\pm 0.001''$  accuracy. The digital caliper used was the steel core 6'' stainless steel digital caliper shown below in Figure 5.3.



Figure 5.2: Digital Caliper Used for Measuring

## 5.4 Test Operations & Results

All test operations will be conducted using the imperial measuring system, more specifically in inches. The imperial system was used due to the cutting tools available being in imperial system and was kept consistent with what is available. The testing blocks will be measured before and after each operation. The tests will be conducted on the machinable wax blocks.

### 5.4.1 Face Milling

The first testing operation that will be conducted will be face milling. Face milling is generally the first step in most CNC operations and is an important step to ensure other operations are measured correctly. This will be conducted by using tool #1 on all test blocks that will be used in other operations. Two blocks of machinable wax were first used, with the first being face milled down 0.1'' twice, with it measured between each operation. The second block was face milled down 0.2'' twice and was again measured between each operation. The tool paths created using Solidworks CAM is shown in Figure 5.2.

Block 1 Figure 5.4(a)

Dimensions Before Face Milling : 3.050 x 2.060 x 1.845 (in)

After 0.1'' Face Mill : 3.050 x 2.060 x 1.746 (in)

After 0.2" Face Mill : 3.050 x 2.060 x 1.647 (in)

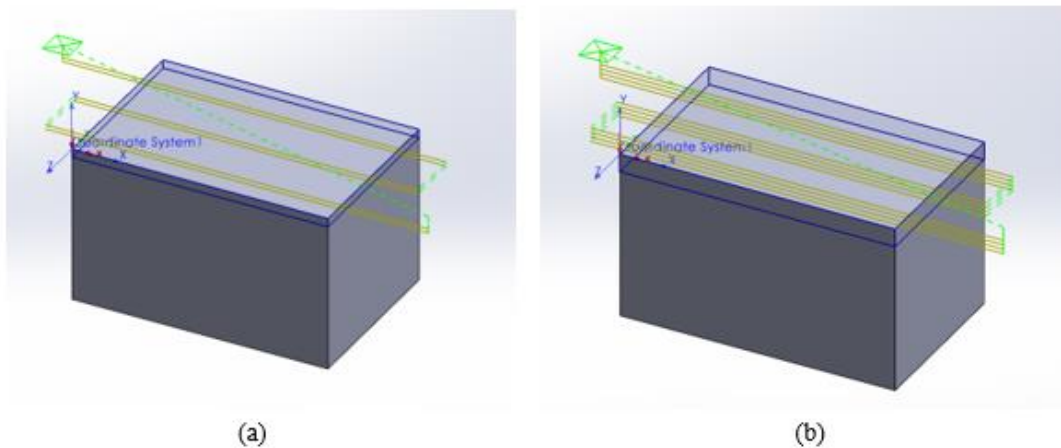
Block 2 Figure 5.4(b)

Dimensions Before Face Mill: 3.045 x 1.837 x 1.867 (in)

After 0.2" Face Mill : 3.045 x 1.837 x 1.670 (in)

After 0.4" Face Mill : 3.045 x 1.837 x 1.471 (in)

The largest error was 0.003" off from the expected value but was off by 0.001" for the other face milling operations. This is in line with what was expected from the converted CNC milling router as this is the general error range for CNC milling routers.



*Figure 5.3: Face Milling Toolpath using Solidworks CAM*

## 5.4.2 Pocketing

The second testing operation that will be conducted will be pocketing. This will be done by creating squares and circular pockets by initially using rough milling to remove larger material first and then contour milling to get a more precise finish. For the first test, a 0.75" square and a circle with diameter of 0.5" with a depth of 0.1" was created as shown below in Figure 5.6 with the tool path shown in Figure 5.5. The final dimensions were 0.752" x 0.753" sides for the square and 0.502" diameter and both had a depth of 0.102". For the second test, the same shapes will be created but with dimensions of 1" sides for the square and 0.75" diameter for the circle and both will have a depth of 0.2". The final dimensions were 1.002" x 1.002" sides for the square and 0.751" diameter and the square had a depth of 0.197" and the circle had a depth of 0.198". Final cut and measurements shown in Figure 5.7. Overall, the dimensions of the shapes were close to

the designed parameters with the largest variation being 0.003” off from the expected value and was most commonly off by 0.002”.

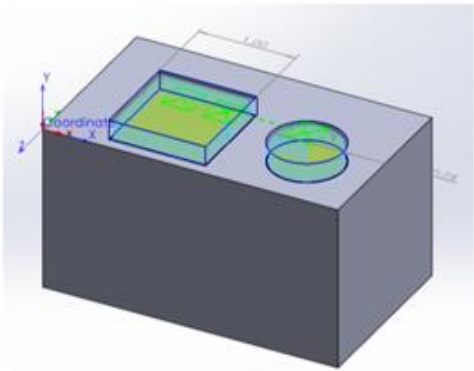


Figure 5.4: Pocketing Toolpath using Solidworks CAM

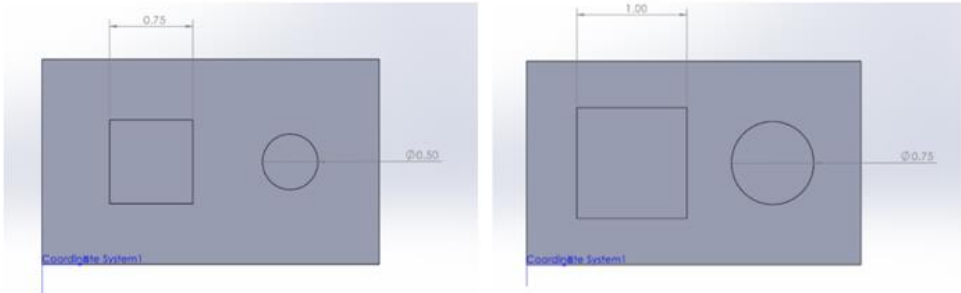


Figure 5.5: Pocketing Dimensions



Figure 5.6: Measurements of Pockets

### 5.4.3 Irregular Shapes

In addition to creating squares and circles, other irregular shapes were created as shown below to ensure that complex cutting operations are feasible.

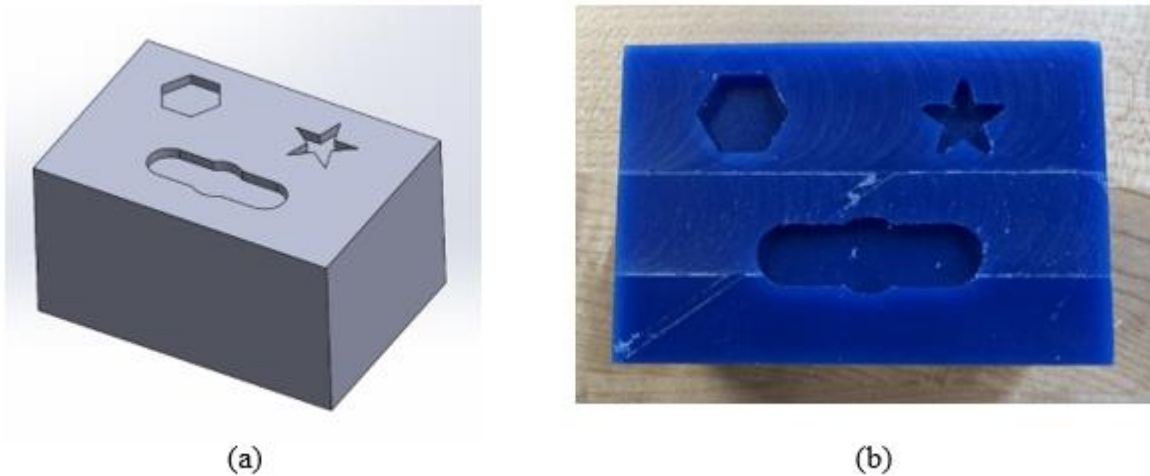


Figure 5.7: (a) Solidworks Model of Irregular Pockets and (b) Machined Parts

## 5.5 Aluminum Face Milling

The only tool available that could machine aluminum properly was the face mill. Face milling of four aluminum blocks was conducted as shown below. The CNC milling router was successfully able to handle the aluminum facing operation with each pass going down 0.01". As shown in Figures 5.8 and 5.9, using coolant during the second facing operation led to smoother finish than the first test and is the recommended method of face milling aluminum due to its relatively low melting point compared to other metals. After face milling of the four aluminum blocks, two with coolant and two without, the surface roughness was measured and compared to standard machining roughness. The average surface roughness of face milled aluminum using commercial CNC machines is around 3.2  $\mu\text{m}$ . The machined aluminum blocks were captured using a Nikon Eclipse MA100L inverted microscope with the NIS Elements D software and processed using ImageJ software. The blocks without coolant measured at 24.29  $\mu\text{m}$  for block 1 and 24.61  $\mu\text{m}$  for block 2 which indicates rough machining marks and surface finish and is far off from commercial CNC. This was also noticeable looking at the aluminum blocks as shown in Figures 5.10 and 5.11. The blocks with coolant measured at 11.66  $\mu\text{m}$  for block 1 and 13.34  $\mu\text{m}$  for block 2 which indicate visible machining marks but smoother finish than without coolant, although still off from commercial CNC machines. Surface plots are also shown in Figures 5.12 and 5.13 further emphasizing the difference in using coolant for face milling.

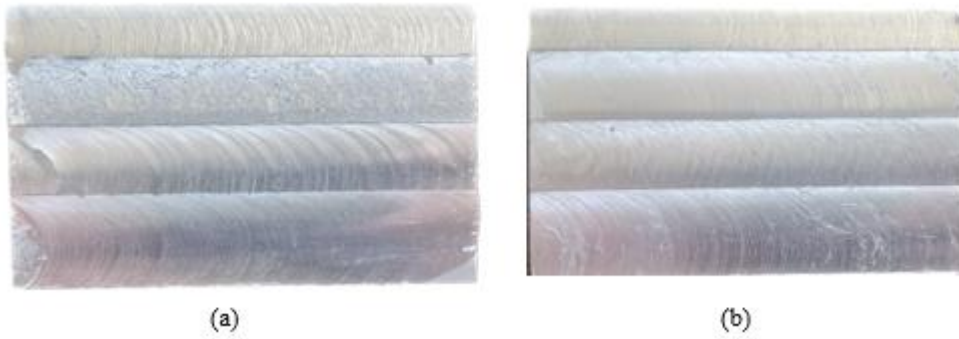


Figure 5.8: Face Milling without Coolant

(a) Block 1 (b) Block 2

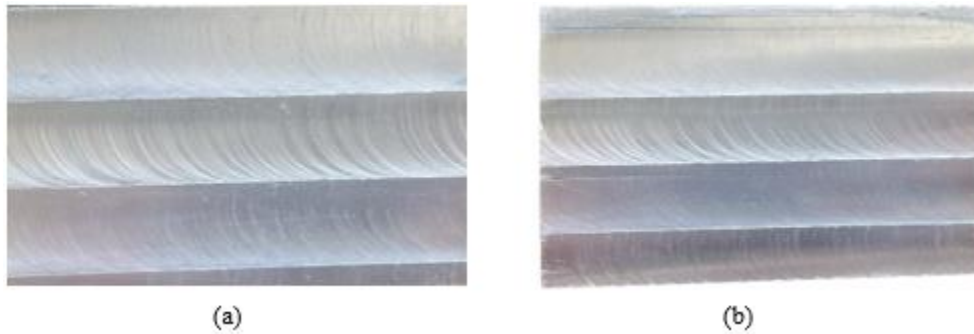


Figure 5.9: Face Milling with Coolant

(a) Block 1 (b) Block 2

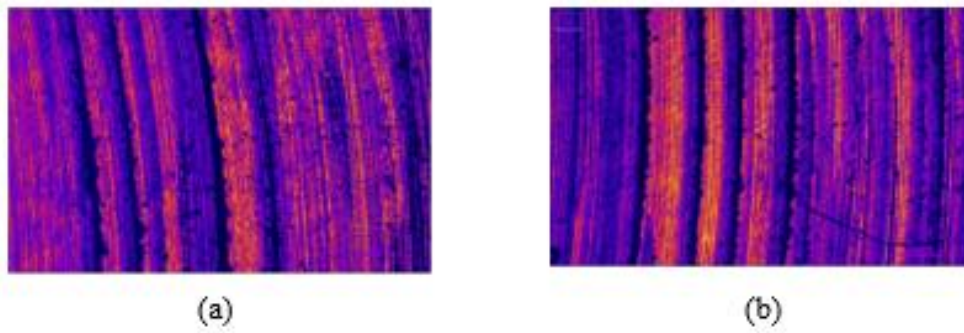


Figure 5.10: Face Milling Roughness without Coolant

(a) Block 1 (b) Block 2

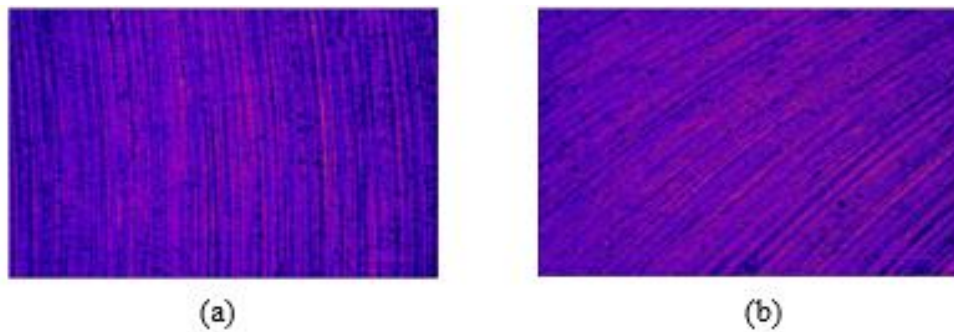


Figure 5.11: Face Milling Roughness with Coolant

(a) Block 1 (b) Block 2

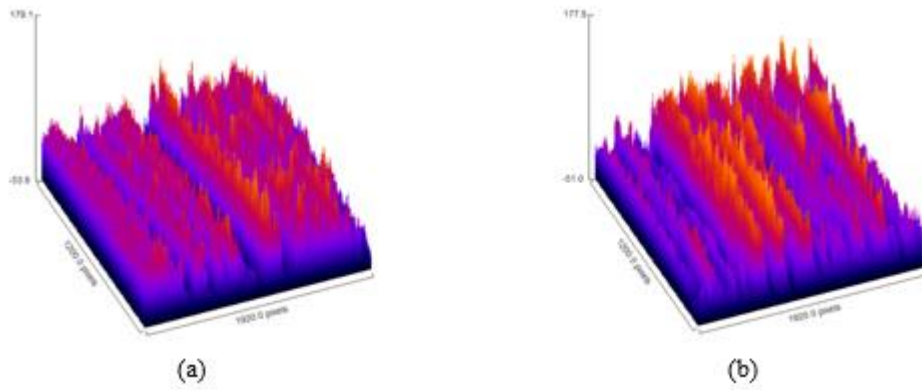


Figure 5.12: Surface Plot of Blocks without Coolant  
 (a) Block 1 (b) Block 2

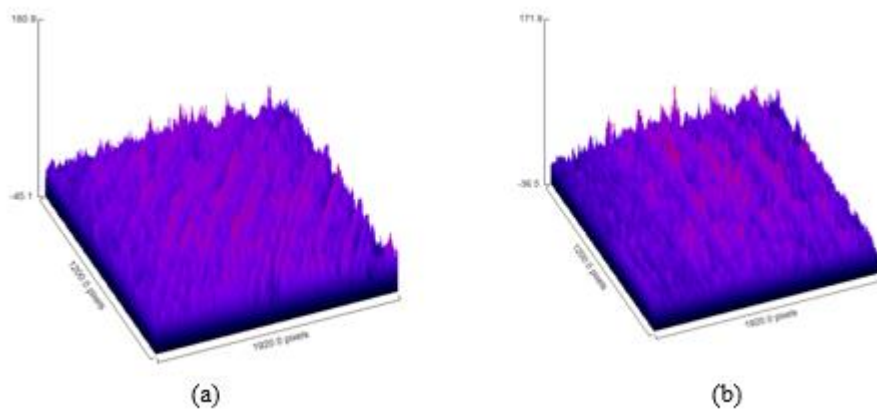


Figure 5.13: Surface Plot of Blocks with Coolant  
 (a) Block 1 (b) Block 2

Table 5.2: Aluminum Face Milling Roughness Results

Block	Surface Roughness (Ra)	Indication
Block #1 No-Coolant	24.29 $\mu\text{m}$	Very Obvious Machining Marks
Block #2 No-Coolant	24.61 $\mu\text{m}$	Very Obvious Machining Marks
Block #1 Coolant	11.66 $\mu\text{m}$	Obvious Machining Marks
Block #2 Coolant	13.34 $\mu\text{m}$	Obvious Machining Marks



## 5.6 Accuracy Results

After finishing testing of machinable wax, the accuracy results are shown in the tables below. The tables show how much deviation there is from the designed measurements.

*Table 5.3: Machinable Wax Face Milling Results*

Face Mill Test	Expected Result (in)	Measured Result (in)	Deviation from Start(in)
0.1" Face Mill #1	1.745	1.746	0.001
0.1" Face Mill #2	1.645	1.647	0.002
0.2" Face Mill #1	1.867	1.870	0.003
0.2" Face Mill #2	1.467	1.471	0.004

*Table 5.4: Square Pocket Test #1 Results*

Square Pocket Test #1	Width (in)	Length (in)	Depth (in)
0.750" Expected	0.750	0.750	0.100
0.750" Measured	0.752	0.753	0.102
Deviation	0.002	0.003	0.002

*Table 5.5: Square Pocket Test #2 Results*

Square Pocket Test #2	Width (in)	Length (in)	Depth (in)
1.000" Expected	1.000	1.000	0.200
1.000" Measured	1.002	1.002	0.197
Deviation	0.002	0.002	0.003

Table 5.6: Circular Pocket Test #1 Results

Circular Pocket Test #1	Diameter (in)	Depth (in)
0.500" Expected	0.500	0.100
0.500" Measured	0.502	0.102
Deviation	0.002	0.002

Table 5.7: Circular Pocket Test #2 Results

Circular Pocket Test #2	Diameter (in)	Depth (in)
0.750" Expected	0.750	0.200
0.750" Measured	0.751	0.198
Deviation	0.001	0.002

The biggest deviation came in the depth of the pockets and milling operations. The bigger error in deviation for the depth can be attributed to error from setup, where the offset is done by using a paper between the workpiece and the tool and relies on the user ensuring that the offset is acceptable. Overall, the result of the testing shows that the CNC machine is highly accurate and repeatable at different dimensions.

## Chapter 6: Conclusion & Future Work

### 6.1 Conclusion

The machine was successfully built and tested. The electronic and mechanical hardware were properly selected and integrated to convert the PCB machine into a CNC milling router. It was tested with UGS to ensure functionality of the GRBL library and had accuracy comparable to common CNC machines. Additionally, the custom-built HMI for controlling the CNC milling router was successful and can be used for machining. Different CNC operations were tested and were conducted on machinable wax as well face milling of aluminum 6061. Overall, the newly developed CNC milling router worked as expected and was considered a success.

### 6.2 Future Work

Although the CNC milling router was successful, upgrades can be made to improve the functionality of the machine. A potential improvement can be made by integrating a vacuum to remove the material removed from the workpiece. Although the removed material is contained within the covers, removing the material through a vacuum will help keep the work area of the machine cleaner and reduce time in between machining different parts. A different workout around to removing material can come in the form of a blower that clears chips during machining operation and prevent material from coming in the way of the cutting tools. Additionally, the router used for the CNC can be replaced with a spindle that is connected to the microcontroller to function without having to be manually turned on and to have more variety in the speeds. The current router works on a 6-speed dial, and this may prove inefficient for materials that require specific RPM speeds. In addition to changing the spindle, a cooling mechanism can also be added to improve finish quality on machined parts and allow the CNC machine to better work on metals such as aluminum. Improvements can also be made for the custom-built HMI by adding a visualizer that shows the toolpath as it is being used. While the HMI is sufficient for working on the CNC, the UGS platform was still superior and was the preferred method of using the CNC machine. Overall, although the CNC milling router worked as expected, it can still be upgraded to perform on a bigger variety of materials.

## References

- Arduino. (n.d.). Arduino Uno. Retrieved April 22, 2023, from <https://www.arduino.cc/en/Guide/ArduinoUno>
- Cloudray.3DM580S Datasheet. Retrieved April 22, 2023, from <https://cloudray2021.ossuswest1.aliyuncs.com/AliExpress/AE04/%E8%AF%B4%E6%98%8E%E4%B9%A6/3DM580S%20User%20Manual.pdf>
- Dassault Systèmes. (2021). SolidWorks 2021-2022 [Computer software].
- Dassault Systèmes. (2021). SolidWorks CAM 2021-2022 [Computer software].
- Diymachining.com. (n.d.). GRBL. DIY Machining. <https://diymachining.com/GRBL/>
- Evans, K. (2018). *Programming of CNC machines (4th ed.)*. McGraw Hill Education.
- Grbl. (n.d.). GitHub repository. Retrieved April 22, 2023, from <https://github.com/grbl/grbl>
- Grbl Wiki (n.d.). URL: [www.github.com/grbl/grbl/wiki](http://www.github.com/grbl/grbl/wiki) (visited on 2023-3-19).
- Jayachandraiah, B., Krishna, O. V., Khan, P. A., & Reddy, R. A. (2014). Fabrication of Low Cost 3-Axis CNC Router. *International Journal of Engineering Science Invention*, 3(6), 23196726.
- LPKF Laser & Electronics. (n.d.). Promat S100 Datasheet. Retrieved April 22, 2023, from [http://www.spezial.cz/pdf/protomat\\_s100.pdf](http://www.spezial.cz/pdf/protomat_s100.pdf)
- Lynch, M. (2016). *Fundamentals of CNC Machining: A Practical Guide for Beginners*. Industrial Press.
- National Institutes of Health (2019). ImageJ (Version 1.52a) [Computer software]. U.S. Department of Health and Human Services, National Institutes of Health
- Nikon Instruments Inc. (2019). NIS Elements D (Version 5.20) [Computer software]. Nikon Instruments Inc.

- Oriental Motor. (n.d.). C9641-9212KGM Stepper Motor. Retrieved April 22, 2023, from [https://www.orientalmotor.com/products/pdfs/stepper-motors-gearheads/stepper-motors-gearheads-specs/C9641-9212KGM\\_Stepper\\_Motor\\_Spec.pdf](https://www.orientalmotor.com/products/pdfs/stepper-motors-gearheads/stepper-motors-gearheads-specs/C9641-9212KGM_Stepper_Motor_Spec.pdf)
- OpenBuilds. (n.d.). Router11 [Product description]. Retrieved from <https://openbuilds.com/builds/router11.9097/>
- Overby, A. (2010). *CNC Machining Handbook: Building, Programming, and Implementation*. McGraw-Hill.
- Pandian, S., & Pandian, S. R. (2014). A Low-Cost Build-Your-Own Three Axis CNC Mill Prototype. *International Journal on Mechanical Engineering and Robotics (IJMER)*, 2(1).
- Parajuli, N., Thakuri, S. S., Shah, S. R., Bista, S., Shrestha, A., & Guragai, M. K. (2021). Modeling and Fabrication of Low Cost 3-Axis Computer Numerical Control (CNC) Machine. *International Journal of Advanced Engineering*, 4(1), 25867652.
- Schneider Electric Motion USA. (n.d.). VRDM366/50LHAOO Stepper Motor. Retrieved April 22, 2023, from <https://www.smamotion.com/products/stepper-motors/berger-lahr/stepper-motor-vrdm36650lhao>
- Smid, P. (2004). *Fanuc CNC Custom Macros: Programming Resources for Fanuc Custom Macro B Users*. Industrial Press Inc.
- Smid, P. (2008). *CNC Programming Handbook: A Comprehensive Guide to Practical CNC Programming*.
- Smith, G.T. (2010). *Cutting Tool Technology: Industrial Handbook*. Taylor & Francis.
- Sonawane, A., Rane, A. B., & Sudhakar, D. S. S. (2017). Development Of A 3-Axis CNC Milling Machine With An Open-Source Controller. *International Journal of Research in Engineering and Technology*, 06(08), 9–15. <https://doi.org/10.15623/ijret.2017.0608002>
- Spilling, T. (2016). Self-Improving CNC Milling Machine [Master's thesis, Norwegian University of Science and Technology]. NTNU Open.

S. S. Sarguroh, A. B. Rane, S. A. Korgaonkar and D. S. S. Sudhakar, "*Elimination of Proprietary Control for Computerized Numerical Control (CNC) Machine*", *Journal of Basic and Applied Research International*, vol. 17, no. 3, pp. 211-217, April 2016.

Suh, S., Kang, S. K., Chung, D., & Stroud, I. (2008). *Theory and Design of CNC Systems*. Springer Science & Business Media.

Universal Gcode Sender. (n.d.). Homepage. Retrieved March 19, 2023, from [https://winder.github.io/ugs\\_website/](https://winder.github.io/ugs_website/)

Wantai Motor. (n.d.). DQ542MA Stepper Motor Driver Datasheet. Retrieved April 22, 2023, from <http://www.wantmotor.com/products/dq542ma-driver.html>

# Appendix

## CNC Milling G-Code

### G-Code

G code	Description
G00	Rapid positioning
G01	Linear interpolation
G02	Circular interpolation clockwise (CW)
G03	Circular interpolation counterclockwise (CCW)
G04	Dwell - as a separate block only
G09	Exact stop check - one block only
G10	Programmable data input - Data Setting
G11	Data Setting mode cancel
G15	Polar Coordinate Command cancel
G16	Polar Coordinate Command
G17	XY-plane designation
G18	ZX-plane designation
G19	YZ-plane designation
G20	Imperial units of input
G21	Metric units of input
G22	Stored stroke check ON
G23	Stored stroke check OFF
G25	Spindle speed fluctuation detection ON
G26	Spindle speed fluctuation detection OFF
G27	Machine zero position check
G28	Machine zero return (reference point 1)
G29	Return from machine zero
G30	Machine zero return (reference point 2)
G31	Skip function
G40	Cutter radius compensation cancel
G41	Cutter radius compensation - left
G42	Cutter radius compensation - right
G43	Tool length compensation - positive
G44	Tool length compensation - negative
G45	Position compensation - single increase
G46	Position compensation - single decrease
G47	Position compensation - double increase
G48	Position compensation - double decrease
G49	Tool length offset cancel
G50	Scaling function cancel
G51	Scaling function

G code	Description
G52	Local coordinate system setting
G53	Machine coordinate system
G54	Work coordinate offset 1
G55	Work coordinate offset 2
G56	Work coordinate offset 3
G57	Work coordinate offset 4
G58	Work coordinate offset 5
G59	Work coordinate offset 6
G60	Single direction positioning
G61	Exact stop mode
G62	Automatic corner override mode
G63	Tapping mode
G64	Cutting mode
G65	Custom macro call
G66	Custom macro modal call
G67	Custom macro modal call cancel
G68	Coordinate system rotation
G69	Coordinate system rotation cancel
G73	High speed peck drilling cycle (deep hole)
G74	Left hand threading cycle
G76	Fine boring cycle
G80	Fixed cycle cancel
G81	Drilling cycle
G82	Spot-drilling cycle
G83	Peck-drilling cycle (deep hole drilling cycle)
G84	Right hand threading cycle
G85	Boring cycle
G86	Boring cycle
G87	Back boring cycle
G88	Boring cycle
G89	Boring cycle
G90	Absolute dimensioning mode
G91	Incremental dimensioning mode
G92	Tool position register
G98	Return to initial level in a fixed cycle
G99	Return to R-level in a fixed cycle

## M-Code

Function	Description
M00	Compulsory program stop
M01	Optional program stop
M02	End of program (usually with reset, no rewind)
M03	Spindle rotation normal (CW for R/H tools)
M04	Spindle rotation reverse (CCW for R/H tools)
M05	Spindle stop
M06	Automatic tool change (ATC)
M07	Coolant mist ON <i>(usually an option)</i>
M08	Coolant ON (coolant pump motor ON)
M09	Coolant OFF (coolant pump motor OFF)
M19	Spindle orientation
M30	Program end (always with reset and rewind)
M48	Feedrate override cancel OFF <i>(deactivated)</i>
M49	Feedrate override cancel ON <i>(activated)</i>
M60	Automatic pallet change (APC)
M78	B axis clamp <i>(nonstandard)</i>
M79	B axis unclamp <i>(nonstandard)</i>
M98	Subprogram call
M99	Subprogram end



## Error Codes

Error Code	Message	Description
1	Expected command letter	G-code words consist of a letter and a value. Letter was not found.
2	Bad number format	Missing the expected G-code word value or numeric value format is not valid.
3	Invalid statement	Grbl '\$' system command was not recognized or supported.
4	Value < 0	Negative value received for an expected positive value.
5	Setting disabled	Homing cycle failure. Homing is not enabled via settings.
6	Value < 3 $\mu$ sec	Minimum step pulse time must be greater than 3 $\mu$ sec.
7	EEPROM read fail. Using defaults	An EEPROM read failed. Auto-restoring affected EEPROM to default values.
8	Not idle	Grbl '\$' command cannot be used unless Grbl is IDLE. Ensures smooth operation during a job.
9	G-code lock	G-code commands are locked out during alarm or jog state.
10	Homing not enabled	Soft limits cannot be enabled without homing also enabled.
11	Line overflow	Max characters per line exceeded. Received command line was not executed.
12	Step rate > 30kHz	Grbl '\$' setting value cause the step rate to exceed the maximum supported.
13	Check Door	Safety door detected as opened and door state initiated.
14	Line length exceeded	Build info or startup line exceeded EEPROM line length limit. Line not stored.
15	Travel exceeded	Jog target exceeds machine travel. Jog command has been ignored.
16	Invalid jog command	Jog command has no '=' or contains prohibited g-code.
17	Setting disabled	Laser mode requires PWM output.
20	Unsupported command	Unsupported or invalid g-code command found in block.
21	Modal group violation	More than one g-code command from same modal group found in block.
22	Undefined feed rate	Feed rate has not yet been set or is undefined.
23	Invalid gcode ID:23	G-code command in block requires an integer value.
24	Invalid gcode ID:24	More than one g-code command that requires axis words found in block.
25	Invalid gcode ID:25	Repeated g-code word found in block.
26	Invalid gcode ID:26	No axis words found in block for g-code command or current modal state which requires them.
27	Invalid gcode ID:27	Line number value is invalid.
28	Invalid gcode ID:28	G-code command is missing a required value word.
29	Invalid gcode ID:29	G59.x work coordinate systems are not supported.
30	Invalid gcode ID:30	G53 only allowed with G0 and G1 motion modes.
31	Invalid gcode ID:31	Axis words found in block when no command or current modal state uses them.
32	Invalid gcode ID:32	G2 and G3 arcs require at least one in-plane axis word.
33	Invalid gcode ID:33	Motion command target is invalid.
34	Invalid gcode ID:34	Arc radius value is invalid.
35	Invalid gcode ID:35	G2 and G3 arcs require at least one in-plane offset word.
36	Invalid gcode ID:36	Unused value words found in block.
37	Invalid gcode ID:37	G43.1 dynamic tool length offset is not assigned to configured tool length axis.
38	Invalid gcode ID:38	Tool number greater than max supported value.

## Alarm Codes

Alarm Code	Message	Description
1	Hard limit	Hard limit has been triggered. Machine position is likely lost due to sudden halt. Re-homing is highly recommended.
2	Soft limit	Soft limit alarm. G-code motion target exceeds machine travel. Machine position retained. Alarm may be safely unlocked.
3	Abort during cycle	Reset while in motion. Machine position is likely lost due to sudden halt. Re-homing is highly recommended. May be due to issuing g-code commands that exceed the limit of the machine.
4	Probe fail	Probe fail. Probe is not in the expected initial state before starting probe cycle when G38.2 and G38.3 is not triggered and G38.4 and G38.5 is triggered.
5	Probe fail	Probe fail. Probe did not contact the workpiece within the programmed travel for G38.2 and G38.4.
6	Homing fail	Homing fail. The active homing cycle was reset.
7	Homing fail	Homing fail. Safety door was opened during homing cycle.
8	Homing fail	Homing fail. Pull off travel failed to clear limit switch. Try increasing pull-off setting or check wiring.
9	Homing fail	Homing fail. Could not find limit switch within search distances. Try increasing max travel, decreasing pull-off distance, or check wiring.

# Operation Manual

This is an operating manual for the 3 Axis CNC Milling Router. The CNC Milling Router is intended for instructional purposes at the University of Central Oklahoma. Please use machine only for its intended purposes.



## ⚠ Safety Precautions ⚠

Operating any CNC Machine can be hazardous, and safety precautions must be taken before any machining operation. Here is a list of safety precautions when operating the CNC Milling Router.

1. It is recommended to wear Personal Protective Equipment (PPE), such as safety glasses and closed toed shoes when operating the CNC Milling Router.
2. Never reach into the workspace during operations.
3. Ensure that the cover is closed before starting any machine operation.
4. Do not attempt to clean or remove debris during machining operations, ensure that the machine is stopped when opening the cover.
5. Do not leave the CNC Milling Router unattended during operations.
6. Use only recommended materials for machining.
7. Use only appropriate cutting tools and speeds to prevent injury and damage to machine.
8. Make sure that the workpiece is properly secured and use appropriate clamping tools if necessary.
9. Follow all necessary instructions and safety precautions when operating CNC Milling Router.

## Powering & Connecting CNC Milling Router to UGS

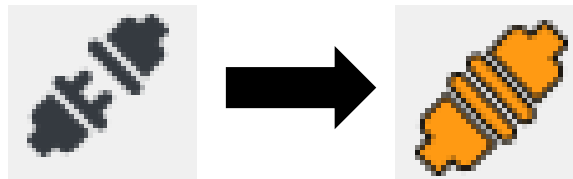
1. Plug CNC Milling Router outlet into 120VAC wall plug. Flip Power switch on right side of machine to “ON” position.
2. Using USB Type A to Type B cable, plug Type A into the computer and Type B into the Arduino on the backside of the machine.



3. Using Universal G-code Sender (UGS), connect to COM port for Arduino at a Baud rate of 115,200. If COM is not shown after plugging in cable, use refresh button.



4. Click on the Connect button in UGS.



5. Now the CNC Milling Router is ready to be used.

# Manually Operating CNC Milling Router using UGS

## 1. Manual Jogging using Jog Controller

The image shows a manual jogging controller interface with the following components:

- Buttons to manually move CNC:** A grid of buttons for X+, Y+, Z+, X-, Y-, Z-, and a central keyboard icon.
- Step Size in mm to move X & Y Axis:** A field labeled "Step size XY" with a value of 20 and a unit selector set to "Millimeters".
- Step Size in mm to move Z Axis:** A field labeled "Step size Z" with a value of 5.
- Not used, for 4 & 5 Axis CNC:** A field labeled "Step size ABC" with a value of 1.
- Feed rate during Manual Jogging:** A field labeled "Feed rate" with a value of 1,500.
- Measurement and Step Size Controls:** Buttons for "Larger" and "Smaller" to adjust step sizes, and a "Millimeters" button to toggle between mm and inches.

## 2. Actions Toolbox

The image shows an Actions Toolbox interface with the following buttons and their functions:

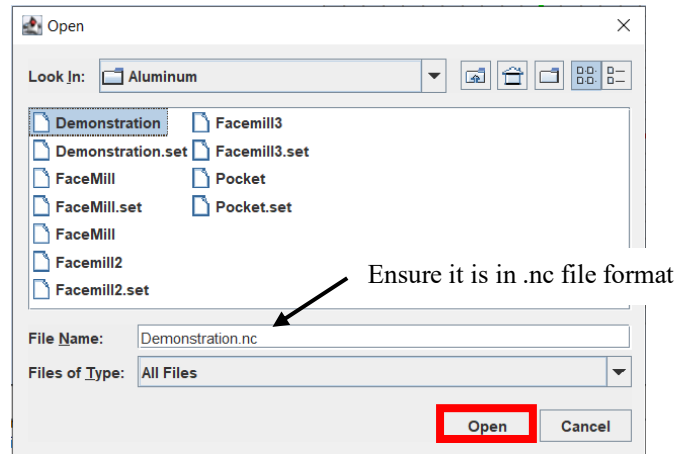
- Reset Zero Position to current position:** A button labeled "Reset Zero" with a zero icon.
- Return CNC to Zero Position:** A button labeled "Return to Zero" with a return icon.
- Resets Arduino after Alarm trigger:** A button labeled "Soft Reset" with a refresh icon.
- Begin Homing Operation:** A button labeled "Home Machine" with a home icon.
- Unlocks CNC:** A button labeled "Unlock" with a lock icon.
- Get current set G-code functions:** A button labeled "Get State" with a magnifying glass icon.
- Get CNC Machine Settings:** A button labeled "Check Mode" with a checkmark icon.

## Uploading G-code Programs

1. To upload a G-code program file, click on the file open icon.



2. In the File Dialog, click on the appropriate .nc file and click open to upload it to UGS.



3. To begin the program machining operation, press the play button.



## Router

The router used is manually turned on/off using the switch at the top, using the dial, the following speeds can be achieved:

Dial	Speed (rpm)
1	10,000
2	14,000
3	18,000
4	23,000
5	27,000
6	32,000

## Visual Studio Code

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.IO.Ports;
using System.IO;

namespace WindowsFormsApp2
{
    public partial class Form1 : Form
    {
        private String SerialDataIn;
        private bool AxisClick = false;
        private bool xclick = false;
        private bool yclick = false;
        private bool zclick = false;
        private bool send2Click = false;
        private bool SendClick = false;

        public Form1()
        {
            InitializeComponent();
        }
    }
}
```

```
}
```

```
private void Form1_Load(object sender, EventArgs e)
```

```
{
```

```
    button_open.Enabled = true;
```

```
    button_close.Enabled = false;
```

```
    button_send.Enabled = false;
```

```
    button_X_plus.Enabled = false;
```

```
    button_X_minus.Enabled = false;
```

```
    button_Y_plus.Enabled = false;
```

```
    button_Y_minus.Enabled = false;
```

```
    button_Z_plus.Enabled = false;
```

```
    button_Z_minus.Enabled = false;
```

```
    button1_Send.Enabled = false;
```

```
    button_axis.Enabled = false;
```

```
    button_clear.Enabled = false;
```

```
    button_Start.Enabled = false;
```

```
    button_Status.Enabled = false;
```

```
    button_Stop.Enabled = false;
```

```
    button_HomingSettings.Enabled = false;
```

```
    button_OverrideAlarm.Enabled = false;
```

```
    button_RunHoming.Enabled = false;
```

```
    button_status2.Enabled = false;
```

```
    button_zero.Enabled = false;
```

```
    button_disable.Enabled = false;
```

```
    button_reset.Enabled = false;
```

```
    textBox1.Enabled = false;
```

```
    textBox2.Enabled = false;
```

```
    textBox3.Enabled = false;
```

```
    textBox4.Enabled = false;
```



```

button_resetZero.Enabled = false;
button_setParameters.Enabled = false;
button_Unlock.Enabled = false;

richTextBox1.Enabled = false;
progressBar1_StatusBar.Value = 1;
label_status.Text = "DISCONNECTED";
label_status.ForeColor = Color.Red;

comboBox1_COMPort.Text = "";
comboBox2_BaudRate.Text = "115200";
string[] portLists = SerialPort.GetPortNames();
comboBox1_COMPort.Items.AddRange(portLists);
button_refresh.Font = new Font("Wingdings 3", 8, FontStyle.Bold);
button_refresh.Text = Char.ConvertFromUtf32(81);
}
private void button_open_Click(object sender, EventArgs e)
{
    try
    {
        serialPort1.PortName = comboBox1_COMPort.Text;
        serialPort1.BaudRate = Convert.ToInt32(comboBox2_BaudRate.Text);
        serialPort1.Open();
        button_open.Enabled = false;
        button_close.Enabled = true;
        button_send.Enabled = true;
        button_X_plus.Enabled = true;
        button_X_minus.Enabled = true;
        button_Y_plus.Enabled = true;
        button_Y_minus.Enabled = true;
    }
}

```

```
button_Z_plus.Enabled = true;
button_Z_minus.Enabled = true;
button1_Send.Enabled = true;
button_axis.Enabled = true;
button_clear.Enabled = true;
button_Start.Enabled = true;
button_Status.Enabled = true;
button_Stop.Enabled = true;
button_HomingSettings.Enabled = true;
button_OverrideAlarm.Enabled = true;
button_RunHoming.Enabled = true;
button_status2.Enabled = true;
button_zero.Enabled = true;
button_disable.Enabled = true;
button_reset.Enabled = true;
textBox1.Enabled = true;
textBox2.Enabled = true;
textBox3.Enabled = true;
textBox4.Enabled = true;
button_resetZero.Enabled = true;
button_setParameters.Enabled = true;
button_Unlock.Enabled = true;
richTextBox1.Enabled = true;
progressBar1_StatusBar.Value = 100;
label_status.Text = "CONNECTED";
label_status.ForeColor = Color.Green;
serialPort1.ReadTimeout = 1000;
serialPort1.WriteTimeout = 1000;
richTextBox1.AppendText("Connection Started\n");
}
```

```

catch (Exception error)
{
    MessageBox.Show(error.Message);
}
}
private void serialPort1_DataReceived(object sender, SerialDataReceivedEventArgs e)
{
    SerialDataIn = serialPort1.ReadExisting();
    this.Invoke(new EventHandler(AxisData));
}
private void AxisData(object sender, EventArgs e)
{
    String DData = SerialDataIn;
    richTextBox1.Text += DData;
}
private void button_close_Click(object sender, EventArgs e)
{
    if (serialPort1.IsOpen)
    {
        try
        {
            serialPort1.Close();
            button_open.Enabled = true;
            button_close.Enabled = false;
            button_send.Enabled = false;
            button_X_plus.Enabled = false;
            button_X_minus.Enabled = false;
            button_Y_plus.Enabled = false;
            button_Y_minus.Enabled = false;
            button_Z_plus.Enabled = false;

```

```

        button_Z_minus.Enabled = false;
        button1_Send.Enabled = false;
        button_axis.Enabled = false;
        button_clear.Enabled = false;

        textBox1.Enabled = false;
        textBox2.Enabled = false;
        button_resetZero.Enabled = false;
        button_setParameters.Enabled = false;
        button_Unlock.Enabled = false;

        richTextBox1.Enabled = false;
        progressBar1_StatusBar.Value = 1;
        label_status.Text = "DISCONNECTED";
        label_status.ForeColor = Color.Red;
        richTextBox1.AppendText("Disconnected\n");
    }

    catch (Exception error)
    {
        MessageBox.Show(error.Message);
    }
}

private void Form1_FormClosing(object sender, FormClosingEventArgs e)
{
    if (serialPort1.IsOpen)
    {
        try

```

```

        {
            serialPort1.Close();

        }
        catch (Exception error)
        {
            MessageBox.Show(error.Message);
        }
    }
}
private void button_send_Click(object sender, EventArgs e)
{
    try
    {
        serialPort1.WriteLine(textBox1.Text);
        textBox1.Clear();
        SendClick = true;
    }
    catch (Exception error)
    {
        MessageBox.Show(error.Message);
    }
}
private void button5_Z_plus_Click(object sender, EventArgs e)
{
    if (textBox2.Text == "")
    {
        MessageBox.Show("Enter Position Value");
    }
    else

```

```

{
    try
    {
        serialPort1.WriteLine("Z" + textBox2.Text);
        zclick = true;
    }
    catch (Exception error)
    {
        MessageBox.Show(error.Message);
    }
}
}

private void button6_Z_minus_Click(object sender, EventArgs e)
{
    if (textBox2.Text == "")
    {
        MessageBox.Show("Enter Position Value");
    }
    else
    {
        try
        {
            serialPort1.WriteLine("Z-" + textBox2.Text);
            zclick = true;
        }
        catch (Exception error)
        {
            MessageBox.Show(error.Message);
        }
    }
}

```

```

    }
}

private void button1_X_plus_Click(object sender, EventArgs e)
{
    if (textBox2.Text == "")
    {
        MessageBox.Show("Enter Position Value");
    }
    else
    {
        try
        {
            serialPort1.WriteLine("X+" + textBox2.Text);
            xclick = true;

        }
        catch (Exception error)
        {
            MessageBox.Show(error.Message);
        }
    }
}

private void button2_X_minus_Click(object sender, EventArgs e)
{
    if (textBox2.Text == "")
    {
        MessageBox.Show("Enter Position Value");
    }
}

```

```

    }
else
{
    try
    {
        serialPort1.WriteLine("X-" + textBox2.Text);
        xclick = true;
    }
    catch (Exception error)
    {
        MessageBox.Show(error.Message);
    }
}
}
private void button3_Y_plus_Click(object sender, EventArgs e)
{
    if (textBox2.Text == "")
    {
        MessageBox.Show("Enter Position Value");
    }
else
{
    try
    {
        serialPort1.WriteLine("Y+" + textBox2.Text);
        yclick = true;
    }
    catch (Exception error)
    {

```



```

        MessageBox.Show(error.Message);
    }
}

private void button4_Y_minus_Click(object sender, EventArgs e)
{
    if (textBox2.Text == "")
    {
        MessageBox.Show("Enter Position Value");

    }
    else
    {
        try
        {
            serialPort1.WriteLine("Y-" + textBox2.Text);
            yclick = true;
        }
        catch (Exception error)
        {
            MessageBox.Show(error.Message);
        }
    }
}

OpenFileDialog ofd = new OpenFileDialog();
private void button1_OpenFile_Click(object sender, EventArgs e)
{

    richTextBox2.Clear();

```

```

ofd.Filter = "TXT|*.txt";
ofd.ShowDialog();
try
{
    textBox3_fileName.Text = ofd.FileName;
    string filePath = ofd.FileName;
    button1_Send.Enabled = true;
    List<string> Lines = File.ReadAllLines(filePath).ToList();
    foreach (string line in Lines)
    {
        if (!line.Contains('('))
        {
            if (!string.IsNullOrEmpty(line))
            {
                richTextBox2.AppendText(line);
                richTextBox2.AppendText("\n");
            }
        }
    }
}
catch (Exception error)
{
}
}
private void button1_Send_Click(object sender, EventArgs e)
{
    textBox3_fileName.Text = ofd.FileName;
    string filePath = ofd.FileName;
    List<string> Lines = File.ReadAllLines(filePath).ToList();
    try

```

```

    {
        foreach (string line in Lines)
        {
            if (!string.IsNullOrEmpty(line))
            {
                serialPort1.WriteLine(line);
                send2Click = true;
            }
        }
    }
    catch (Exception error)
    {
    }
}

private void button1_Click(object sender, EventArgs e)
{
    AxisClick = true;
    serialPort1.WriteLine("?");
}

private void button_clear_Click(object sender, EventArgs e)
{
    richTextBox1.Clear();
}

private void richTextBox1_TextChanged(object sender, EventArgs e)
{
    richTextBox1.SelectionStart = richTextBox1.Text.Length;
    richTextBox1.ScrollToCaret();
}

```

```

}
private void button_Unlock_Click(object sender, EventArgs e)
{
    serialPort1.WriteLine("$");
    serialPort1.WriteLine("$");
}
private void button2_Click(object sender, EventArgs e)
{
    serialPort1.WriteLine("X0 Y0 Z0"); }
private void button1_Click_1(object sender, EventArgs e)
{
    serialPort1.WriteLine("$100=200");
    serialPort1.WriteLine("$101=200");
    serialPort1.WriteLine("$102=100");
    serialPort1.WriteLine("$110=200");
    serialPort1.WriteLine("$111=200");
    serialPort1.WriteLine("$112=100");
}
private void button1_Click_2(object sender, EventArgs e)
{

    serialPort1.WriteLine("$24="+textBox3.Text);
    serialPort1.WriteLine("$25="+textBox4.Text);
    serialPort1.WriteLine("$26=50");
    serialPort1.WriteLine("$27=2");
    serialPort1.WriteLine("$22=1");
    progressBar_homing.Value = 100;
    label_homing.Text = "Enabled";
    label_homing.ForeColor = Color.Green;
}

```

```

private void button3_Click_1(object sender, EventArgs e)
{
    serialPort1.WriteLine("$22=0");

    progressBar_homing.Value = 0;
    label_homing.Text = "Disabled";
    label_homing.ForeColor = Color.Red;
}
private void button2_Click_1(object sender, EventArgs e)
{
    serialPort1.WriteLine("$X");
}
private void button3_Click(object sender, EventArgs e)
{
    serialPort1.WriteLine("$H");
}
private void button5_Click(object sender, EventArgs e)
{
    serialPort1.WriteLine("!");
}
private void button4_Click(object sender, EventArgs e)
{
    serialPort1.WriteLine("~");
}
private void button_Status_Click(object sender, EventArgs e)
{
    serialPort1.WriteLine("?");
}
private void button1_Click_3(object sender, EventArgs e)
{

```

```

        serialPort1.WriteLine((char)24 + "x");
    }
    private void button2_Click_2(object sender, EventArgs e)
    {
        serialPort1.WriteLine("$$");
    }

    private void button_refresh_Click(object sender, EventArgs e)
    {
        comboBox1_COMPort.Items.Clear();
        string[] portLists = SerialPort.GetPortNames();
        comboBox1_COMPort.Items.AddRange(portLists);
    }
    private void button_zero_Click(object sender, EventArgs e)
    {
        serialPort1.WriteLine("G92 X0 Y0 Z0");
    }
}
}

```