CNC machine based complete PCB assembly line (milling)

Submitted in partial fulfilment of the requirements
of the degree of

Bachelors of Engineering

by

Samir Alam Roll No. 02
Arvind Kushwaha Roll No. 53
Pronoy Mandal Roll No.56
Abin Nair Roll No. 59

Supervisor:
Dr. (Mrs.) Asawari Dudwadkar

Department of Electronics
V.E.S. Institute of Technology/University of Mumbai
2019 - 20

CERTIFICATE

This is to certify that the project entitled "**CNC machine based complete PCB assembly line (milling)**" is a bonafide work of **Samir Alam (Roll No. 2)**, **Arvind Kushwaha (Roll No. 53)**, **Pronoy Mandal (Roll No. 56)** and **Abin Nair (Roll No. 59)**, submitted to the V.E.S. Institute of Technology in partial fulfillment of the requirement for the award of the **Bachelor of Engineering in Electronics**.

Supervisor/Guide
Dr. (Mrs.) Asawari Dudwadkar

Head of Department
Mrs. Kavita Tiwari

Principal
Dr. (Mrs.) J. M. Nair

Project Report Approval for B. E.

This project report entitled CNC machine based complete ***PCB assembly line (milling)*** by ***Samir Alam***, ***Arvind Kushwaha***, ***Pronoy Mandal*** and ***Abin Nair*** is approved for the degree of ***Bachelor of Engineering in Electronics***.

Examiners

1. _____

2. _____

Date:

Place:

DECLARATION

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Samir Alam (02)	Arvind Kushwaha (53)
Pronoy Mandal (56)	Abin Nair (59)

Date:

Abstract

A CNC based PCB Milling Machine incorporates the plan and usage of a CNC (Computer Numerical Control) machine to make a PCB (Printed Circuit Board) in a particular set up. Creation of a PCB within the present market is extremely costly when the manufacture of testing circuits is taken into account. Along these lines, this CNC based PCB machine would be an efficient instrument through which circuits are often scratched and penetrated (or engraved) at an inexpensive rate. The creation of this CNC machine is to diminish the cost and multifaceted nature of machine. This venture manages the plan of a programmed scaled down CNC machine for PCB drawing and processing. A PCB mill is a gadget that engraves out details from a Gerber file on a copper clad board with the end goal that it makes a Printed Circuit Board (PCB). PCBs are utilized in the field of electrical design to interface electrical parts to each other.

In CNC machine the program is present or loaded in the memory of a computer system. The software engineer can easily compose the codes and alter the projects according to their requirements. These projects can be utilized for various parts and they don't need to be rehashed once more. The CNC machine offers more prominent adaptability and computational ability. New frameworks and components can be joined into the CNC controller essentially by reconstructing the unit.

The CNC machine involves a PC or a computation device in which the program is present for cutting of the metal from the job according to the given specifications. All the slicing forms that are to be completed and all last measurements are encouraged into the PC through a program (essentially G-code). The PC realizes what precisely is to be done and controls all the cutting procedures. CNC machine works like a robot, which must be bolstered with the program that it is supposed to take after every one of your directions. A subset of the normal machine devices that can keep running on the CNC are: drilling machine or milling machine.

The principle motivation behind these machines is to expel a portion of the metal to give it appropriate shape, for example, round, rectangular and so on, the software guides the machine apparatus to perform different machining operations according to the program of guidelines given by the user, all the CNC machines are intended to meet close correctness.

The CNCs have tremendous advancement potential and many machining procedures can be communicated by methods for basic operations, for example interpretations, revolutions and by controlling the status of the CNC head. Numerous CNCs are controlled utilizing a unique programming dialect, named G-Code.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Project name and title

The name of our project is CNC machine based complete PCB assembly line (milling).

1.2 Problem statement

Current factory-based PCB manufacturing targets large clients primarily for mass production. However, when it comes down to normal students, hobbyists, engineers mass production of PCBs is not what they expect. These small scale customers want fast but at the same time reliable and quality output for their products.

To solve the same this project attempts to create a CNC based PCB milling machine primarily for such small scale customers. Standard home-based PCB manufacturing techniques waste a considerable amount of time in the revelation of copper tracks on the PCB by sequential steps including (but not restricted to) UV exposure, etching, engraving, cleaning etc. till all tracks on a typical copper-clad are fully revealed. Although for PCBs of constant dimensions (which small scale clients usually prefer) each of the aforementioned steps takes on an average of 3 to 3.5 minutes.

Hence, the total time required to develop a full PCB comes out to be 4 to 6 hours and may extend into multiple days if the concerned machines for the above processes are working under heavy load from other clients. This leads to a complete product delivery time over 1-2 days (per PCB).

1.3 Proposed solution

It turns out that the processing time can be greatly reduced if we manage to target and reduce the time taken in those steps which are consuming a constantly large amount of time for every PCB irrespective of the design on it.

Here the word design although a layman term refers to many things at once.

(a) The total number of tracks on the PCB

(b) The complexity in which the tracks are arranged amongst themselves

(c) The types of tracks (in terms of drill depth, length, thickness, clearance etc.)

The time taken for the UV exposure process, etching process etc. are invariant of the design (word used in the same context as above) configuration of the PCB in terms of (a), (b) and (c).

A solution would be viable if we manage to directly engrave the tracks on a PCB as soon as the schematic file (.sch) or the board file (.brd) for a particular design is generated.In standard factory settings, if the above step is completed the PCB can proceed directly for drilling and component placement.However, the scope of our project is restricted to the successful completion of the former step in minimum time.

1.4 Scope and report contents

Chapter 2 starts with a critical analysis of the previous works that have been done using CNC machines including (but not restricted to) their applications in the PCB manufacturing industry. It goes on to discuss how every unit of the manufacturing process is done by optimal usage of CNC based processes. Added to that it also discusses few papers, journals as well as some commercial CNC instruction manuals.

Chapters 3 to 5 give a very detailed explanation of the current investigation. Chapter 3 deals with the manufacturing and construction of the CNC machine. Chapter 4 mostly deals with the main electronics and driving circuitry used in this project. While Chapter 5 details the reader on the software methodology and concludes the explanation of the current investigation. All the required steps are given sequentially in the order in which were carried out.

The final two chapters i.e. 6 and 7 conclude with the findings of the current investigation, the logical results and analysis associated with it as well as give an insight into the future work plan of the project.

Chapter 2

Literature review

2.1 Review paper 1 - Prototype CNC machine design

2.1.1 CNC Module introduction

The Computer Numeric Control (CNC) [1] is a technology which aims to generate, parse and execute sequential action. It helps for the development of small and big sized CNC machines, based on a system with the capability of communication through a communication bus. It is also required electronic devices to run a CNC machine like controller circuit, as well, a software developed in LabVIEW to establish the communication between the machine and the computer. The objective of this work is the development of a machine which allows future use for PCB track designing with better efficiency than traditional methods.A CNC prototype machine was designed, with three-axis movement, with 300 mm of length both X and Y axes and 10 mm of the length Z-axis.

CNC Machines are often used in metal machining, like drilling, milling etc. This kind of machine consists usually of a servo mechanism controlled by a computer, a high-speed spindle and dedicated tools. This servo mechanism can be realized in a closed-loop fashion, using servo controllers driving DC or synchronous AC machines, or open-loop fashion, using stepper motors. small loads required an open-loop system for a better fit. Therefore, industrial-sized machines do not, in general, use open-loop machines.

The machine is based on the removal of the material of a workpiece. Nowadays, there are machines with six or more axes, which allow the machining of pieces which can handle high levels of complexity that is feasible by all other means.

2.1.2 Use of CNC machine

CNC machines can be used in many different ways, including (but not restricted to):

(a) Making circuits boards - They are used to make printed circuit boards for different products

(b) Making enclosures - The different types of CNC machines like laser cutters, mills and lathes use some parts to make enclosures. Sticker cutters are also used to make decorations for enclosures.

(c) For Computer Integrated Manufacturing - They are programmed in a specific way according to instructions, and this can save hundreds of thousands of hours of manual programming.

The CNC machine is used in different industries, including:

(a) Agriculture - The agriculture industry requires reliable, high-quality production of machine parts and other products.

(b) Transportation - Some vehicle parts like gears, brakes, pins and shafts need precise CNC machining in their manufacturing.

(c) Firearms - CNC machines are needed to create complex barrels, plates and triggers, among other things.

2.2 Review paper 2 - Automatic mini CNC machine for drawing and drilling

2.2.1 Methodology

[2] The G code is interfaced with an Arduino CNC based controller by software which is used to convert into equivalent controller instructions. Hence it acts as an interfacing module between PC to Controller. This code is further passed to the stepper motor by easy drivers which convert the code and as per instructions the stepper motor moves. We need three axes X, Y, Z which operates as follows X stepper motor moves left and right Y stepper motor moves front and back and Z stepper motor up and down as per given dimensions these axes will move on.

2.2.2 G codes and M codes

PC G code is a converted instruction form of programming language which defines instructions on where to move, how fast to move with the path.

2.2.3 CNC shield

- GRBL 0.8c compatible. (Open supply microcode that runs on associate degree Arduino UNO that turns G-code commands into stepper signal.)
- 4-Axis support (X, Y, Z, A which can duplicate X, Y, Z or do a full 4th axis with custom firmware using pins D12 and D13)
- 2 x finish stops for every axis (6 in total)

- Spindle enable and direction
- Coolant enable
- Uses removable Pololu A4988 compatible stepper drivers. (A4988, DRV8825 and others)
- Jumpers to line the Micro stepping for the stepper drivers. (Some drivers just like the DRV8825 will do up to 1/32 micro-stepping)
- Compact design
- Stepper Motors may be connected with four pin Molex connectors.
- Runs on 12-36V DC. (At the instant solely the Pololu DRV8825 drivers will handle up to 36V thus please take into account the operation voltage once powering the board.

2.2.4 Generic block diagram

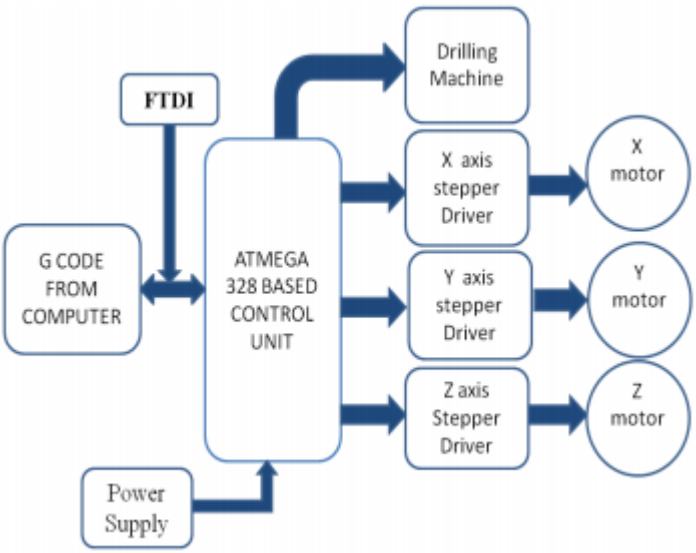


Figure 2.1: Block diagram of the CNC machine developed in the paper

2.2.5 Stepper motor and associated parts

NEMA 17 stepper motor

Stepper motors are sort of DC motors that move in increments or steps, they move at a legendary interval for every pulse of power. These pulses of power are provided by a motor driver and are referred to as a step. As every step moves a legendary distance it makes them handy devices for repeatable positioning. Stepper motors are usually used on CNC Machines such as 3D Printers, Laser Cutters, CNC Routers.

2.2.6 Micro controller requirements

Micro controller - Arduino UNO r3

The Arduino UNO R3 could be a microcontroller board supporting the ATmega328. It has fourteen digital input/output pins (of that vi may be used as PWM outputs), 6 analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything required to support the microcontroller; merely connect it to a PC with a USB cable or power it with an AC-to-DC adapter or battery to induce power.

The Uno differs from all preceding boards in that it doesn't use the FTDI USB-to-serial driver chip. Instead, it options the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter. Revision a pair of the Uno board features a resistance propulsion the 8U2 HWB line to ground, making it easier to put into DFU.

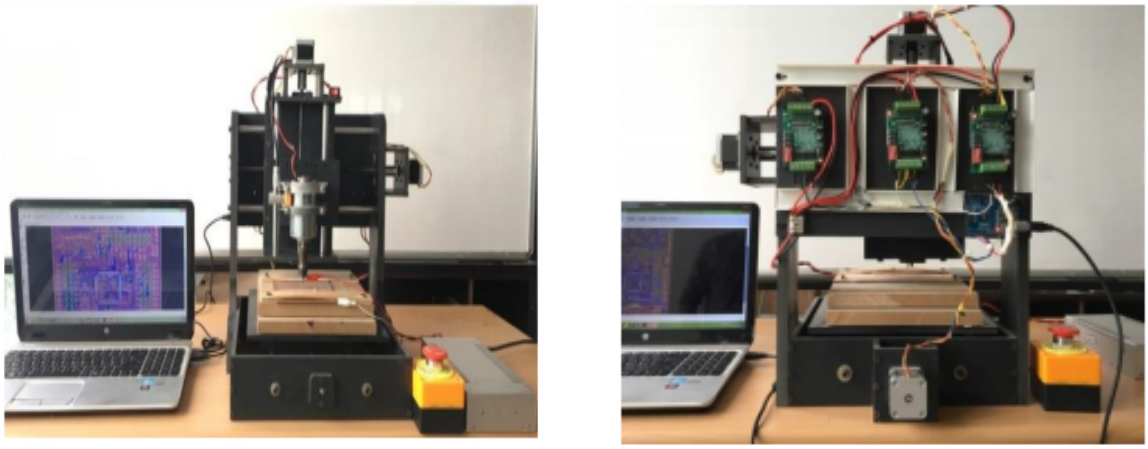
Revision three of the board has the subsequent new features:

- Added SDA and SCL pins that are near to the AREF pin and 2 other new pins placed near the RESET pin, the IOREF that allow the shields to adapt to the voltage provided from the board. In future, shields are compatible with the board that uses the AVR, which operates with 5V and with the Arduino Due that operate with 3.3V. The other could be a not connected pin, that is reserved for future functions.
- Stronger RESET circuit.
- Atmega 16U2 replaced the 8U2.

2.3 Review paper 3 - CNC fabrication and user manual

This reference [3] instead of being a paper happens to be a detailed user manual regarding how one should use a typical CNC machine for PCB prototyping purposes. It starts on similar lines of needs and requirements as the CNC machine being developed by the authors of this report. It then goes on to explain various software aspects of PCB design followed by which it describes what are the standard and special procedures used in CNC machines such as moving the tooltip to the origin/soft - home position, calibration of various important parts to name a few.

A BYO PCB prototyping machine capable of developing both through hole and Surface Mount Technology on single sided and double sided PCB boards has been used as a standard reference throughout the manual. Following is a detailed description of only the important and relevant sections of the concerned manual.



(a) Mechanical Setup of the Machine (b) Electronics hardware assembly

Figure 2.2: The PCB prototyping machine as described in the reference manual

2.3.1 Layers of a PCB copper clad

Understanding PCB composition in hindsight is simply understanding how layers of a PCB are stacked over one other and what is the significance of each layer. The most important layer(s) of the PCB copper cladding is the copper layers themselves. These are the layers through which the intended electrical signals pass depending on the application. The copper layers sandwich a layer which is usually the thickest layer in a copper cladding. It is an insulating layer called the substrate. Depending on the application the substrate layer could be made of various materials of various thicknesses. The most common ones are FR4 which is a NEMA grade designation for glass-reinforced epoxy laminate material and FR2 which is a phenolic cotton paper. FR2 is very common in single-sided PCBs in consumer electronics while FR4 is universal in multi-layer PCBs and in general.

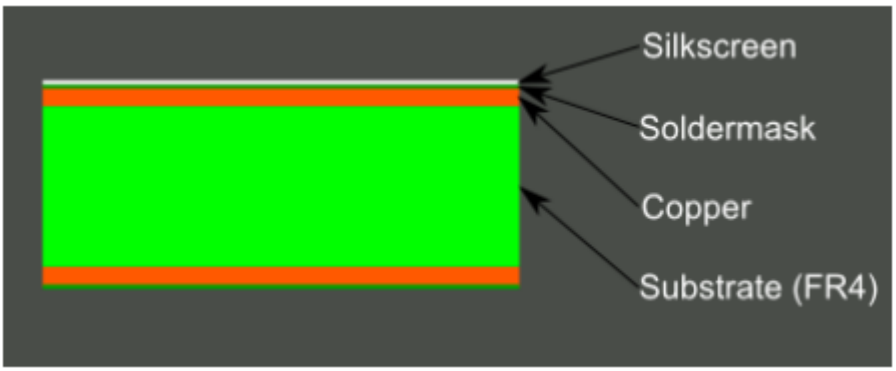


Figure 2.3: The layers of a double-sided PCB

Now to insulate and protect the copper layers they are covered by a thin lacquer-like material called the solder mask. This layer is responsible for the characteristic "green" colour of PCBs or whatever colour (red, blue etc.) they are supposed to have. Then the final set of top and bottom layers of a PCB is the silkscreen which has a look and feels similar to ink and can be used to add texts and logos to a PCB. Following is an illustration of the various layers of a double-sided PCB copper cladding.

The PCB designer has access to all the layers at any given instant of time during

the development of a particular PCB. For the designer using the EAGLE software, a palette of colours (as shown below) is used to represent the various layers of a PCB. They can be toggled according to the convenience of the designer by going into the layer settings of the board and clicking the layer number to toggle it in view.

2.3.2 CAM processing

CAD/CAM processing belongs to a much larger set/group of processes used in the industry collectively called machining. **Machining** is any of the varied processes during which a bit of staple is dug into a desired final shape and size by a controlled material removal process. The common theme of machining remains as controlled material removal which could be either additive or subtractive. As the words themselves suggest additive manufacturing shall add material to the original raw material while subtractive manufacturing refers to the opposite.

Nowadays, all machining has been converted into OR is exclusively carried out by Computer Numerical Control (CNC). Computer-Aided Design (CAD) and Computer-Aided Manufacturing (CAM) fall under this much larger umbrella of "CNC jobs". This paper explains how to use the CAM processor present in the EAGLE software to generate a CNC job corresponding to a given board file. The same CAM processor can be used to generate "Gerber" files for all the layer(s) of a given PCB as well as "Excellon" files for drilling purposes of the concerned PCB. These types of files come under the much larger umbrella of PCB NC formats which manage to convey routing and drilling information to the concerned CNC machine.

Later the Gerber files generated are sent to "FlatCAM" which is a CAM processing software responsible for generating CNC jobs from Gerber files obtained from any PCB CAD program. It does the additional job of generating corresponding "g codes" for routing purposes and allowing the user to edit the same.

2.3.3 Special procedures

Several challenges arise when designing complex PCB designs; these challenges could range from manufacturing multilayered boards to working on uneven surfaces. The associated software for a given CNC machine and some simple procedures help the users overcome these challenges. Few of those which were discussed in this paper are described below.

Auto levelling

It is not uncommon to have inconsistent traces in larger PCBs while using a CNC based milling machine. Such inconsistencies arise due to minor height variations of the surface on which the PCB is kept ("wasteboard") OR the PCB by itself could be bent or warped. It should be noted that even minor height variations of about 1 mm could change the groove width by about 0.672 mm. To overcome this problem we need specialised software called auto levelling software. By definition, Auto-levelling is a process that tries to **compensate** for height differences instead of **trying to prevent them.**

Auto levelling software depending on their sophistication may approach this problem in different ways and use different forms of sensory data to solve the problem. The most common way to approach this problem is by dividing the entire PCB into uniform geometrical grids. This gives specific sampling positions which could be probed to get an overall idea of the non-uniformities of the surface. E.g. a 40 x 40 mm surface could be divided into 100 grids of 4 x 4 mm and the surface at the centre of the grids is always probed. Probing the height of a PCB locally (i.e. in every grid) is typically done by slowly lowering an end mill until an **electric connection is made between the end mill and board's copper layer**. All these things are usually automated by the software. However, if it is not the case the designer has to design compensation algorithms on their own.

First, all the collected data is accumulated in a single file called the probe file. The generated G code and this file are superimposed and fed to the software. The software now tries to fit a *quadratic surface* (say of the form $f(x^2, y^2)$) on the probe data and accordingly adjusts the G code in the vertical direction. This adjusted G code can be fed into the CNC machine like any other G code file thereby eliminating any problems that may result from an uneven or warped surface.

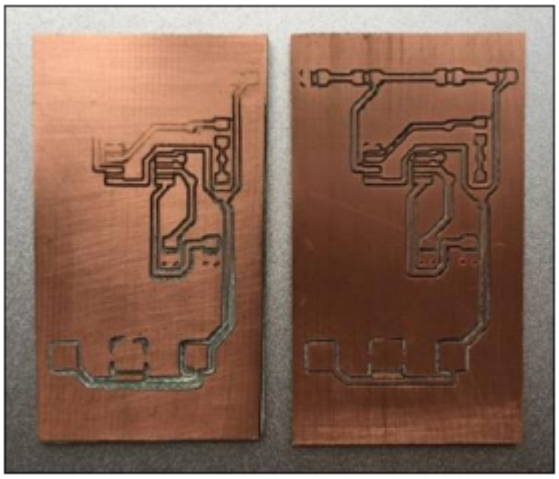
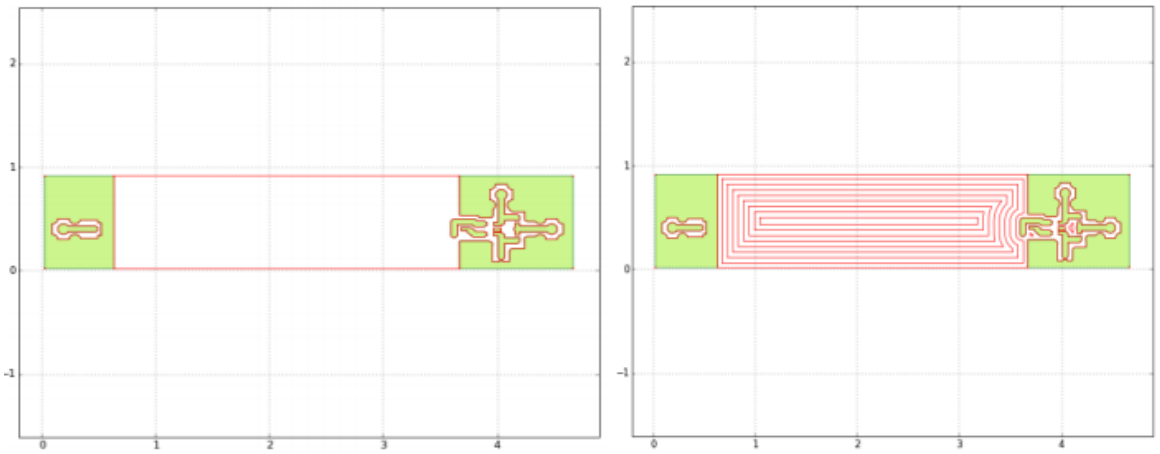


Figure 2.4: Showing the making of PCB board with and without the use of Autoleveller on a skewed platform

Copper area clear

With RF circuit PCBs or any PCB, in general, comes the problem of clearing large areas of unwanted copper traces which is just unwanted core load. The manual explains how to use the various properties available under the "Gerber object" tab to eliminate unwanted copper traces. After specifying the margin dimensions, a boundary (called the "bounding box") is created. A geometric subtraction of the Gerber object from the boundary will give rise to another Gerber object which doesn't have areas with copper on them. Several polygons constitute this newly formed Gerber object. The user gets to choose which Gerber object to keep and what not to. To do so the correct tool dimensions with its specifications are selected and the concerned/desired polygon is painted. "Painting" refers to covering the whole polygon with the tool paths (whose specifications were selected

within the previous step). In this manner, all unwanted copper load can be removed from any type of PCB.



(a) Geometry object with polygons covering the areas without copper

(b) New Geometry Object with the desired tool paths that can overlap too.

Figure 2.5: The software steps during the procedure of *copper clear*

Board cutout

PCBs need to be cut out from much larger and blank copper claddings, otherwise, we shall incur large material wastage in terms of area and cost. The FlatCAM software allows the user to do so by selecting the board cutout tab after choosing a Gerber file. The desired margin (dimension) is selected which tells how far the cutout boundary is from any element of the Gerber source of the PCB. After this "gaps" can be selected for the cutout in terms of their dimensions and their number. It is always recommended to have gap size twice the diameter of the tooltip. FlatCAM allows two gaps (either on top and bottom OR left and right) or four gaps for a given PCB. Once, this step is complete, just like the previous procedure the geometry for the tool path is generated. A CNC job is created from this geometry just like any other geometry and now the job is ready to be processed. Hence, this feature allows precise cutting of a PCB from its larger blank cladding once the main CNC job is done.

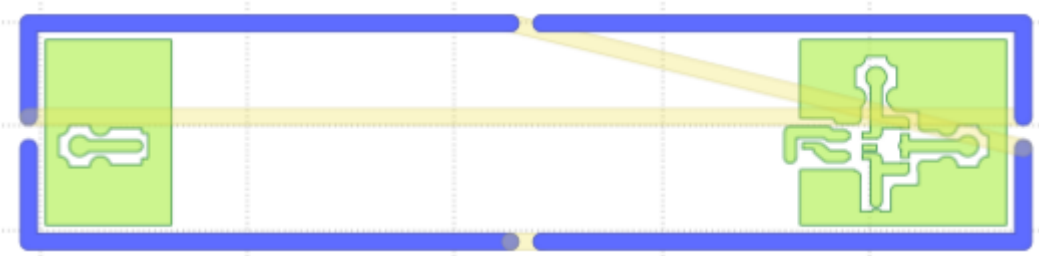


Figure 2.6: Typical CNC job corresponding to a *board cutout*

2.3.4 Computing resolution and accuracy

A lot of experimentation needs to be carried out to verify the practicality of the CNC machine. A typical experimentation procedure needs to be carried out to find out the wire width and resolution capabilities of the PCB. The next part is computing the resolution for SMT fabrication purposes. Both the techniques have been briefly discussed in the manual.

Computing wire width and resolution

To find out the level of resolution while tracing out or engraving wires, a good way is to make a very simple PCB consisting of wires of various widths, to be precise, in decreasing order. The wire of largest possible width (available in the schematic design software) is made followed by the next largest and so on until the wire of minimum possible width possible in the schematic design software is designed. Now the G codes corresponding to this sample PCB are sent to the CNC machine before milling the same.

After the PCB tracks have been engraved on to the copper cladding by the CNC machine, the absolute deviation between the width of the actual engraved tracks and input width for the concerned track provided as input from the software is computed. (Insert equation) This denotes the resolution of the CNC machine. It should be noted that it is not compulsory to compute the absolute deviation, positive and negative deviations are both valid. To carry out this measurement, a handheld microscope is a suitable instrument. For the CNC machine in question, the wire of the smallest width that could be fabricated was found out to be 0.2 mm with an allowable deviation of -0.012 mm.

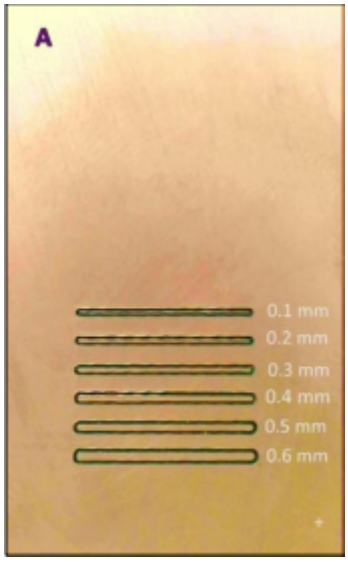


Figure 2.7: PCB fabricated with sample wires of varying widths

Resolution of SMT

Various standard and commonly used SMT packages were fabricated by the CNC machine. It was found out that the minimum pitch size of an SMT that could be fabricated was

about as low as 0.3 mm. A handheld USB microscope was used for measurements in each case.

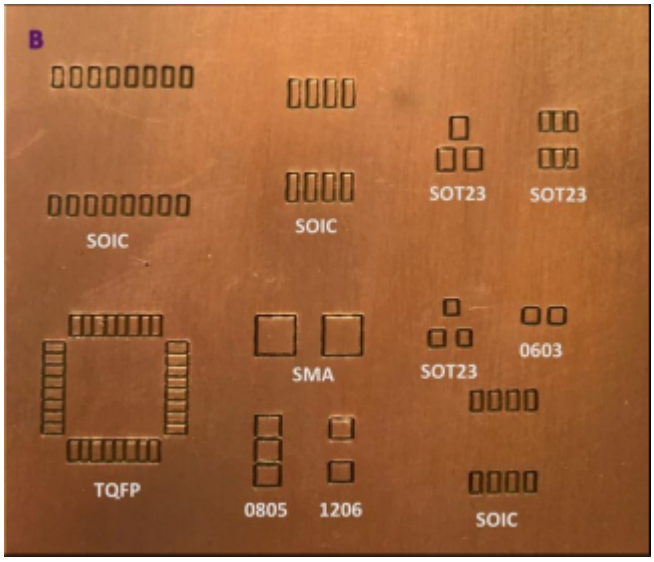


Figure 2.8: PCB fabricated with SMTs of varying footprints

Chapter 3

Construction

3.1 Base material

To accurately describe the expected dimensions of this base material it is important to know the target PCB size for the concerned CNC machine since along with the required machinery and hardware that would be mounted on the base material, the PCB is the main working component. For this project the designers have targeted **15 cm × 15 cm single sided copper clad PCBs**.

The net dimensions of the base material would need to be greater than that of the target PCB size. After performing multiple calculations as described in the following sections. The final base material size has been fixed at **40 cm × 40 cm** .

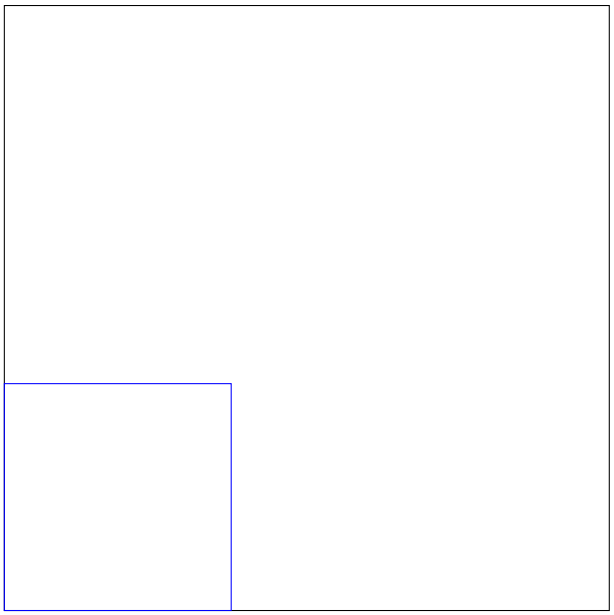


Figure 3.1: Drawing showing maximum PCB dimensions that can be manufactured (in blue) superimposed on base material dimensions represented on a 1 : 5 scale

3.2 XY axes made up of normal unpolished wood

The two segments of wood which must have the primary load bearing capacity are the XY axes. The X segment consists of a wooden plank like structure with slots of length 15 mm and depth 8 mm each to insert wooden plates in it. A similar structure exists for Y - axis segment as well. Dimensions for both the segments are illustrated below.

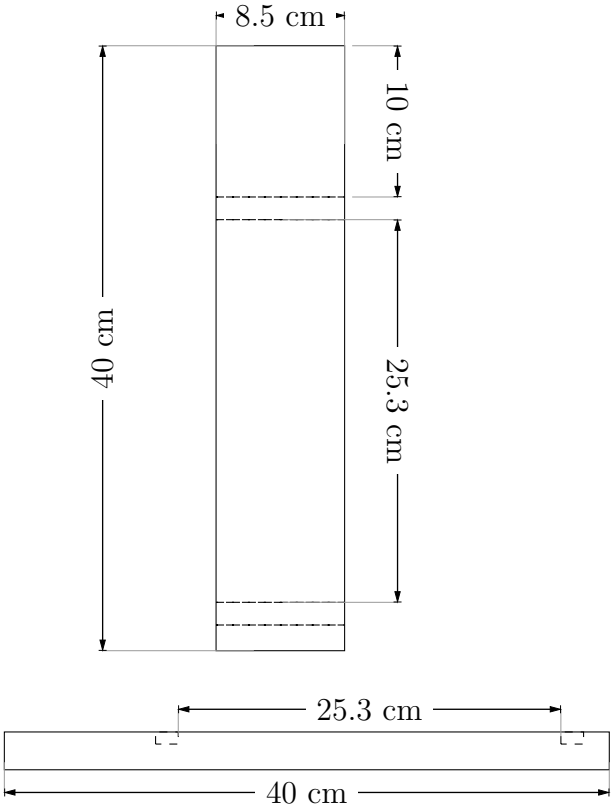


Figure 3.2: (From top to bottom) Top view and front view of the X-axis segment with dimensions represented on a 1 : 5 scale

Following page shows the structure of the Y - axis segment which has a similar scheme of construction but with slightly reduced dimensions.

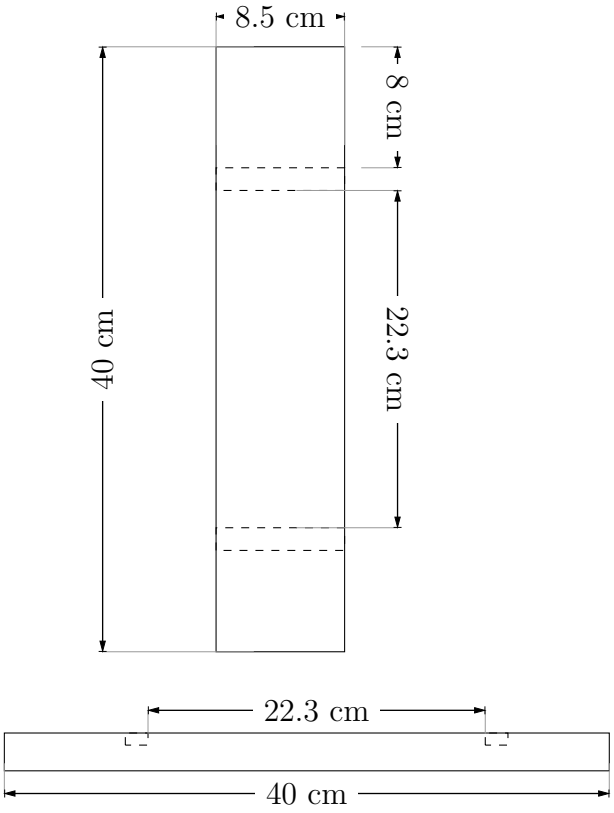
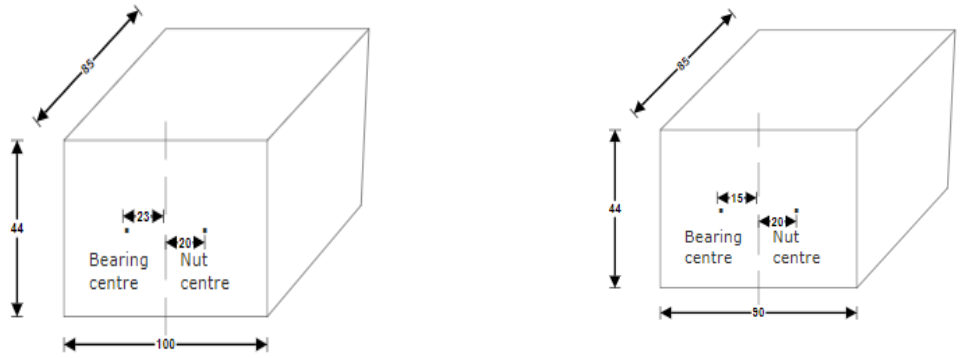


Figure 3.3: (From top to bottom) Top view and front view of the Y-axis segment with dimensions represented on a 1 : 5 scale

The motor resides behind one block (say B_1) while the block on the other end is idle (say B_2). The face of B_1 which faces B_2 has two holes drilled in it. The guide rod connected to the motor shaft is inserted into the larger hole while the support rod is inserted into the smaller hole. The respective dimensions of the hole are clearly mentioned in the diagram and are identical for both B_1 and B_2.

Additionally as mentioned before dimension buffers are always necessary. The buffer which has been used here (both for B_1 and B_2) is: the center of the smaller hole is located at a distance of 1.5 cm from the left end while the center of the larger hole is 3 cm from the right end. The overall buffer which was available at the end for the two blocks were (Left buffer − radius of smaller hole) + (Right buffer − radius of larger hole) $= (1.5 - 0.6) + (3 - 1.65) = 0.9 + 1.35 = 2.25$ cm

The block B_2 as mentioned earlier is idle - the face of the block which does not face B_1 is fully empty. And the sole purpose of B_2 is to stabilise other ends of the main guide rod as well as the support rod. This is accomplished by the bearing arrangements made on the face facing the concerned face of B_1 as mentioned earlier. Details of the bearing design and the means through which they become compatible with the wooden block faces and the concerned guide or support rods are mentioned in the subsequent sections.



(a) Block B_1 (b) Block B_2

Figure 3.4: Three - dimensional representation of wooden blocks B_1 and B_2 with all the relevant dimensions and annotations (not to scale)

3.3 Guide rod and complementary support rods

A pair of guide rod and complementary support rods are required for the XY axes. Before proceeding further it should be noted that the guide rod is threaded while the support rod is smooth. The threaded rod is made up of mild steel while the smooth rod material is stainless steel. As per design specifications, these rods must run perfectly parallel to each other. Ensuring the same is a critical design challenge.

The length of the threaded rod for the X- axis segment is 30 cm while for the Y-axis it comes out to be 27.5 cm. Further, there is no proper specification for the smooth rod lengths rather a larger sized ready made rod is cut to a length so that it fits between the blocks B_1 and B_2 (with all other dimensions taken into consideration).

However, although the rod (threaded) lengths are non identical, the thread pitch, the hole (to be drilled on the motor shaft end) specifications and the diameter are identical. The thread pitch was decided at 1.25 mm. Since the rod on both ends will be held by smooth bearings, some part of the threading on both ends need to be removed. From the shaft side the threading has been deprecated by 3.3 cm while on the other end the threading has been reduced by 1.3 cm. The original diameter of the threaded rod was 1.2 cm however due to the thread deprivation on both sides the diameter has reduced to 1.0 cm on both ends for the concerned amount of thread deprived lengths. For the hole on the shaft side: the drill depth must be exactly equal to the shaft length with zero buffer. The shaft length is 2.3 cm and hence it is the drill depth as well. The diameter of the drill hole is fixed at 5 mm. Another hole must be drilled perpendicular to the rod axis at a distance of 1 cm from the shaft end. A fastening screw shall pass through this hole intersecting the motor shaft axis at a right angle.

Figure 3.5: The guide rod and complementary support rods pictured integrated into an axis segment

3.4 Wooden plates for the XY axes

It should be noted from the previous sections and images that the rods for both the axis segments need to be suspended mid air. To do so, we need separate wooden plates to be attached on both the sides of both the wooden axis segments. The wooden plate (depending on which axis it is being made for) has two holes of appropriate dimensions drilled into it at a specific distance. One of the holes is always larger to support a metallic bearing (more in the next section) which has the role of reducing friction on the threaded rod as they rotate. These wooden plates are sufficiently larger as compared to the wooden blocks. An illustration of one constructed for the project is shown below.

Figure 3.6: A wooden plate for one of the axes with drill holes made for insertion of threaded rod (right) and guide rod (left) visible clearly

Following are the images of the two wooden axes after they have been completely integrated

Figure 3.7: The X - axis segment integrated with other components as per the specifications mentioned in the previous sections

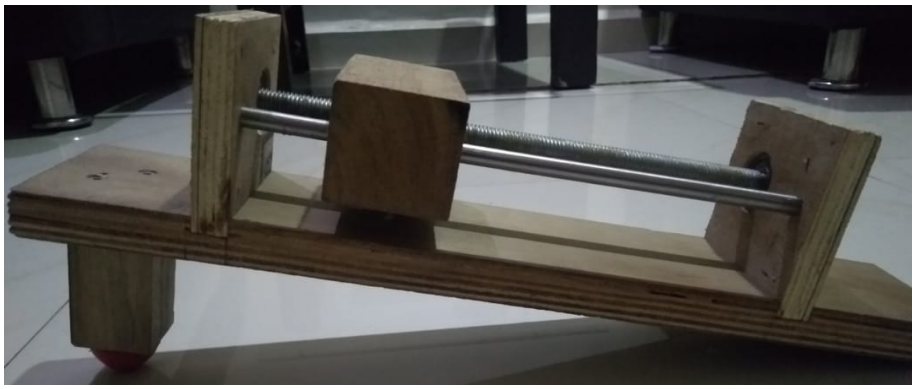


Figure 3.8: The Y - axis segment integrated with other components as per the specifications mentioned in the previous sections

3.5 Bearing design

As stated in the previous section it is obvious that a total of four bearings are needed: one on both ends of threaded rods of both the axes. Metal bearings in the manufacturing industry are specified by their inner diameters. Small spherical balls line up the space between the inner diameter and the outer diameters. These spherical balls reduce friction while the threaded rod rotates under the actuating action of the motor. These spherical balls are the actual bearings where the name of this machine part has come from.

In our project the inner diameter of the bearing is 1.0 cm (matching with the thread deprived rod diameter as mentioned in the previous section) while the outer diameter of the same comes out to be 3.3 cm.

Figure 3.9: Metal bearings similar to the ones used in the project

3.6 Roller wheel section

One end of the Y-axis (Wooden structure) is rested on top of the X-axis, while the other end is freely hanging. This might cause the Y-axis to lean towards that end. To make sure that the axis is stable, we have used a "ball castor wheel" which is attached to the free end of the Y-axis. This helps the axis to be perpendicularly aligned with X-axis and aligned in parallel with the base. To attach the wheel to the Y-axis we have made a wooden block which is of sufficient height to align the axis parallel to the base.

Figure 3.10: A set of castor wheels with a rubber endings similar to the one used in the project

3.7 Engraving bits' selection

The most important part of the CNC machine is the tooltip and its material constituents. The tooltip in case of our CNC machine is an engraving bit made of Tungsten Carbide. In general, such bits are expensive for either engraving or drilling applications. However, they are quite invaluable and universal for almost all CNC based applications. For a CNC based PCB milling machine, a lot of attention is paid on the dimensions of the engraving bit because it directly corresponds to the accuracy of the system. For PCB milling purposes the minimum dimensions of the engraving bit that was purchased were 0.7 mm while the largest measured at about 3.2 mm. Following is an illustration of an

engraving bit with all the relevant dimensions that would help the designer choose an appropriate size for a given set of specifications.

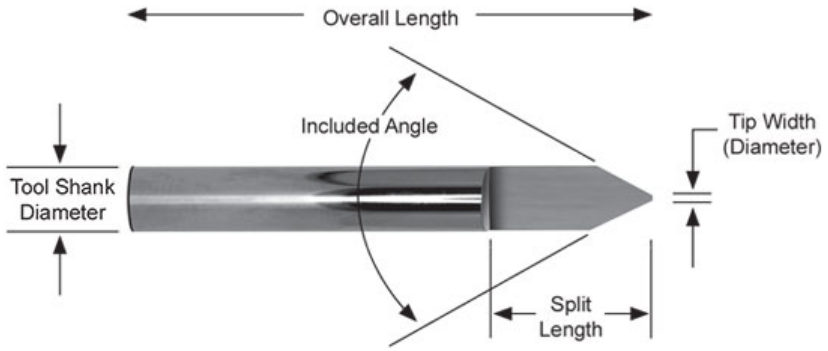


Figure 3.11: An illustration of a Tungsten Carbide engraving bit with all the relevant dimensions

3.8 Coupling and fastening screws

These two types of screws are required in abundance throughout the mechanical assembly for either fixing components to their respective base(s) of different materials OR to join/fix simultaneously actuated (usually) components. Now the primary parameter deciding screw specifications is material thickness on which the screw shall impinge.

For fastening screws impinging on the base or the plank material 1.5 mm screws are sufficient. For coupling screws, used only in one application: secure coupling of threaded rod and motor shaft, a specification of 3 mm is sufficient.

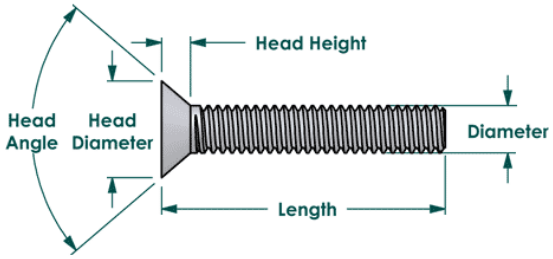


Figure 3.12: A coupling screw with all the relevant dimensions and similar to the ones used in the project

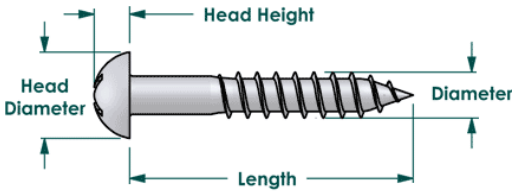


Figure 3.13: A fastening screw with all the relevant dimensions and similar to the ones used in the project

Chapter 4

Electronic and driving circuitry

4.1 Power supply

In our project, there are three major electronic components which have different power ratings. We have to power an Arduino UNO board, portable mini electric drill and a CNC shield. The Arduino UNO board [4] is often powered via the USB connection or with an external power supply. The power source is selected automatically. External External power may come either from an AC-DC adapter or a battery. The adapter is often connected by plugging a 2.1mm centre-positive plug into the board's power jack. Leads from a battery are often inserted within the GND and Vin pin headers of the source connector. The board may also operate on an external supply ranging from 6 - 20 V.The board may also operate on an external supply ranging from 6 - 20 V. If furnished with 7V, however, the 5V pin could provide but five volts and therefore the board may become unstable. If using quite 12V, the voltage regulator may overheat and damage the board. However, the recommended range is 7 - 12 V. The current requirement ranges from 1A - 3A.

On the other hand, the electric drill comes in with an attached AC/DC adapter which has an output rating of 9V/1A and is sufficient to rotate the drill bit at nominal speeds. The drill also has manual control for the drill speed (in rpm) which internally adjusts a potentiometer to get the required speed. Similarly, the CNC shield has a rated voltage requirement of 12 - 36 VDC. Now, the problem is to come up with a single (if possible, customised) Power Supply Unit (PSU) which can handle all the devices at once. The 9V rated AC/DC adapter in the electric drill is sufficient to power the Arduino UNO board although it is not even barely sufficient to power the CNC shield (let alone after the current calibration process). At the same time, the current rating of the drill is unknown. Assuming the current rating is in the neighbourhood of 1A (same as the output of the complementary adapter) then splitting its output wires won't help because the Arduino board won't receive the expected current for its stable or nominal operation (even if we assume it works in the neighbourhood of the lower bound of the range 1A-3A). These conditions imply that developing a customised PSU for the entire system is very difficult. Even if we use an AC/DC adapter which has a voltage rating of 9V - 12V and a current rating of 1A - 5A with separated output lines it would simply manage to power the

Arduino UNO and the electric drill but not the CNC shield. It implies that an adapter having a voltage rating somewhere near 24V will suffice but separate resistors would need to be used (preferably in a voltage divider network) to provide reduced voltage supply to the Arduino UNO board.

However, it should be noted that no customised PSU was developed for this project nor were these two electronic components powered by the same unit. The Arduino UNO development board was powered via the USB connection from a laptop while the drill was used along with its complementary AC/DC adapter without any modifications. The CNC shield, on the other hand, was directly powered from a DC regulated power supply.

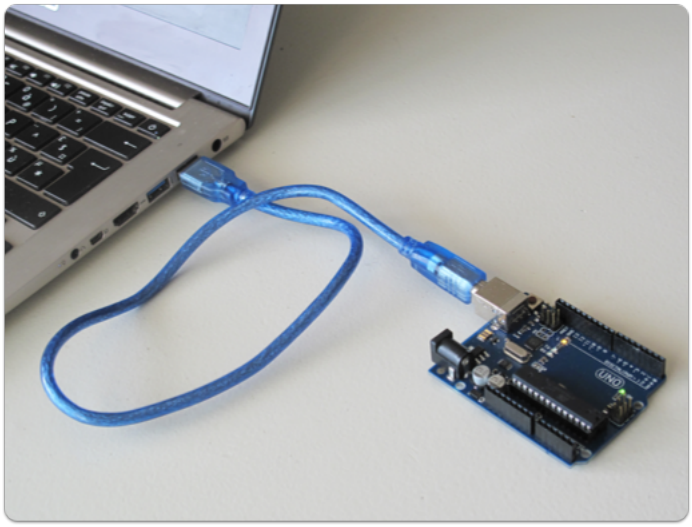


Figure 4.1: The Arduino UNO board being powered by USB supply from a laptop (the other power supplies have not been shown for the sake of convenience)

4.2 CNC motors

The most important part of this project is the CNC shield and associated motors. Our project consists of three axes (X-axis, Y-axis and Z-axis) which are controlled by two stepper motors (for X and Y axes) and a single servo motor (for vertical movement of drill bit i.e. Z-axis).

4.2.1 Stepper motors

Both X and Y axes are independently controlled by NEMA 17 stepper motors so we require two stepper motors in this project. The stepper motor assigned for the X-axis segment must be able to rotate a coupled rod, resting on which is the entire Y-axis segment (a part of whose weight is resting on this threaded rod and another smooth support rod). So a large capacity stepper motor needs to be considered. If possible, going for a larger capacity stepper motor (say NEMA 23) for the X-axis would ease out the operations in terms of fast and precise movement.

The stepper motor for the Y-axis segment is relatively less constrained in terms of weight capacity and any suitable capacity motor which can rotate a threaded rod will

do. The NEMA 17 stepper motors used for this project are bipolar i.e. there is only a single winding per phase. The driving circuit must be more complicated to reverse the magnetic pole, this is often done to reverse the current in the winding. However, all that complexity is handled by the dedicated drivers onboard the CNC shield shown in section 4.4. Following is an illustration of the motor (identical to the one used in the project) followed by a table of its summarised electrical and mechanical specifications.

Figure 4.2: NEMA 17 stepper motor similar to the one used in the project

Table 4.1: Table of brief specifications of single length NEMA 17 stepper motor

Mechanical specifications	
Weight	210 g
Holding torque	23 Ncm
Detent torque	1.2 Ncm
Rotor inertia	0.038 $kgcm^2$
Electrical specifications	
Phase current	1.5 A
Phase resistance	1.3 Ω
Phase inductance	2.1 mH

4.2.2 Servo motor

The only function of the servo motor of Z-axis is to produce minimal linear actuation of the drill bit (which is attached to the electric drill at its bottom) along the positive and negative Z-axis. Such type of behaviour may be seen when the CNC machine moves from one milling position to another by moving over non-processable space or area on the PCB. For this, we have used a servo motor which can raise the drill bit with gear precision. In terms of G code following is a sample self-explanatory code which may lead to such a movement of the Z-axis.

```
G91 (use relative positioning)
G1 Z10 (move z-axis up by 10 mm)
       (irrespective of initial position)
```

It should be noted that there are many other equivalent instructions which would produce the same linear actuation. Depending on what the software (described in section 5.2) determines to be feasible in terms of time and space, the appropriate set of G code(s) would be used at the concerning point in the operation of the CNC machine. Following is an illustration of the motor (identical to the one used in the project) followed by a table of its summarised electrical and mechanical specifications.

Figure 4.3: Gear precision servo similar to the one used in the project

Table 4.2: Table of brief specification of MG90S metal gear servo motor

Mechanical specifications	
Weight	13.4 g
Dimensions	22.5 × 12 × 35.5 mm
Stall torque	1.8 kgf·cm (4.8V), 2.2 kgf·cm (6 V)
Electrical specifications	
Operating speed	0.1 s/60° (4.8 V), 0.08 s/60° (6 V)
Operating voltage	4.8 V - 6.0 V
Dead band width	5 μs

4.3 Micro controller

The use of a microcontroller [4] for this specific project is to read the G codes and M-codes and interpret it into corresponding pulses to run the motors. Thus the microcontroller to be used can be of basic specifications. However, it should be noted that the CNC shield should be compatible with the microcontroller development board being used. At the same time, the total cost of the board and the shield shouldn't exceed the budgetary requirements of the project.

Depending on the above conditions we have used an Arduino UNO which is based on ATmega328 microcontroller. A compatible shield for this board is also available in the market. Out of 14 digital I/O pins(6 can be used as PWM outputs and hence major significance in this project), 6 analog inputs and a 16 MHz ceramic resonator (CSTCE16M0V53-R0), a USB connection, an influence(power) jack, an ICSP header, and a reset button.

It contains everything needed to support a microcontroller, we just simply have to connect it to a computer (or any other programming device having sufficient capabilities and relevant programming software onboard) with a USB cable.

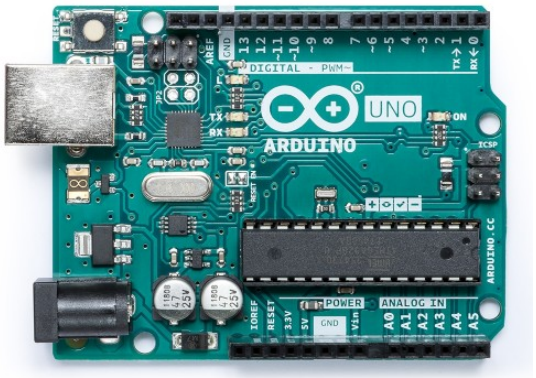
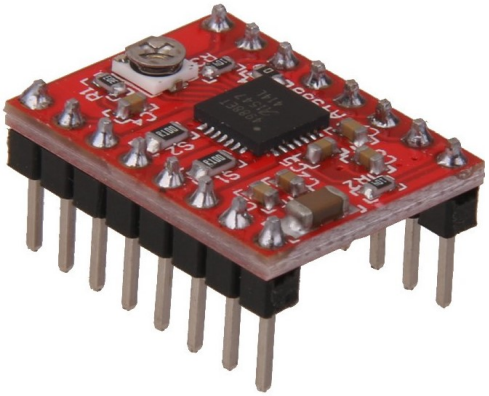


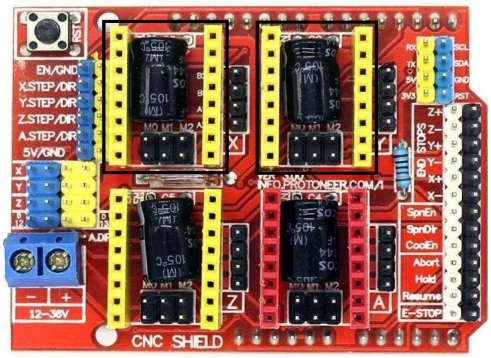
Figure 4.4: Arduino UNO, the microcontroller used in this project

4.4 Motor drivers and CNC shield

We have to drive three motors (two stepper motors and a single servo motor), for this, we have used a CNC shield. The CNC Shield V3 for Arduino is an Arduino compatible board that modifies the Arduino to a CNC controller. Using an open-source firmware it can control up to 4 Stepper motors using DRV8825 or A4988 stepper motor drivers making it easy to get your CNC projects up and running in a few hours.



(a) A4988 motor driver



(b) The CNC shield for Arduino used in this project

Figure 4.5: The two stepper motor drivers are placed in the two slots (indicated in black) on the CNC shield

4.5 Heat sinks

A heat sink is a passive heat exchanger that transfers the heat dissipated from the device to fluid medium, thereby allowing the device to work properly and efficiently without any issues created by the rise in temperature. In our project, the driver module may heat up due to the load on both the stepper motors. Hence, we have used two heat sinks for the two stepper motors connected via the driver module A4988. The heat sinks (which are standard metallic grills) sit on top of the two motor drivers.

Figure 4.6: Heat sink used for thermal management of the A4988 motor drivers used in this project

4.6 Hardware integration

In the following subsections, we describe in detail how the hardware interfacing is to be carried out. Apart from that certain calibration steps and standard conventions (in terms of wire colour and selection) have also been discussed at length.

4.6.1 Wiring and interfacing

The basic advantage of having shields for arbitrary applications related to microcontroller-based development is that it reduces the number and density of the wires required for a particular project or system. As stated before we have selected an Arduino UNO and a compatible CNC shield i.e. the Arduino CNC shield V3. Because of this, there are no wired connections between the microcontroller board and the shield and the shield resides perfectly on top of it by getting directly interfaced to the relevant pins.

However, choosing compatible shields has an added disadvantage - the shield completely consumes all of the available pins on the concerned development board. It could be the possibility that *not all pins* available on the development board are of any use in the project or the functioning of the overall system. However, due to the placement of the shield (on top of the board), there is absolutely no way to access the unused pins heavily affecting effective pin usage and any possible optimisation. This ***may*** create problems in larger systems and where other supplementary/complementary peripherals may need to be interfaced with the controller. The designer may end up purchasing excess computation units for the project which would again affect budgetary constraints decided for the project. In this project since there is no need for any additional peripherals other

than the one mentioned, choosing a compatible shield for a board turns out to be viable. Following is the pin mapping between the board and the shield for reference. [5]

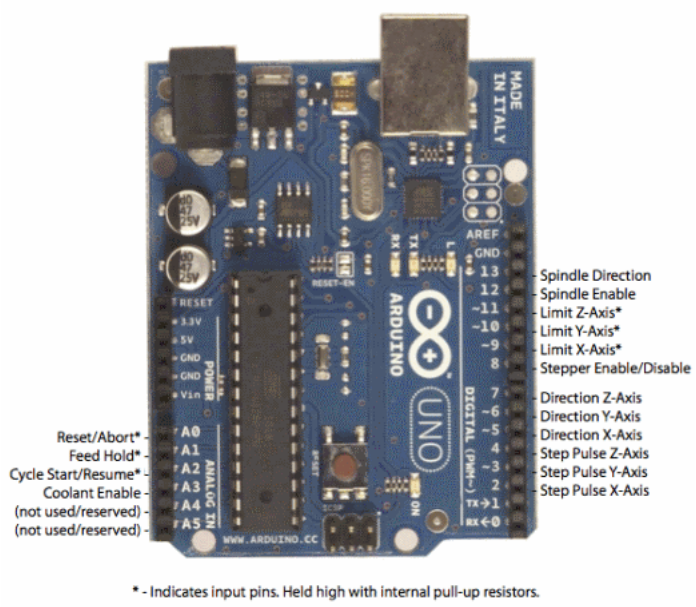


Figure 4.7: Pin mapping between the Arduino UNO board and the CNC shield showing the interfacing between the pins from both the modules once they are mounted suitably

Once the CNC shield has been mounted on top of the board we can proceed with attaching the A4988 motor drivers. Now with the reset button (RST) on the top left of the CNC shield (for directional ref.), the pin headers for attaching the motor drivers (in an anti-clockwise sense) are for the X, Y, A and Z axes respectively. Just as seen in the previous case here also we don't need any separate wires to interface the motor drivers to the shield thereby reducing the number of wires required and the wire interfacing complexity. All three motor drivers are identical to each other so their choice for the individual axes doesn't matter. The motor drivers for the X and Y axes are connected to their respective pin headers. These driver chips have extremely small hook-like structures (similar to pins) to attach heat sinks. The heat sinks are to be attached using these structures on to the motor drivers.

After these connections have been carried out the motors as well as a suitable DC regulated power supply need to be interfaced with the CNC shield. The stepper motors (corresponding to the X and Y axes) are now to be connected to the motor coils. The connection for the coils on the motor driver is annotated by pins A1, A2, B1 and B2 respectively. It should be noted that the annotation greatly varies across electronic components which are usually interfaced with stepper motors. However, there is no proper annotation on the actual stepper motor so standard wire colour conventions are to be followed. The motor coils (when viewed with all the wire outlets facing the user, starting from left to right) are to be connected to the driver pins according to the following table.

Since the pin names are quite confusing we have provided a table on the following page for the reference of the reader. It contains the equivalent pin names on the CNC shield, the motor driver and the wires from the stepper motor to be connected to the respective pins.

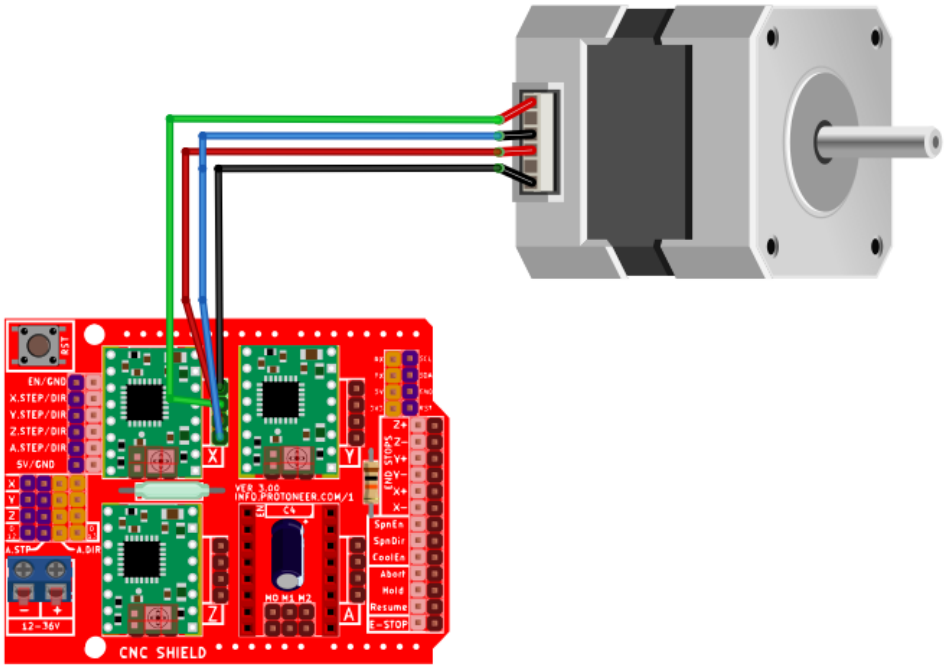


Figure 4.8: Interfacing of a single stepper motor with a corresponding A4988 motor driver mounted on the CNC shield

Table 4.3: Table of equivalent motor coil connections

Coil pair	Wire colour	Terminal Annotation	CNC Pin names	Order from Left to Right
1	Black	D	2B/B2	4
1	Green	B	2A/A2	1
2	Red	A	1A/A1	3
2	Blue	C	1B/B1	2

Interfacing the servo motor (which doesn't require a driver) to the CNC shield is relatively simple as opposed to interfacing of motors which require the same. The pulse input pin of the servo is interfaced with the **End stop Z+** pin on the CNC shield (as shown below). The Vcc and the GND pin of the servo motor are interfaced with the 5V and the GND pin on the CNC shield respectively.

After successfully interfacing the motors to the Arduino board via the CNC shield the board needs to be powered up. To do so it should be noted that the input voltage of Arduino CNC Shield V3.0 is DC 12V-36V, however, the upper bound should be avoided at all costs. The reason being although the input voltage supports power supplies up to 36V the motor drivers interfaced with the same have a supply voltage (V_{MOT}) rating of less than 36V. E.g. the A4988 motor driver used in this project has a supply voltage in the range 8-35V. Using anything outside this range will burn the motor driver. So, the power supply selection should be done carefully after referring to the corresponding motor driver's datasheet. In our project, a DC regulated power supply rated at a maximum of

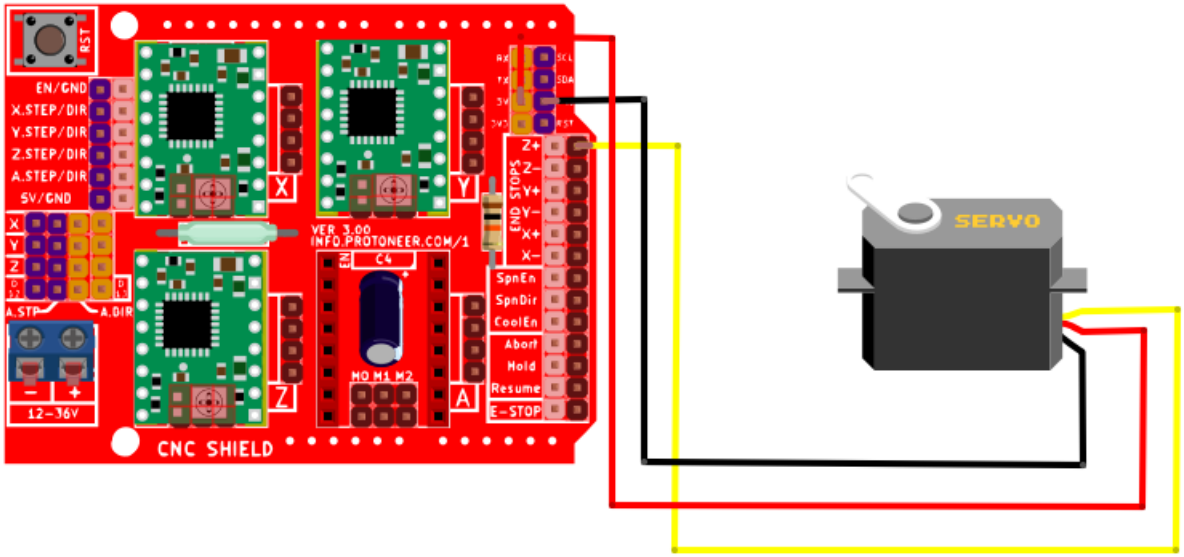


Figure 4.9: Interfacing a servo motor with the CNC shield

30V. In this project voltages in the range of 17V - 20V has been used for powering up the CNC shield.

Once the motor assemblies have been wired up to the controller via the CNC shield then the entire setup is now ready to be powered up. However, we still need to calibrate the motor drivers for optimal performance which has been explained in the following section.

Following is the overall circuit of the CNC machine

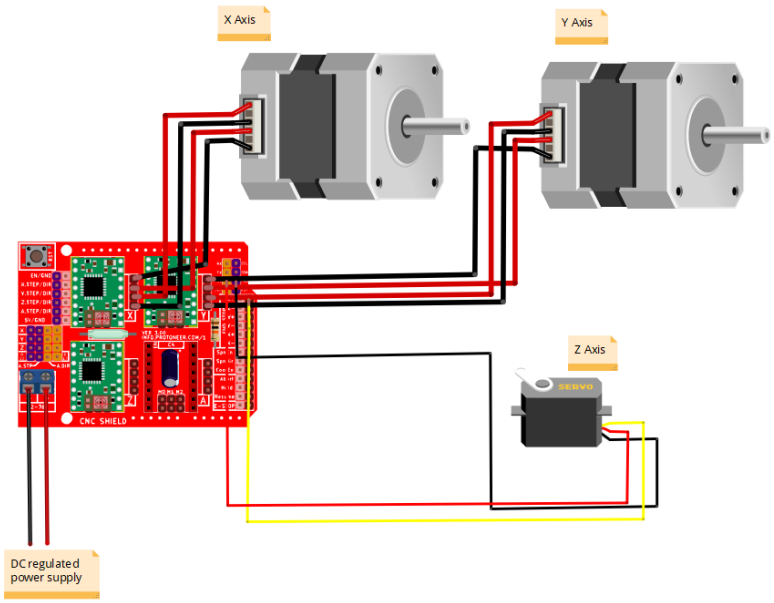


Figure 4.10: Complete electronic circuit for the CNC machine with various motors labelled along with their corresponding axes

4.6.2 Motor driver current calibration

For all CNC based applications, motor driver calibration is an important step before the entire system is powered into normal operation. A slight adjustment to the motor driver is required to ensure that it is not over or under powered. Not over or under powering is important to provide the ideal amount of torque for the extruder to run without skipping steps or overheating and damaging the motor. Before proceeding ahead with the calculations it should be noted that there are two types of motors based on their power and capacity used in many CNC based application or CAD/CAM processing in general. They are:

(a) For the compact but powerful motor, the maximum rated current to be set for stepper driver to output is 1.68A (These motors usually constitute the main electric drill used for engraving or drilling purposes).

(b) For the slimline motor, the maximum rated current to be set for stepper driver to output is 1.4A. (These motors usually constitute the gantry motors i.e. those responsible for the movement of the base and other components of the CNC machine).

There are usually two ways to calibrate the current limiting settings of an A4988 motor driver. Both are described below out of which the latter one is described in more detail and has been used in this project. In both cases, the trimmer potentiometer aboard the motor driver is used in the calibration process. [6]

By measuring the coil current

One way to line the current limit is to place the driving force into full-step mode and measure the current running through one motor coil while adjusting the current limit potentiometer. This should be through with the motor holding a hard and fast position (i.e. without clocking the STEP input). It should be noted that the current that is being measured is only 70% of the actual current limit setting since both coils are always on and limited to this value in full-step mode, so if the micro-stepping modes are enabled later, the current through the coils exceeds the measured full-step current by 40% (1/0.7) on certain steps; This should be taken into account when using this method to set the current limit. Also, it should be noted that this adjustment is to be performed again if the logic voltage is ever changed i.e. V_{DD} since the reference voltage that sets the current limit may be a function of V_{DD}. Further, it should also be noted that the coil current can be very different from the power supply current, so the current measured at the power supply shouldn't be used to set the current limit. Instead, the current meter should be placed is in series with one of the stepper motor coils.

By calculating the reference voltage

(a) To get the accurate result from the subsequent calculation, the value of the resistors used onboard the A4988 motor driver needs to be figured out. Different manufacturers use different resistors which affect the final settable figure. In terms of Pololu

A4988's, boards made before Jan 2017 used a Rsense (R_{CS}) value of 0.050Ω, boards made after Jan 2017 use a value of 0.068Ω. However, different boards use different values and the same should be verified from the concerned datasheets.In our project we use the variants manufactured before Jan 2017 i.e. those having $R_{CS} = 0.050\Omega$

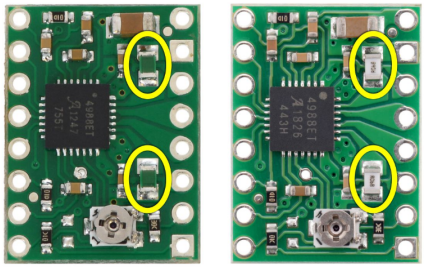


Figure 4.11: Pololu A4988 motor driver variants (resistors circled in yellow) with $R_{CS} = 0.050\Omega$ (left) and $R_{CS} = 0.068\Omega$ (right)

(b) The current limit $\boldsymbol{I_{MAX}}$ is related to the reference voltage $\boldsymbol{V_{REF}}$ as follows

$$\boldsymbol{I_{MAX} = \frac{V_{REF}}{8R_{CS}}}$$

Rearranged to solve for $\boldsymbol{V_{REF}}$ we get $\boldsymbol{V_{REF} = 8 \cdot I_{MAX} \cdot R_{CS}}$ where $\boldsymbol{R_{CS}}$ is the sense resistor described above. In this project the decided value of $\boldsymbol{I_{MAX} = 1.7A}$ and value of sense resistor i.e. $\boldsymbol{R_{CS} = 0.050\Omega}$.
Substituting in the above formula we get $\boldsymbol{V_{REF} = 1.7 \times 8 \times 0.050 = 0.68V}$.

(c) Now a multimeter is to be used for measuring the current value of reference voltage. The 2V DC option is to be set on the multimeter. The CNC shield should not be merely powered by the USB cable rather it should be fully powered to get a correct reading.

Figure 4.12: A multimeter with the required options set for calibration

(d) Now the red and black probes of the multimeter are connected to the potentiometer and the ground pin of the motor driver respectively as shown on the following page.

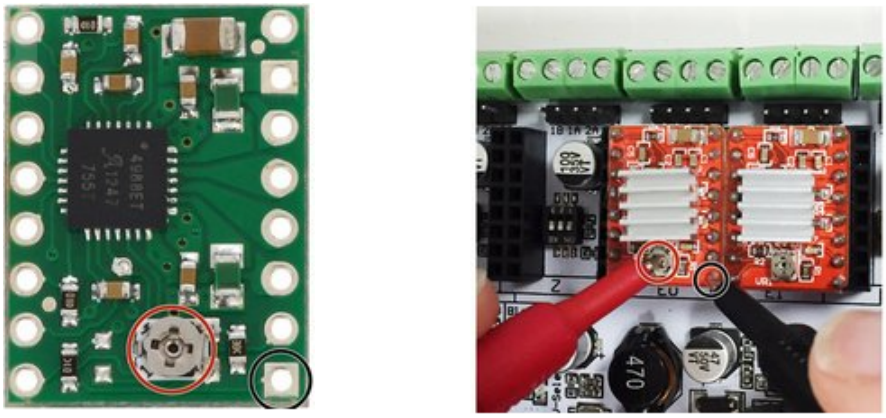


Figure 4.13: The points on the driver to be probed encircled in red and black (left) and multimeter probes of the same colour (right)

A ceramic screwdriver should be used to rotate the potentiometer to prevent shorting out any of the components whilst performing the adjustment as shown on the following page.

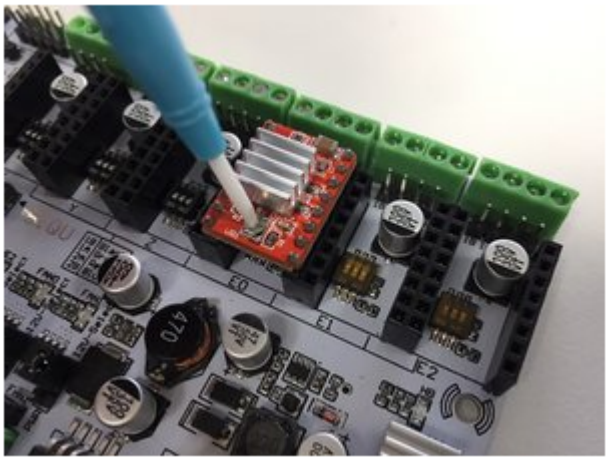


Figure 4.14: A ceramic screwdriver

(e) The first reading that would be obtained is the current value of the reference voltage (this would be higher or lower than the calculated value in the previous step indicating that the motor may either overheat or won't be sufficiently powered for nominal performance). On rotating the potentiometer anticlockwise or clockwise the value of reference voltage can be decreased or increased respectively to reach the calculated value from the previous step. Depending on the manufacturer of the board the direction of rotation may be different so the same should be checked before turning it too far.

(f) Correctly setting the potentiometer to the required value (from the multimeter reading) concludes the current calibration process of the motor driver.

4.6.3 Wire selection based on component interfacing

The contents of this section are optional, however, these are some recommended conventions and rules which are ought to be followed to develop a properly working self-explanatory system. Before proceeding further it should be noted that the CNC module is directly mounted on the Arduino while the motor drivers are directly mounted on the CNC shield. The heat sinks are fixed onto the motor drivers. Any form of wired connection is **not needed** for these purposes which further emphasises the advantages of using dedicated microcontroller shields for various applications wherein required.

Wires for generic interfacing

As far as generic interfacing is concerned red coloured wires are suitable for DC power supply while nornmal black wires are suitable for representing GND or negative terminals. All connections between a single peripheral to the controller should be coloured similarly. For e.g. blue. The colour should be changed as soon as we shift to a new peripheral.

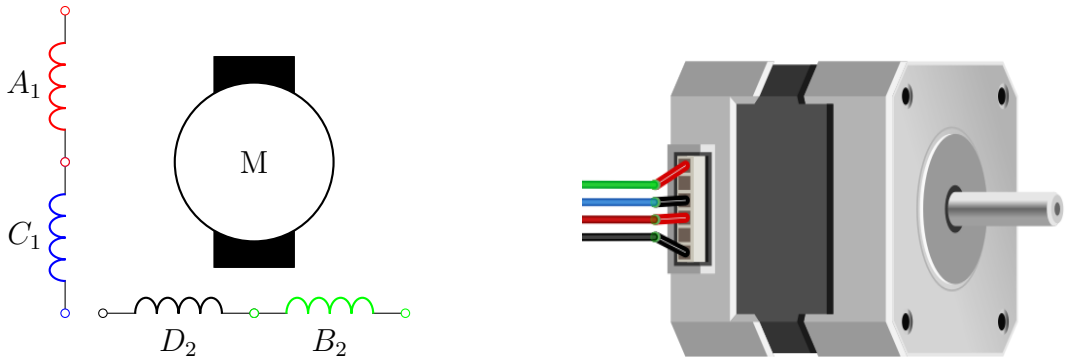
Wires related to interfacing between motor and CNC shield

Motor coil wires are connected to dedicated driver pins on the CNC module. However, as stated in section 4.6.1 there is no form of annotation on the motors indicating which coil is to be connected to what. Additionally, the wires are not labelled in any form indicating their actual electrical significance with the coils (inductor coils) present inside and their order of magnetisation. So the primary, as well as the proper way of identifying the wires, is their **colour**. In this project, we are using a pair of bipolar stepper motors which have a total of four leads/wires which are connected to the internal winding and can be accessed. Since there is only a single winding per phase, reversing the magnetic pole is relatively complicated as opposed to a unipolar stepper motor. However, that complexity is taken care of by using dedicated motor driver chips.

The four leads of the stepper motor [7] are customary to be coloured distinctively. In our project, the stepper motor wires are coloured as Red, Blue, Green and Black. The former two belong to the first coil pair and the latter two belong to the second coil pair. An illustrative figure is shown below with all the motor coils alongside the image of all the wires coming out from a stepper motor.

The exact terminology for the wires and their connection with the dedicated pins is better represented in table 4.3.

Apart from the wire colour, the length of the wires from stepper motors is also to be taken into account while designing the project. It should be noted that the stepper motor responsible for the movement of the X-axis will be stationery itself. So the four leads/wires from it can be cut to length to just match or exceed its (the motor) distance from the CNC shield. This will ensure proper wire management and avoid inter looping and tangling between the wires in any circumstances. However, at the same time, it should be noted that the stepper motor responsible for the movement of the Y-axis will be mobile. So there is a high chance that they may get tangled amongst themselves and severe the connections between the motor and the CNC shield during actual operation.



(a) Circuital representation of motor coils in a stepper motor

(b) Different coloured wires jutting out from a stepper motor

Figure 4.15: Illustration representing the coils of a stepper motor and their equivalent coloured annotation

Therefore it is recommended that they are cut to a length equal to the maximum possible linear displacement of the tooltip in the Y-direction. If the shield is placed at half the maximum possible displacement in either of the axes then wires from both the stepper motors are supposed to be cut to a length just exceeding this value. However, in general, the wires associated with the stepper motor of the Y-axis are supposed to be tauter as well as more flexible as compared to the ones associated with the X-axis.

Also apart from the above conventions, thin wires are supposed to be always wrapped in insulating heat shrink material to avoid wear and tear from repeated use cycles. A representative illustration has been given below (although they have not been used as of now).

Figure 4.16: (From bottom to top) naked wire junction, heat shrink tubing placed on the junction and the shrunk tubing on application of heat

The stepper motor wires which are supported by loose jumper headers at the ends for interfacing purposes should preferably be soldered to the female headers on the CNC shield at the end once the entire design is finalised.

Wires related to power supply based connections

This section is relevant because no customised universal PSU was designed for this project.

The AC/DC adapter for the portable electric drill should be long if possible or should be used with an extension cord because the continuous operation of the drill requires that the wire is taut at all times yet it should not be too taut that a miniature movement shall severe the connection. At the same time, the adapter plug should be tight.

Similarly, single-stranded wires should suffice for the power connection of the CNC shield but should be properly insulated with heat shrink tubing after the pair of ends are screwed tightly into the input DC jack of the CNC shield. Even better, double-stranded wires could be used for this purpose eliminating the need for insulating tubing.

A USB Type-A to Type-B cable is required to interface an Arduino with a programming device. Both the ends should be equally tight which would prevent probable loss of connection during the phase of flashing a program onto the controller. This would handle both the power supply and programming requirements of the microcontroller development board. If the power supply is to be provided using the onboard DC jack then it should be ensured that the input side fits snugly into the jack.

Chapter 5

Software methodology

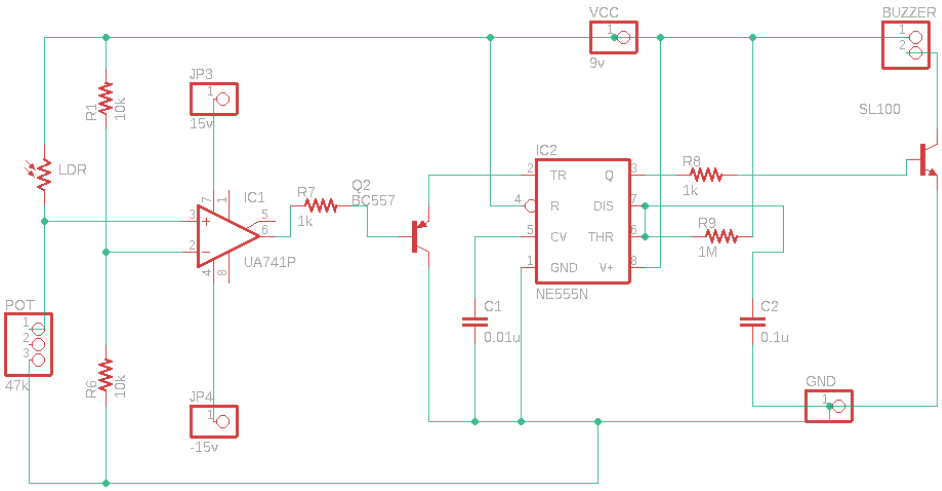
As discussed before in section 2.3, the designers of CNC machines for hobbyists and small scale applications rarely design an integrated software platform i.e. all sequential steps of the software process flow are NOT integrated into a single easy to use platform for the end-user. There could be multiple reasons for doing the same as discussed before. Not having enough knowledge about software development and design could be one of them. The process flow being considerably shorter as compared to what is commonly used in professional and commercial applications could be another. Hence, the sequential steps for the same have been elaborated below.

5.1 Schematic design software

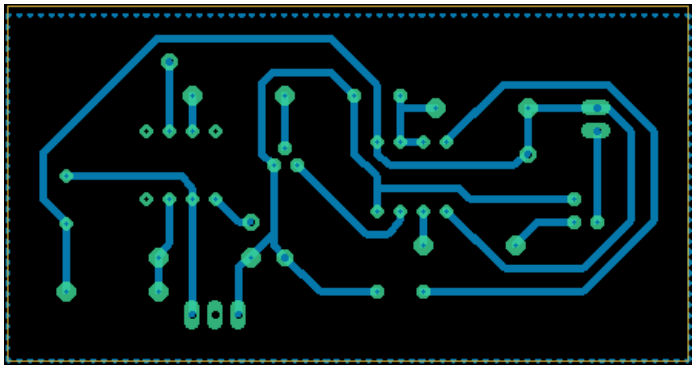
The first and foremost step in PCB design and fabrication is designing a schematic for the same. From the same schematic, an equivalent board file is designed. A schematic merely represents the interconnection between various electronic components and doesn't give any idea regarding how components are placed or appear on an actual PCB after it has been successfully manufactured and assembled in its entirety. However, the board file (.brd) gives the designer an exact idea regarding the area, footprint and placement of all electronic components in actual sense and also relative to each other. Hence, generating a proper board file is of utmost importance to the designer as this is the first step in our software process flow.

In our CNC machine, EAGLE ® is being used to design the schematic as well as the board file for the schematic design. They appear as separate files and on different tabs/windows on the EAGLE software. It should be noted that Eagle being a professional PCB design software offers a multitude of features. One of them being able to fabricate multilayered PCBs. Firstly, it should be noted that the designed CNC machine is capable of manufacturing only single-sided PCBs. In the worst-case scenario, the user may opt to manufacture a single layer (top or bottom) out of many possible layers present in the PCB using the concerned CNC machine. Secondly, our CNC machine's associated software flow doesn't include an image processing section so the user should be aware that the Gerber files generated by Eagle for a particular board would be generated "as is" and processed further without taking into consideration any possible faults that may occur or that may

remain in the board file. Following is a sample board file shown in the Eagle environment along with its corresponding schematic.



(a) Schematic for an Intruder alarm circuit



(b) Board file for an Intruder alarm circuit

Figure 5.1: A sample schematic and its corresponding board file used to test the CAM processor wizard in Eagle software

After checking for any possible faults in the board file regarding traces, routing, clearances etc. the corresponding Gerber files are generated. To do so, the CAM processor icon (which is available only in the board view) is selected and the appropriate layers are selected for generation of the corresponding Gerber files. Now since the concerned CNC is only capable of manufacturing single-sided PCBs, Gerber file of either the top or bottom layers is only generated. Drill files are also pre-generated which should be ignored since the CNC is capable of only engraving the wire tracks and is not capable of drilling any holes with the drilling equipment available to it. The output file paths for the generation of the Gerber files are usually fixed however, they can be changed wherein required. Once the required Gerber file is generated, it concludes the first step of the software process flow. Following is the image from the procedure.

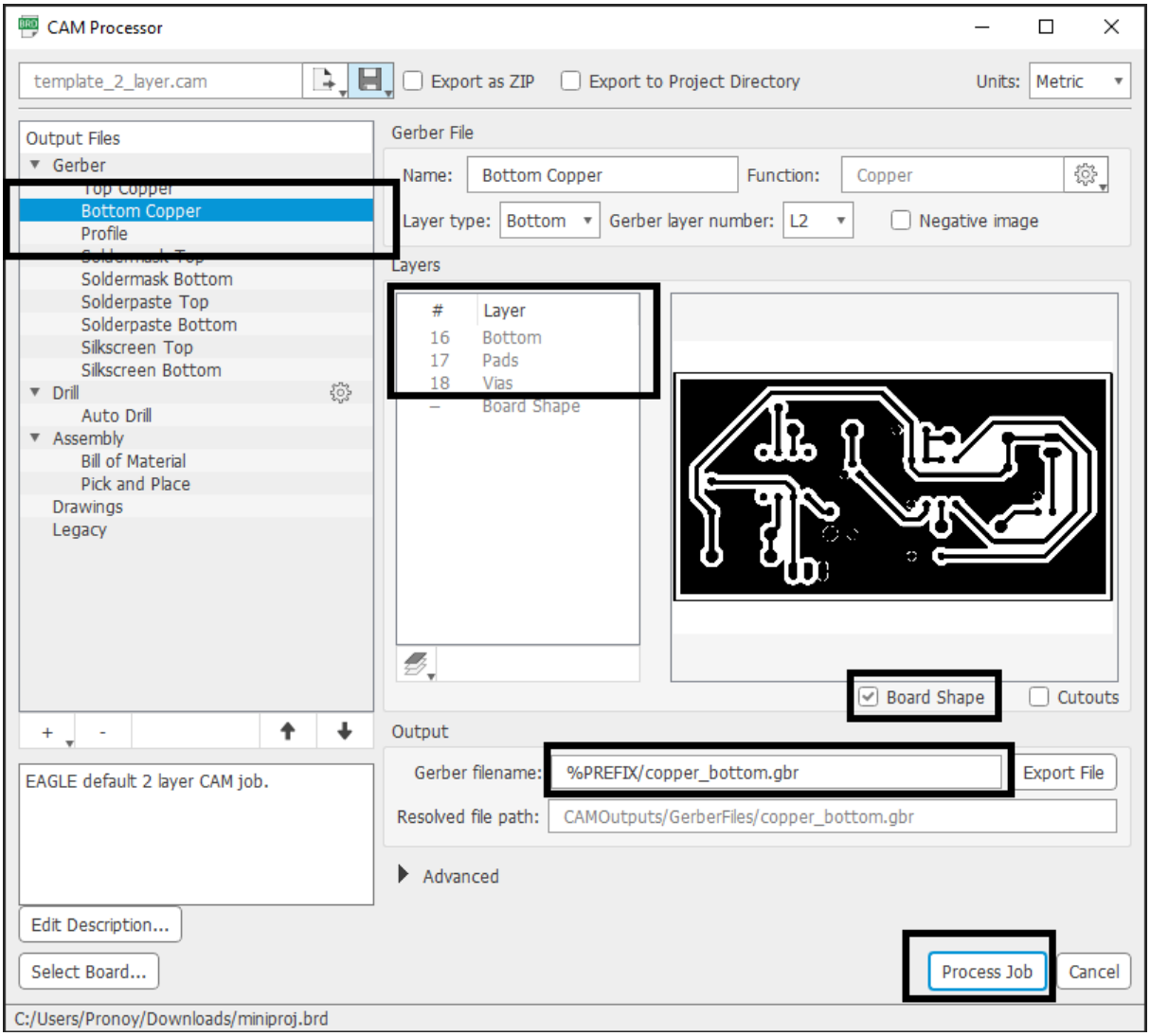


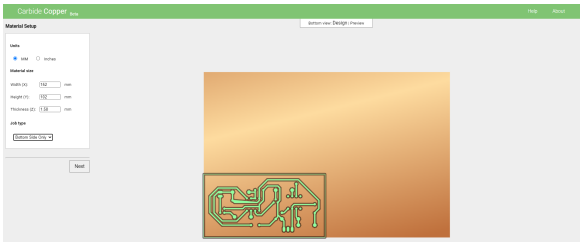
Figure 5.2: A snapshot of the CAM processor dialog box in Eagle (Starting from top left to bottom right): The layer for which the Gerber will be generated, the sub layers included, whether to incorporate the shape of the board, file name and location and **Process Job** button

5.2 G code generation

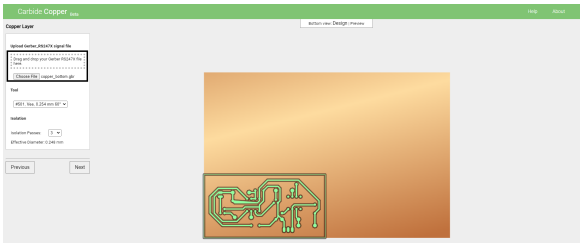
The next step is to generate the g code corresponding to the Gerber file obtained from the previous step. The software process flow moves online at this stage and hence requires internet access. G-code instruction of a PCB board is generated through the online platform Carbide3D (`http://copper.carbide3d.com/`). It is a free (however, not open-source) software for various CNC jobs and has multiple variants of itself for other specialisations such as woodwork, sheet metal engraving, carving, sculpting etc.

As soon as the online application is opened, it asks the user to enter the dimensions of the material in either of the desirable units namely mm and inches and also asks the user the concerned job type (i.e whether the top or bottom of the board is supposed to be traced). Followed by this it asks the user to upload the concerned Gerber file. It should

be noted that by using Eagle software, dedicated Gerber files are obtained for each of the layers i.e. a wrong choice of the job type (top or bottom) won't affect the final job. The user gets to choose the tool bit that is going to be used as well.



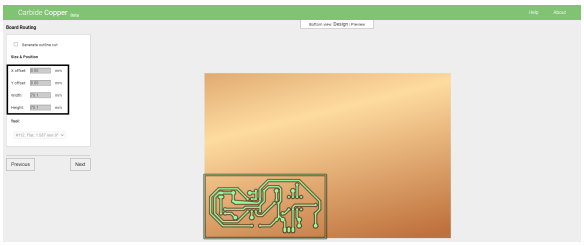
(a) Choosing the system of units and board dimensions



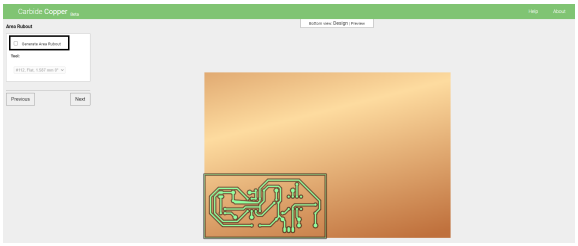
(b) Uploading Gerber file from the previous step (outlined in black)

Figure 5.3: The first two steps for G-code generation

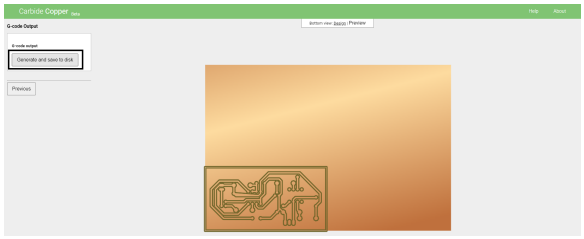
This is followed by asking the specifications regarding the drilling job of the concerned PCB which should be completely ignored. This step is followed by the routing step where again the tool dimensions for engraving purposes will be asked. Designers at this stage can choose to give various dimensional offsets if the need arises or else may choose to skip this step. Also if the boundary cutout for a PCB needs to be made it can be chosen at this stage itself. This step is followed by a relatively minor step of choosing whether to generate an area rub-out or not and any specialized tool to do the same. The following step successfully generates the G-code for the PCB and can be downloaded into the user's local machine. The software at each stage provides a preview of what is going to be traced out by the final CAM processor. Generating a G-code file for a PCB and saving it to local storage concludes the penultimate step in our software process flow. Following are some snapshots taken from this step of the software process flow.



(a) Routing and offset based settings (outlined in black)



(b) Options for area rub out and outline cut (outlined in black)



(c) Downloading the generated G-code file to disk (outlined in black)

Figure 5.4: The final steps for G - code generation

5.3 CAM processor

The final section [8] of the software flow is responsible for sending the obtained G code to the CNC machine which is understandable by the same. To do so the software must be able to recognise the port to which the Arduino board (together with which the CNC shield is connected). There would be two sets of software required for this step i.e. the Arduino compiler (along with the IDE if possible) and the Universal G code sender for the concerned OS platform.

To begin with, the Arduino IDE (or whichever relevant text editor is used) is opened up and only a single line of program code is compiled which is given below.

```
#include <grbl.h>
```

This includes the Gerber header files into your Arduino project i.e it gives a way to convert standard g code instructions to Arduino understandable commands. E.g. a single movement of the drill head is converted into a series of digital outputs (PWM signals to be precise) on the pins connected to the CNC shield which are then fed into the motor assembly connected with the shield. Since this header file is usually not an inbuilt file or library in the Arduino IDE it needs to be downloaded from external resources such as the official and updated GitHub repository of grbl - https://github.com/gnea/grbl. After extracting the zip archive the bin folder contains a readymade .ino file which can be simply opened and uploaded to the board (the file contains only the above line). After successfully doing so the serial monitor is opened. It is worthwhile to note that the serial monitor, in this case, acts as an excellent CLI for directly interacting with the motors via the CNC shield and via the grbl parser. Following are a few simple commands which can be used in this instance of the serial monitor. (There are advanced realtime commands which have been omitted from the following table for the sake of simplicity)

Table 5.1: Table of simple grbl commands

$$and $x=val	View and write grbl settings
$#	View G code parameters
$G	View G code parser state
$I	View build info
$N	View startup blocks
$Nx = line	Save startup block
$C	Check G code mode
$X	Kill alarm clock
$H	Run homing cycle
$J = line	Run jogging motion
$RST=$, $RST=#, and $RST=*	Restore grbl settings and data to defaults
$SLP	Enable sleep mode

This concludes the Arduino and grbl setup phase of the CAM processing.

After this basic setup, we can proceed to send the G code files generated in the previous step to the complete CNC assembly. To do so we would be needing a G code sender here, the Universal G code sender will be used (an appropriate version can be down-

loaded from https://winder.github.io/ugs_website/download/). Once the software is downloaded and extracted from its archive it is opened. Now, before proceeding onto uploading the obtained G code in the previous step the port and the baud rate should be the same as the one used in the previous phase (port to which the Arduino is connected and the baud rate that was being used for the serial monitor). After this initial setup, the **connect** icon is pressed (which is located alongside the baud rate selector). If all interfacing has been done properly this will lead to a successful connection between the software and hardware which will be indicated by this icon turning green. If it is not so the user can choose to refresh the ports and the baud rate (this step shall clear any unsent data from each of the COM ports and shall reset the baud rate to a default of 115200). However, it should be noted that the *active* state could still be *ALARM* which can be unlocked by choosing one of the machine actions from Menu bar.

Table 5.2: Table of common software states of the CAM processor

OFFLINE	The software is not connected to the CNC setup
ONLINE	The software is connected to a CNC setup
IDLE	The software is connected to a CNC setup but is not currently processing anything
HOLD	The software halted in the midst of a successful program execution
ALARM	The software encountered a fatal error and halted

Following this, the G code from the previous step can be directly loaded by choosing your file in the *file input* by clicking on the browse button. However, general testing of the motors is always recommended. This software platform is universal in nature as well as feature-rich so it has a custom jog controller (*jog* in the sense small but finite and user-controlled steps). All the connected motors can be rotated in both the directions by clicking on the + and − icons. A single press of this button corresponds to a single input pulse to the respective motors. Multiple presses may be needed to emulate an actual machine movement. All the implicit G code generated while testing the motors can be seen on the console on the bottom right. After some basic testing, the G code in the previous step is loaded and the CNC job is activated by clicking on the **RUN** button. If all goes well the CNC machine should begin its job immediately as per the commands written in the G code file. The movements of the tooltip could be visualized in 3D by choosing the 3d visualiser window from the Window option in the menu bar. It should be noted that the Grbl being a fully compiled parser for G code it could throw errors at the beginning itself on encountering unsupported commands. These commands could either be unsupported by the current version of the software OR could be machine codes (M codes) which are beyond the capabilities of the CNC machine. To find out or detect such commands the parser simply *reads all the ports* and finds out unavailable port addresses or ports which are not interfaced to any peripheral. After successfully removing all the errors depending on the complexity of the G code the CNC job would run successfully and produce the final engraved/etched PCB design. This concludes the overall software process flow and satisfies the main objective of this project.

5.4 Overall software process flow

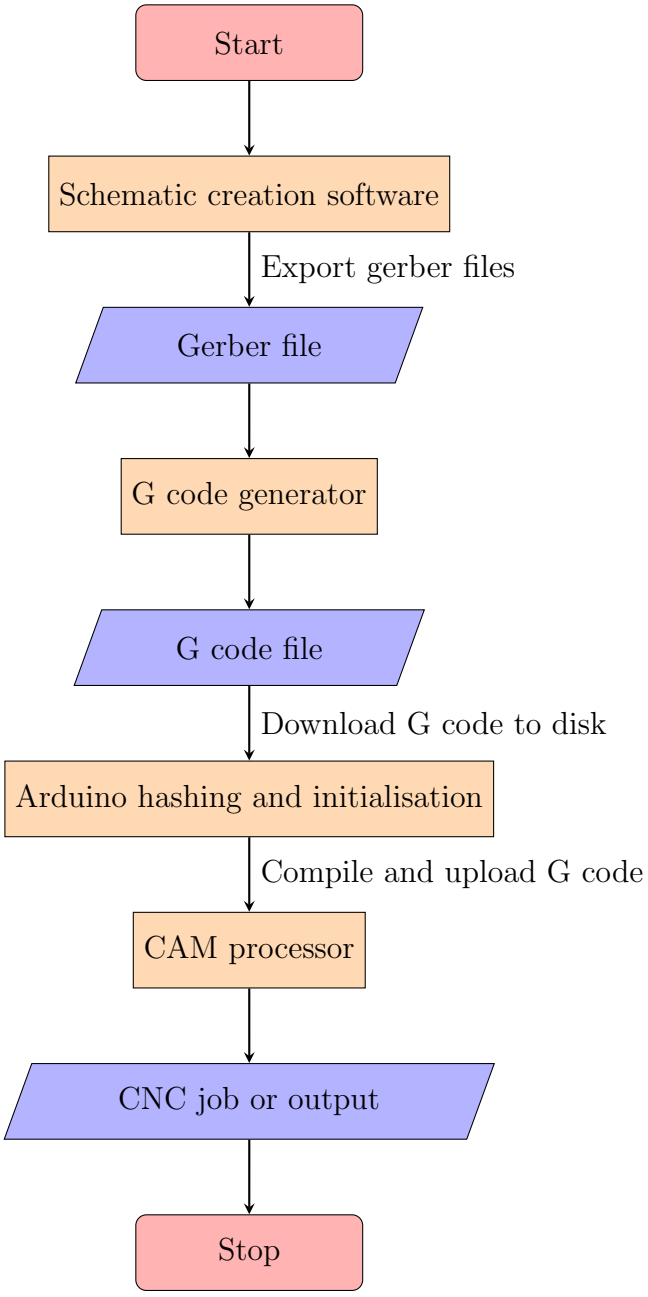


Figure 5.5: Overall software process flow represented as a simple flowchart

Step 1: Designing the schematic and the board file in Eagle or any other suitable schematic design software and then exporting the Gerber files from the same.

Step 2: Generating G-code corresponding to the Gerber files in Carbide3D (`http://copper.carbide3d.com/`) (section 5.2) and downloading the same to the local machine.

Step 3: Importing generated G-code file from the system into Universal G-code sender software for execution.

Step 4: Including the Grbl header file in an Arduino file (.ino) and compiling and uploading the same to the Arduino UNO board.

Step 5: Executing the G-code file from Universal G-code sender to instruct the CNC machine to process the job.

Chapter 6

Results and discussions

Following subsections shall discuss in detail, the various conclusions that were inferred by the developers/designers from the present investigation, a brief overview of the methodologies learned, deviations from ideality (if any), safety precautions which were followed and a creator's road map for implementation of a similar project and other moderately or less important details.

6.1 Basic inferences

The following list has been divided into two main sub-lists, namely a set of direct inferences (those inferences which are easily derived by following the normal development road map) and a set of indirect inferences (those inferences which need a deeper understanding of the application requirements of the project).

6.1.1 Direct inferences

D1 – Large sized assemblies must be supported by suitable base material

It is obvious from section 3.1 that the piece of polished wood is optimal for the base material design. The main reasons supporting this claim are

- Wood as a material, when used with considerable and uniform dimensions forms a good integrating platform on which other *heavy machinery* can be mounted. *Heavy machinery* here refers to any object of considerable weight and dimensions which easily couples with the wooden material either employing screw fastening mechanisms or directly integrated into slots made into the material. The bottom line is that both types of component material whether similar or dissimilar to wood, wood forms a natural integrating platform for all.
- Wood is light (has lesser weight in comparison with the sum of weights of other discrete components), sturdy, easy to clean as well as rests easily on a flat surface.

Hence, the choice of the base material.

D2 – Usage of buffered dimensions instead of going for exact measurements

This point has been reiterated through the entire report in various sections of importance. The reasons of consideration of buffered dimensions are that it always leaves out a tolerance band for error, i.e. even if some form of error creeps in due to improper or faulty machining work of a wide variety of instruments, they can be rectified easily. Although any form of error may lead to material wastage it is still way better than a faulty piece of material which is fundamentally useless and needs a replacement altogether. The downside is that the sum of all buffered dimensions would create a moderate to a large increase in overall dimensions of the entire physical structure/assembly thereby also hampering its overall portability.

D3 – Choice of proper adhesive material

As stated in section 3.8 a suitable material should be chosen for such purposes. Various types of fast-drying glue may seem to be an intuitively easy option however, they don't stand the test of time and are not viable in the long run. Therefore, throughout the entire assembly screws have been used for fixing and coupling together components of similar and dissimilar materials.

D1 – Optimum power supply

As stated multiple times in section 4.1 a single power supply should be sufficient for the entire project. However, designing a dedicated PCB for the purpose would have been simply tedious and time-consuming and hence was omitted for the current iteration of the project. But at the same time this increased density of power lines and cables. Additionally, it should be noted that power supply units are in general bulky and heavy e.g. adapters, single/dual regulated supplies etc. So it pays off to design a single dedicated circuit for all such powering purposes.

6.1.2 Indirect inferences

ID1 – Consideration of weight capacity of stepper motors

This point has not been stated anywhere else in the report, however, a little insight into what all would be coupled with a stepper motor shaft tells us that it will be bearing a considerable amount of heavy loads.The stepper motor assigned for the X-axis segment must be able to rotate a coupled rod, resting on which is the entire Y-axis segment (a part of whose weight is resting on this threaded rod and another smooth support rod). Invariably and indirectly the stepper motor for the X-axis is supporting the weight of both the rods as well as the weight of the complete Y-axis segment. So a large capacity stepper motor needs to be considered.The stepper motor for the Y-axis segment is relatively less

constrained in terms of weight capacity and any suitable capacity motor which can rotate a threaded rod will do.

ID2 – Stability of the entire structure

A little thought has been given to this part however while the CNC drill engraves tracks on the PCB, it should be noted that tremendous amounts of vibrations are produced in the system. A sturdy base material does help as stated in 6.1.1 D1 however, additional counterbalance weights at strategic positions is suggested for enhanced stability.

ID3 – Engraving and drilling bits

The review from section 2.3.4 should be taken into account while discussing engraving and drilling bits. There are two points to be kept in mind while choosing these bits for a CNC machine: the dimensions and the material. The ideal material is Tungsten Carbide although it is expensive. Drills for glass cutting and engraving may not be always suitable for PCB engraving purposes. For understanding what dimensions would be appropriate for the various engraving bits an iterative approach could be taken up. With the *available bits*, the smallest and largest possible wire thicknesses should be engraved. If the output quality is within the acceptable error limits set by the designers then the available set of bits is sufficient for the machine. If it is not, choose finer and harder bits for the same purpose and repeat the above process. Continue until the desired quality of output is obtained. Choose these engraving and drilling bits for your CNC machine.

6.2 Safety precautions undertaken

Throughout the potentially hazardous phase of the machining work, the operators of various machining tools, as well as the designers of the project, took multiple safety precautions. Following is a recommended set of safety precautions that must be followed whenever similar kind of projects is undertaken:

(a) Only experts and professional operators who are familiar with the machine should handle it under all circumstances.

(b) Keep a safe distance from rotating lathes, metal drills while they are in operation at least by 1.5 to 3 ft.

(c) While large-sized machinery is in operation, maintain a safe distance from its HV power supply.

(d) For rotating spindles, drill bits, etc. always make it a habit to spray lubricant on which the job is being done.

(e) Vices, holders, metallic jaws and other such similar equipment always pose a pinching hazard, handle them carefully.

(f) As a general rule, look up to the safety of yourself and others and be fully aware of your surroundings.

6.3 Standard testing procedures

Testing for any project is always divided into two major parts: each of the small components (either software or hardware) is tested at an individual level which is termed as unit testing. And after the given project has been completed successfully, its entire performance is judged or inspected end-to-end which is termed as full testing or integrated testing. The procedure for unit testing of various components (and the results obtained from them) are mentioned below. The reasons for incomplete full testing have also been mentioned in the following section.

6.3.1 Unit Testing

Testing of motors

It should be noted that motors **should not** be tested until and unless the calibration of the motor drivers as mentioned in 4.6.2 is fully complete. After the calibration has been completed, both the stepper motors should be interfaced as mentioned in 4.6.1. Now it should be noted that the rotation of the motor shafts needs to be tested for both *coarse* and *fine* movements of the drill head on the copper clad. A typical coarse movement could be drawing the outline cut (section 2.3.3) for a PCB while a fine movement could be drawing any trace width of minute dimensions. Rotation of the motor is very clearly visible for the former, however, for fine movements, the rotation may not be visible. To visualise the same, a paper or a cardboard sheet with the name of the axis written on it must be attached to the motor shafts. A clockwise arrow should be drawn as well. This will ensure that even minor rotations (in either direction) would be visible as the cardboard piece shall rotate with the shaft. Following is an image representing the same.

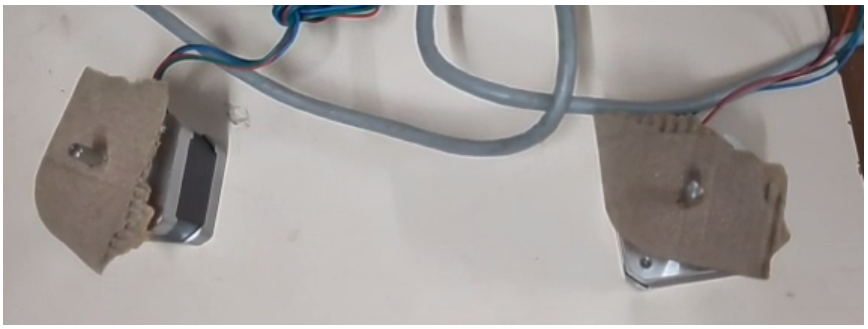


Figure 6.1: An illustration of how rotation of stepper motors is verified visually using cardboard labels during testing

A similar testing routine was developed for the stepper motors of this CNC machine. It was tested using the CAM processor software's *jog controller* feature. Using the jog controller gave the designers an ability to provide fine movement to both the stepper motors (in both the directions) as well as coarse movements by long-pressing any of the controller buttons on the software.The working of the motors was successfully tested and their performance was within their nominal design limits.

Testing of G code generation software

Testing of software is relatively easy as compared to working hardware. For testing the G-code generation software (described in section 5.2), two PCB Gerber files were generated from schematic design software and were fed into the web application. The first Gerber file corresponded to an RC circuit which is relatively simple to design while the second one belonged to a relatively complex PCB design i.e. an intruder alarm circuit (the one illustrated in figure 5.1b). In both cases, the settings regarding the copper-clad thickness, tool dimensions and extra features such as outline-cut were kept the same.

The G-code generated for both the PCBs were judged in terms of size and quality. The testing of the software after the inspection was deemed to be successful and very much within its design limits.

Testing of CAM processor software along with unit components

The CAM processor software is quite vast so the testing of only the features relevant to this project has been detailed here. The main features of this software are the code editor, visualiser, jog controller and a display showing the real-time coordinates of the tooltip in appropriate units. Following is a brief description of how they were tested.

To begin with, the G-code file corresponding to the RC circuit from the previous step was uploaded in the software. All the functionality relating to connecting the CNC setup to the PC is verified. This may include (depending on the software) a COM port indicator, baud rate indicator, an icon denoting successful connection etc. After verifying these settings the code editor that is present onboard can be tested.

To test the editor, deleting, adding, commenting and editing a few lines is recommended. If the editor saves them in realtime then the testing is deemed to be successful. Following that the jog controller can be tested. One by one each of the directional buttons (+/-) corresponding to each of the three axes are tested. If the relevant motors are rotating in the required directions then it can be concluded that the jog controller is working properly. The same was done for this CNC setup as well.

After all the pre-operation steps are completed then only is it possible to test the live visualiser and the indicators showing the controller states. After clicking on the run button, the G-code program starts running. If all goes well, depending on the complexity of the program and the PCB, the entire program may run over five minutes. The real-time machine status is indicated on the **DRO controller** where the coordinates of the tooltip would be displayed in double-digit precision. At the same time, live visualiser can be tested. The entire tool path is visible in yellow (which is precomputed once the G-code is loaded) and only the tooltip is shown. If with the rotation of the motors, appropriate movement of the tooltip is visible on the visualiser then it is highly probable that the CNC set up along with the software is working fine. However, to completely ensure that the CAM processor software is working properly, integrated or full testing would be required. All the aforementioned steps were followed to ensure that the CAM processor software was working properly. The CAM processor software testing was deemed to be successful.

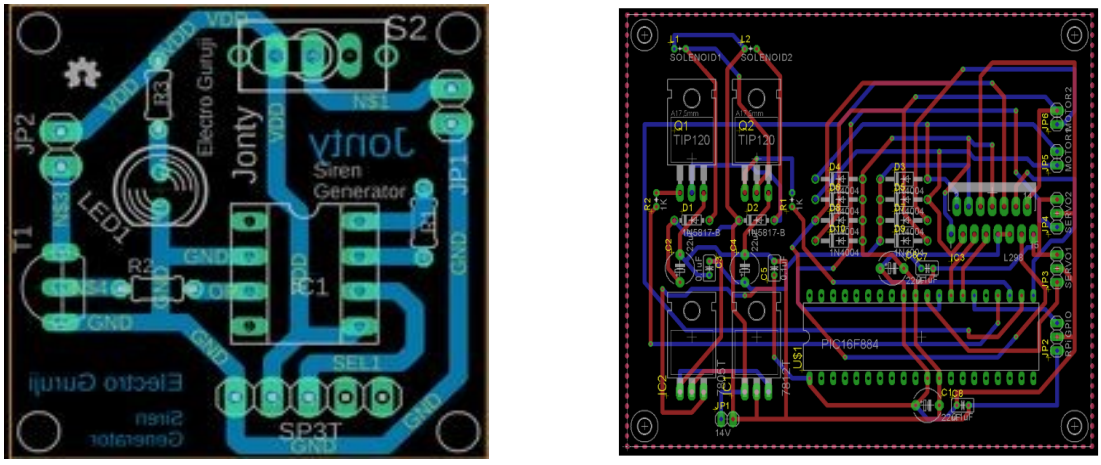
6.3.2 Integrated or full testing

As stated at the beginning of this section, testing of components used to make up a system is a multistage and iterative process as opposed to a single-step process. While testing the motors described in section 4.2.1 it was realised that for coarse movements, the capacity of the stepper motor for the X-axis was insufficient to rotate a heavy threaded rod. Added to this was the fact that an entire identical axis assembly would have been loaded on this axis. The NEMA 17 stepper motor variants were incapable of this job. An ideal upgrade could have been the NEMA 23 variants. However, due to time constraints, the same could not be procured in the available period and **hence the project, as well as its integrated or full testing, remains incomplete**.

6.4 Test case selection

For any given project, testing is invaluable. However, the procedures to be followed for the same are equally important and should be documented well. The reason being that any system has various levels of limitations. It is important to provide test inputs in a way that is within the nominal limits of operation of a machine. Similarly, it is also expected that the output would be within the design limits for the precision and accuracy of the machine. For our CNC machine, we have highlighted some important points to follow and remember while testing our project. These points would also be beneficial to someone who wants to design a dedicated testing rig for a similar such system. [9]

(a) The CNC machine can only handle single-layer PCBs - As of now (stated multiple times in this report) this CNC machine is capable of milling only single-sided or single-layer PCBs. Therefore, it should not be tested against G-code obtained from multilayer PCBs. To genuinely emulate milling of multilayered PCBs, the tester can generate G-code of individual layers (a feature of all common schematic design software) and test the same on our machine. Following is how a and test case looks like.

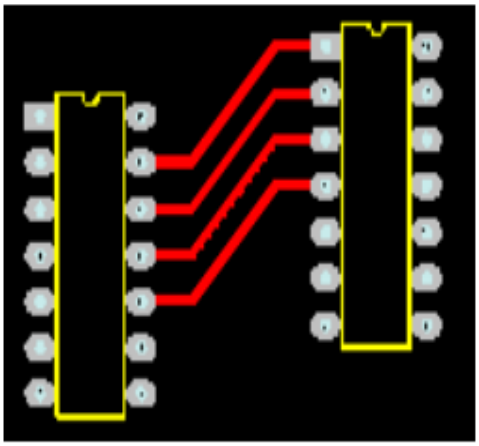


(a) Single layered PCB

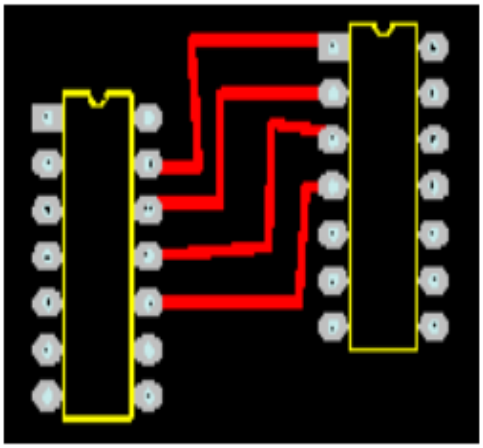
(b) Double layered PCB

Figure 6.2: The two illustrations above represent what could be a good and a bad test case respectively for the first rule

(b) The CNC machine doesn't automatically rectify issues in PCB design - The rules and guidelines for creating a nice industry-grade PCB are large in number. Every PCB given as an input may not obey the said specifications. There could be faults in corners, tracks, track spaces etc.. These problems (and rectification of the same) is the sole responsibility of the user/tester and it should be noted that the software associated with the CNC machine is not capable of understanding the same. If an inappropriately designed PCB (satisfying the previous rule) is given as an input, an equivalent output would be generated within the design limits of the CNC machine. The software doesn't include any image processing routines and hence no errors would be rectified automatically. Following is an example of what could be a *good* and *bad* test case in this scenario.



(a) Components with good routing



(b) Components with bad routing

Figure 6.3: The two illustrations above represent what could be a good and a bad test case respectively for the second rule

(c) G-code containing machine codes - While testing this CNC machine it should be noted that it doesn't include any advanced machine control functions such as *spindle control*, *coolant ON/OFF*, *tool change* etc. although the CNC shield has ports for the same. However, the generated G-code may very often have machine codes indicating one or multiple of the above operations. The tester must ensure that such lines of code are commented OR are removed from the program in their entirety. Additionally, some codes belong to different G-code versions and are not compatible with all machines. Such G-codes **should be** replaced with suitable alternatives.

(d) Placement of the machine on a straight and flat surface - For all testing regimes it should always be ensured that the CNC machine is placed on a straight and flat surface which is (preferably) heavy and stable. Although the machine has a base-board of large dimensions made using wood, the copper-clad itself could have minor and major aberrations or non-uniformities on its surface. Although procedures like auto levelling and others have been mentioned in the literature review, those features could not be implemented due to time constraints. Hence, ensuring a proper flat surface during testing or normal operation is the sole responsibility of the user. Following are images showing what a *good* and *bad* test case would look like.

(a) Clean PCB copper cladding which is straight and flat

(b) An unclean PCB copper cladding which may have micro bends and abrasions on its surface

Figure 6.4: The two illustrations above represent what could be a good and a bad test case respectively for the fourth rule

The above guidelines, if followed properly are sufficient to carry out proper testing for the CNC machine.

6.5 Scope for future work

Although the project was completed it does leave out on many critical points wherein it could have been improvised. Following are some points which can be looked upon by any designer or developer who would like to further develop the project. These key points are only a few of those areas where primary development should be focused on and doesn't necessarily include everything where improvisation could be carried out (that is left as an exercise to developers).

(a) Develop a proper PSU for the entire system.

(b) Develop a single PCB containing the entire circuitry to control the system.

(c) Use a more powerful and dedicated drilling motor for the actual engraving purpose.

(d) Develop a single integrated software platform for the entire engraving process which facilitates every step from G-code generation to the final PCB output.

(e) Optimise or develop G-code generation routines which can handle multiple PCBs at the same time without any manual intervention till their net dimensions are within 15 cm x 15 cm.

(f) Improvise necessary hardware and/or software to handle multi-layered PCBs.

(g) Improvise necessary hardware and/or software to carry out the special procedures listed in section 2.3.3 automatically.

(h) Develop a proper enclosure for the entire CNC machine if possible with cooling and vacuuming systems.

Chapter 7

Conclusions

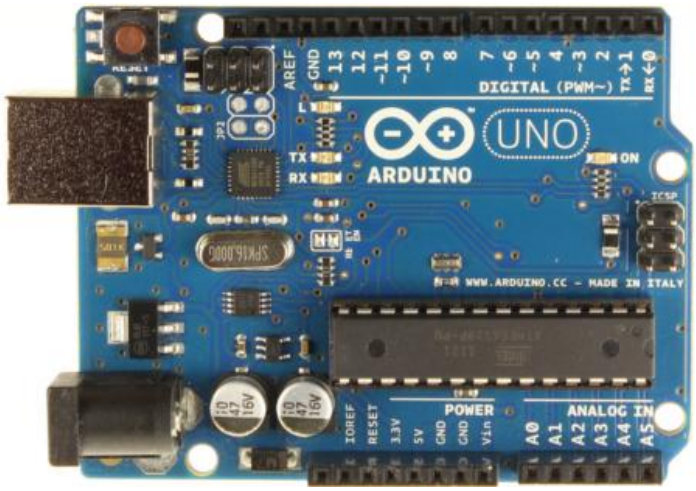
A CNC based PCB milling machine capable of milling only single-sided PCBs of maximum dimensions 15 x 15 cm was developed. All the basics of CNC based milling and CAD/CAM processing were studied and used in this project. The entire structure was primarily made of wood while the various rods and shafts used materials ranging from moderate steel to stainless steel. Various aspects of mechanical engineering were learned in the process. These include (but not limited to) cutting, filing, drilling, polishing of various surfaces and common measurement techniques.

On similar terms, multiple concepts related to electronics were learned during the development phase of the project. These include (but not limited to) standard calibration procedures, understanding the motor operation and capacity limitations, effective wiring and simple software-based concepts such as Gerber files and G-code conversion.

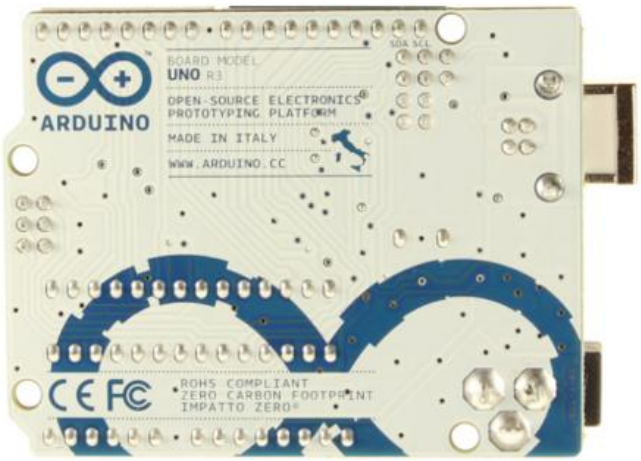
Overall it turned out to be a complete knowledge enriching experience in a multitude of domains.

Datasheets of all components

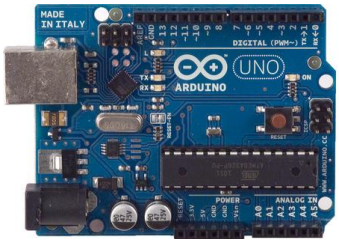
Arduino Uno



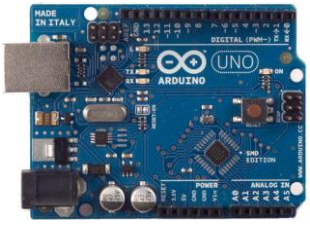
Arduino Uno R3 Front



Arduino Uno R3 Back

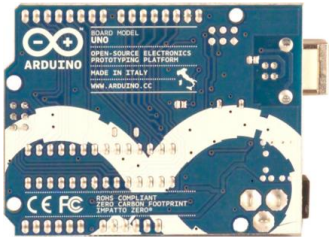


Arduino Uno R2 Front



Arduino Uno SMD

Arduino Uno Front



Arduino Uno Back

Overview

The Arduino Uno is a microcontroller board based on the ATmega328 (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.
The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.
Revision 2 of the Uno board has a resistor pulling the 8U2 HWB line to ground, making it easier to put into DFU mode.
Revision 3 of the board has the following new features:

- 1.0 pinout: added SDA and SCL pins that are near to the AREF pin and two other new pins placed near to the RESET pin, the IOREF that allow the shields to adapt to the voltage provided from the board. In future, shields will be compatible both with the board that use the AVR, which operate with 5V and with the Arduino Due that operate with 3.3V. The second one is a not connected pin, that is reserved for future purposes.
- Stronger RESET circuit.
- Atmega 16U2 replace the 8U2.

"Uno" means one in Italian and is named to mark the upcoming release of Arduino 1.0. The Uno and version 1.0 will be the reference versions of Arduino, moving forward. The Uno is the latest in a series of USB Arduino boards, and the reference model for the Arduino platform; for a comparison with previous versions, see the index of Arduino boards.

Summary

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V

Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz

Schematic & Reference Design

EAGLE files: arduino-uno-Rev3-reference-design.zip (NOTE: works with Eagle 6.0 and newer)
Schematic: arduino-uno-Rev3-schematic.pdf
Note: The Arduino reference design can use an Atmega8, 168, or 328, Current models use an ATmega328, but an Atmega8 is shown in the schematic for reference. The pin configuration is identical on all three processors.

Power

The Arduino Uno can be powered via the USB connection or with an external power supply. The power source is selected automatically.
External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.
The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.
The power pins are as follows:

- **VIN.** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V.**This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board. We don't advise it.
- **3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- **GND.** Ground pins.

Memory

The ATmega328 has 32 KB (with 0.5 KB used for the bootloader). It also has 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the EEPROM library).

Input and Output

Each of the 14 digital pins on the Uno can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

- **Serial: 0 (RX) and 1 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
- **External Interrupts: 2 and 3.** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- **PWM: 3, 5, 6, 9, 10, and 11.** Provide 8-bit PWM output with the analogWrite() function.

- **SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK).** These pins support SPI communication using the SPI library.
- **LED: 13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.

The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function. Additionally, some pins have specialized functionality:

- **TWI: A4 or SDA pin and A5 or SCL pin.** Support TWI communication using the Wire library.

There are a couple of other pins on the board:

- **AREF.** Reference voltage for the analog inputs. Used with analogReference().
- **Reset.** Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

See also the mapping between Arduino pins and ATmega328 ports. The mapping for the Atmega8, 168, and 328 is identical.

Communication

The Arduino Uno has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The '16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However, on Windows, a .inf file is required. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1).
A SoftwareSerial library allows for serial communication on any of the Uno's digital pins.
The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation for details. For SPI communication, use the SPI library.

Programming

The Arduino Uno can be programmed with the Arduino software (download). Select "Arduino Uno from the **Tools > Board** menu (according to the microcontroller on your board). For details, see the reference and tutorials.
The ATmega328 on the Arduino Uno comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK500 protocol (reference, C header files).
You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details.
The ATmega16U2 (or 8U2 in the rev1 and rev2 boards) firmware source code is available . The ATmega16U2/8U2 is loaded with a DFU bootloader, which can be activated by:

- On Rev1 boards: connecting the solder jumper on the back of the board (near the map of Italy) and then resetting the 8U2.
- On Rev2 or later boards: there is a resistor that pulling the 8U2/16U2 HWB line to ground, making it easier to put into DFU mode.

You can then use Atmel's FLIP software (Windows) or the DFU programmer (Mac OS X and Linux) to load a new firmware. Or you can use the ISP header with an external programmer (overwriting the DFU bootloader). See this user-contributed tutorial for more information.

Automatic (Software) Reset

Rather than requiring a physical press of the reset button before an upload, the Arduino Uno is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2/16U2 is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload. This setup has other implications. When the Uno is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.
The Uno contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see this forum thread for details.

USB Overcurrent Protection

The Arduino Uno has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

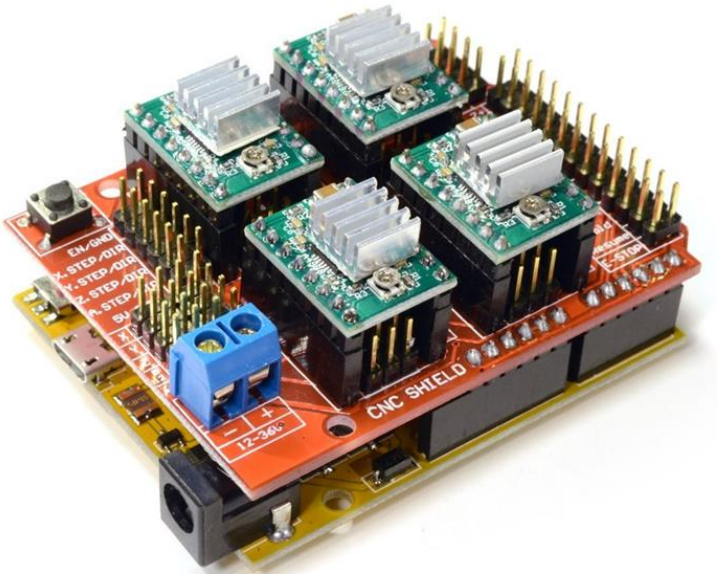
Physical Characteristics

The maximum length and width of the Uno PCB are 2.7 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Four screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), not an even multiple of the 100 mil spacing of the other pins.

User Manual

3-Axis CNC/Stepper Motor Shield for Arduino

The Arduino CNC Shield makes it easy to get your CNC projects up and running in a few hours. It uses opensource firmware on Arduino to control 4 stepper motors using 4 pieces of A4988 Stepper Motor driver breakout board, with this shield and ArduinoUno/Mega, you can build all kinds of robotics, linear motion project or projects including CNC routers, laser cutters and even pick&place machines.



SKU: DRV1001

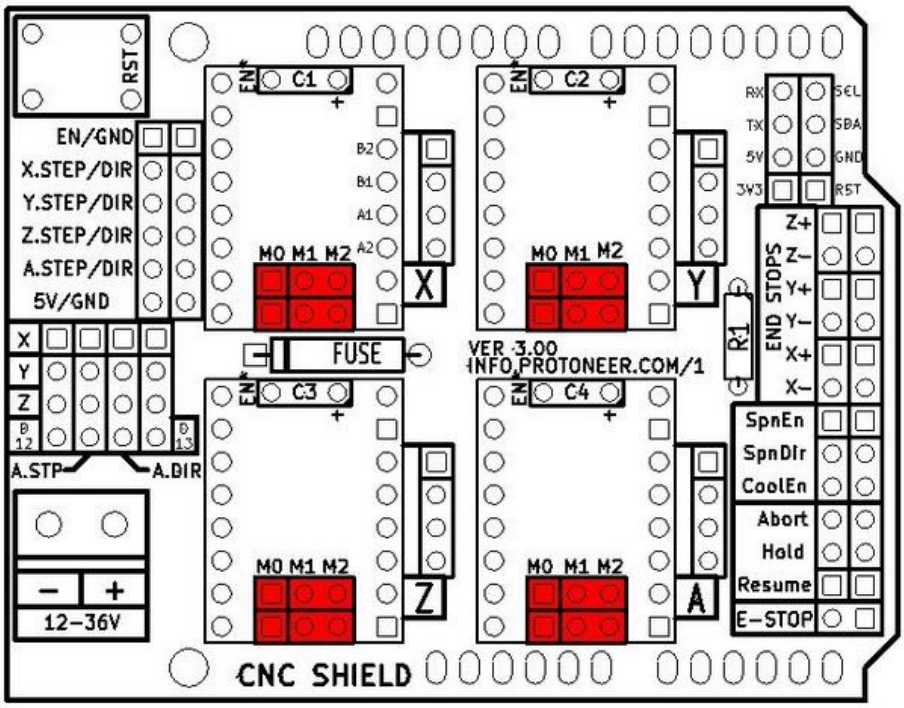
Brief Data:

- GRBL 0.9 compatible. (Open source firmware that runs on an Arduino UNO that turns G-code commands into stepper signals)
- 4-Axis support (X, Y, Z , A-Can duplicate X,Y,Z or do a full 4th axis with custom firmware using pins D12 and D13)
- 2 x End stops for each axis (6 in total)
- Coolant enable
- Uses removable A4988 compatible stepper drivers. (A4988, DRV8825 and others)(Not Included)
- Jumpers to set the Micro-Stepping for the stepper drivers. (Some drivers like the DRV8825 can do up to 1/32 micro-stepping)
- Compact design.
- Stepper Motors can be connected with 4-pin Molex connectors or soldered in place.
- Runs on 12-36VDC. (At the moment only the DRV8825 drivers can handle up to 36V so please consider the operation voltage when powering the board.)

Table of Contents

1. Configuring Micro Stepping for Each Axis

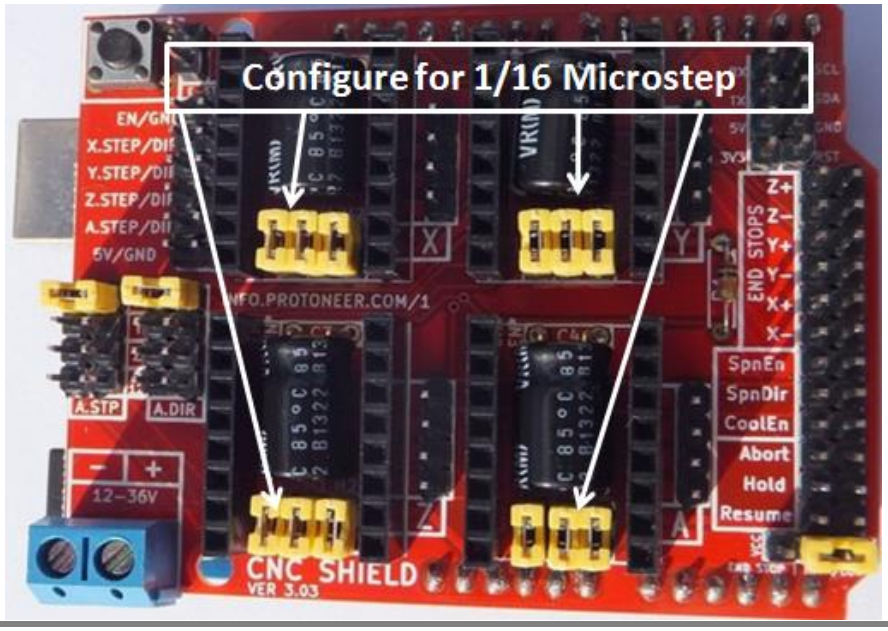
Each axis has 3 jumpers that can be set to configure the micro stepping for the A4988 plug-in driver board.



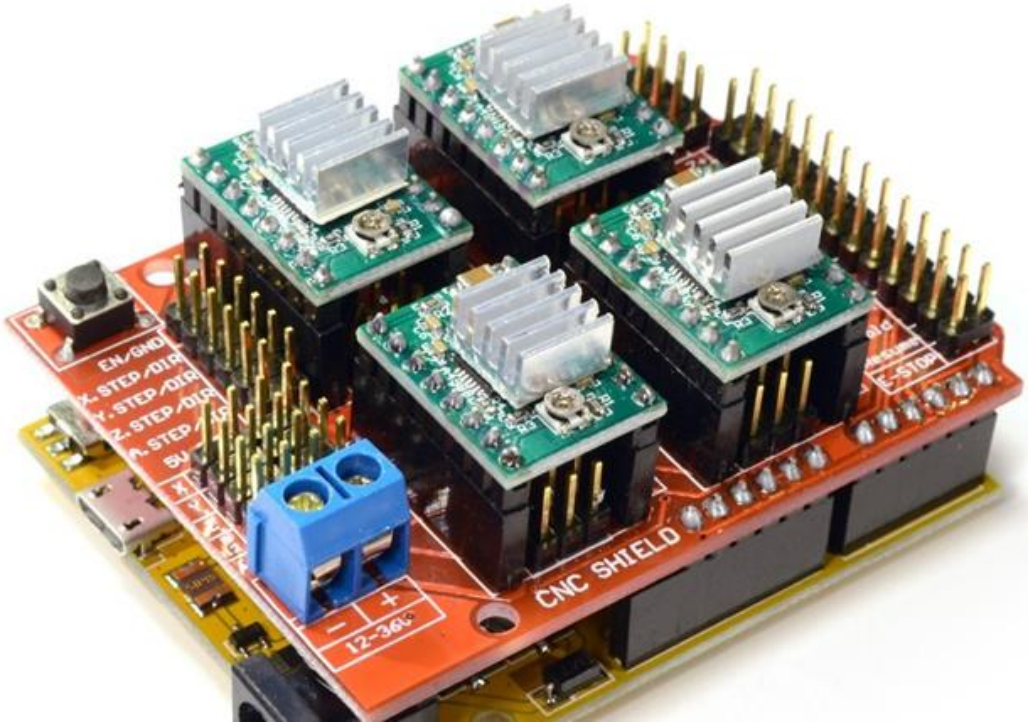
Micro-stepping jumper location, before inserting A4988.

In the tables below ‘High’ indicates that a jumper is insert and ‘Low’ indicates that no jumper is inserted.

MS0	MS1	MS2	Microstep Resolution
Low	Low	Low	Full Step
High	Low	Low	½ Step
Low	High	Low	¼ Step
High	High	Low	$^1/_8$ Step
High	High	High	$^1/_{16}$ Step



After setting the microstep jumper, you can plug-in A4988 driver boards as shown in the photo below. The photo also shown this CNC sit nicely on top of Arduino Uno board, without any external jumper wires.



!!! Beware of the orientation of the A4988 driver boards! You will destroy the A4988 driver board if plug-in with wrong orientation.

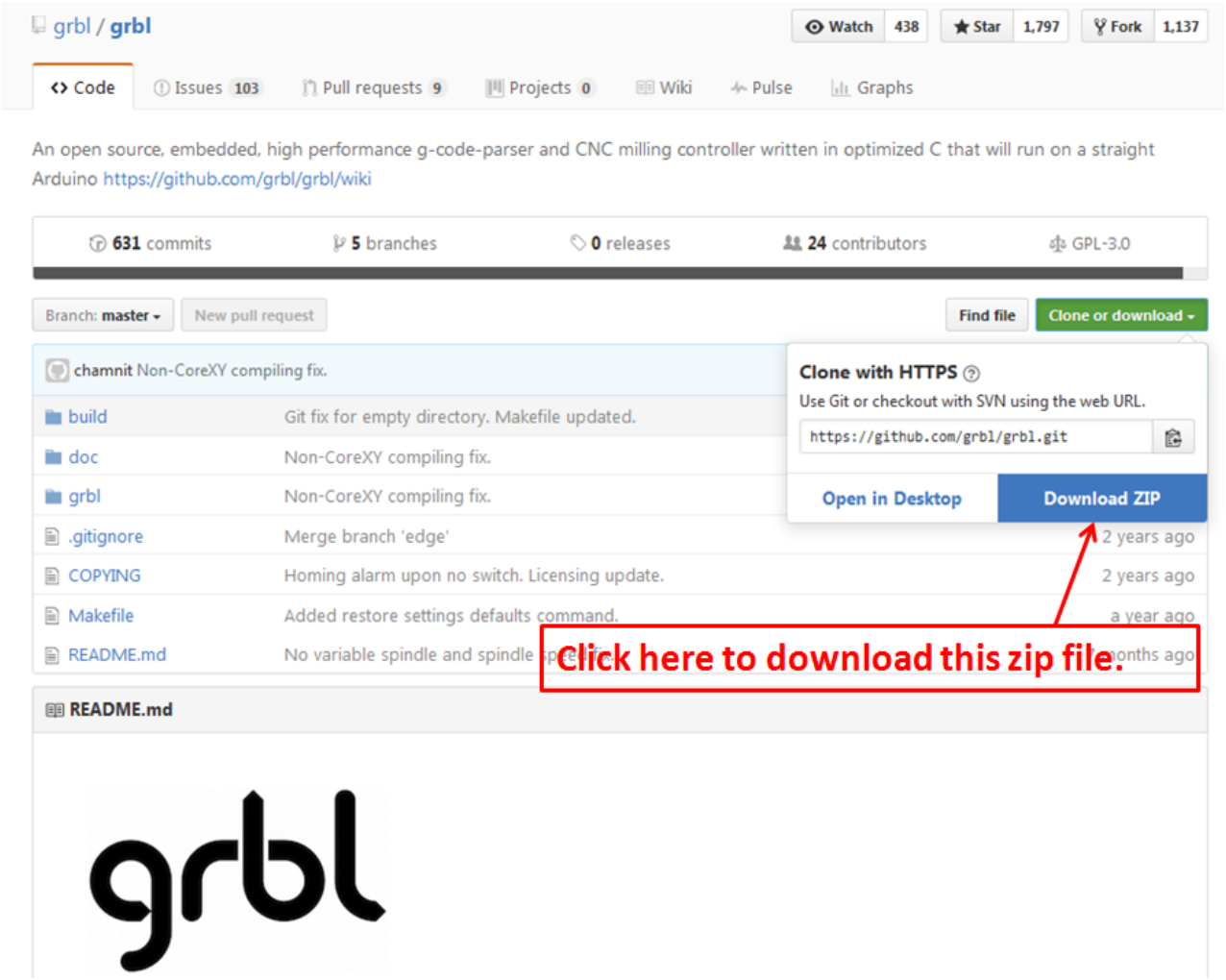
2. GRBL Control Software/Firmware for Arduino

Before you can use this CNC shield with Arduino, a control firmware need to be downloaded into Arduino board. We are going to use 'GRBL' to accomplish our job. GRBL is open-source software that runs on an Arduino Uno that takes G-Code commands via Serial and turns the commands into motor signals. Grbl is a no-compromise, high performance, low cost alternative to parallel-port-based motion control for CNC machine. It accepts standards-compliant g-code and has been tested with the output of several CAM tools with no problems. Arcs, circles and helical motion are fully supported, as well as, all other primary g-code commands. Macro functions, variables, and most canned cycles are not supported, but we think GUIs can do a much better job at translating them into straight g-code anyhow.

A copy of this open-source firmware can be downloaded from the below link:

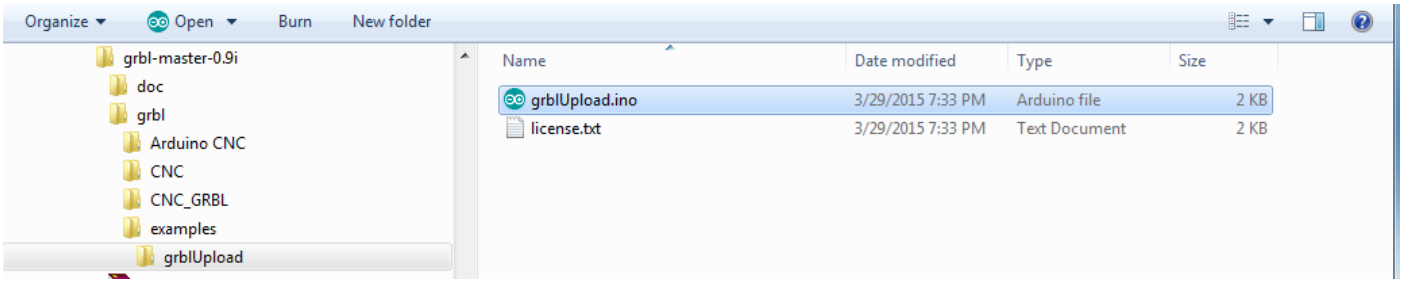
Following the below steps to prepare this CNC Shield board to function properly:

1. Download a copy of GRBL from: https://github.com/grbl/grbl

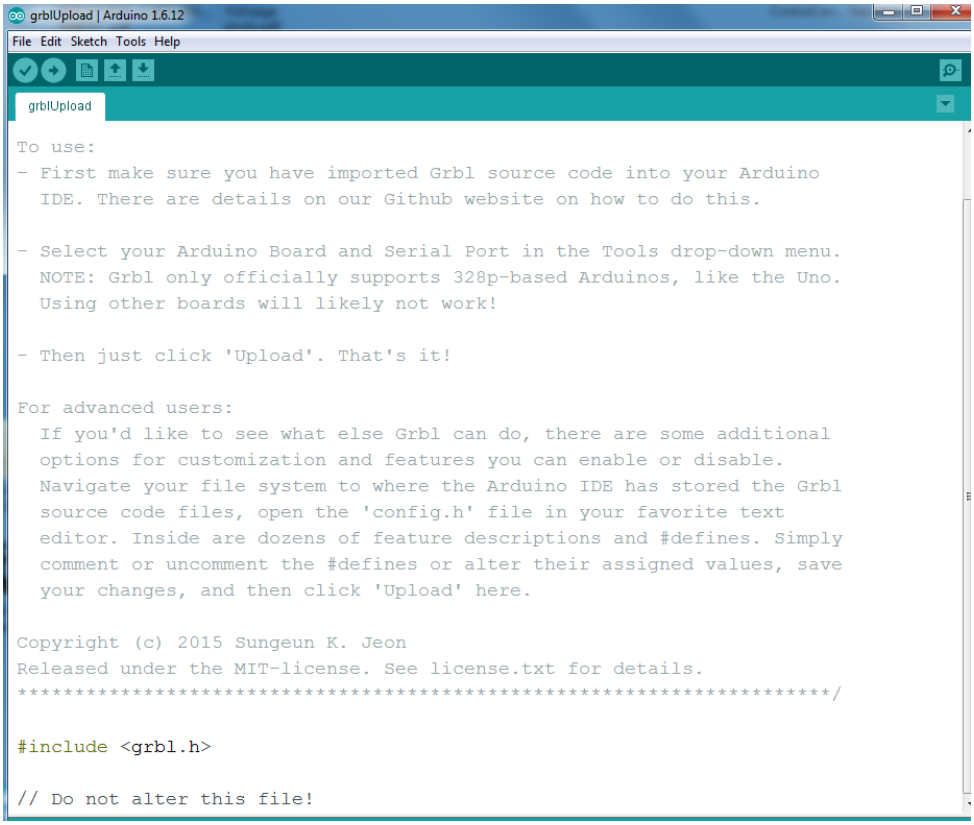


Unzip this file into your local hardisk location, you may want to create a special new folder for this purpose for easy locating of all files needed in your project. Locate an Arduino sketch '***grblUpload.ino***' in this folder where you have unzip the files.

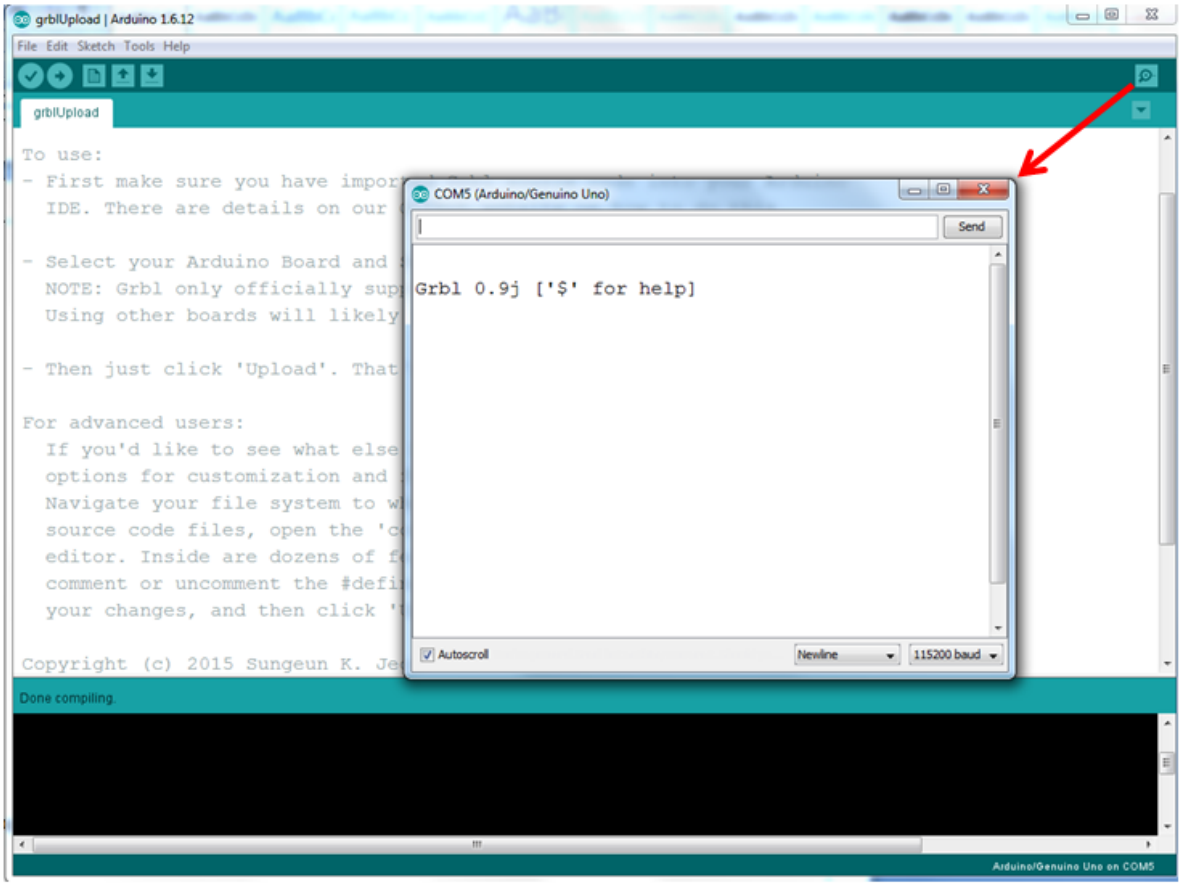
Below is the files structures located in my local hardisk:



Open up this sketch '***grblUpload.ino***' and you should see the screen as below:



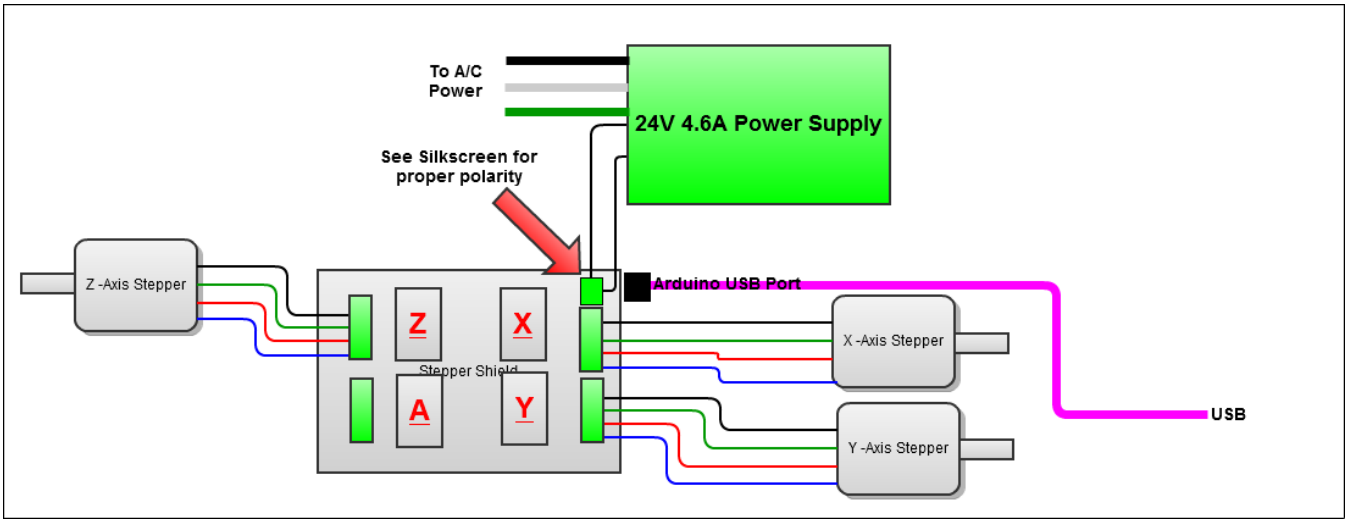
Click the upload icon as usual to ‘compile/upload’ as you normally upload Arduino sketch. When you see ‘done uploading’, click the ‘Serial Monitor’ on Arduino IDE as shown below:

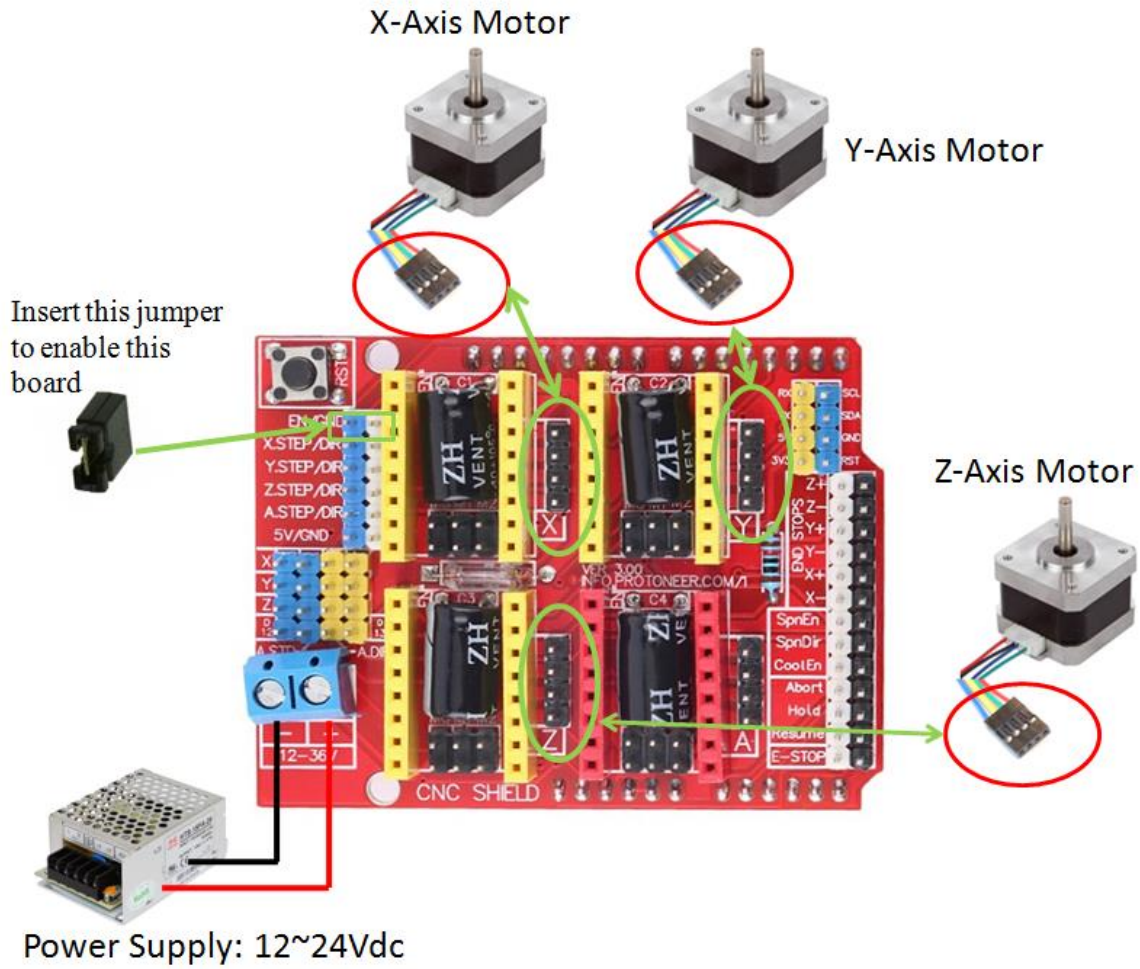


If you can receive response message " ***Grbl 0.9j ['$' for help]*** " from your Serial Monitor, congratulation! You have successful uploaded the 'GRBL' firmware into your Arduino board.

3. Hooking Up the Stepper Motor to CNC Shield

Connect steppers motor to CNC Shield board as the below block diagram. of the CNC Shield connected to 3-stepper motor:





Your CNC Shield board is now ready to go for a test run, let's try to turn the motor as to our instruction !!

4. G-Code Sender

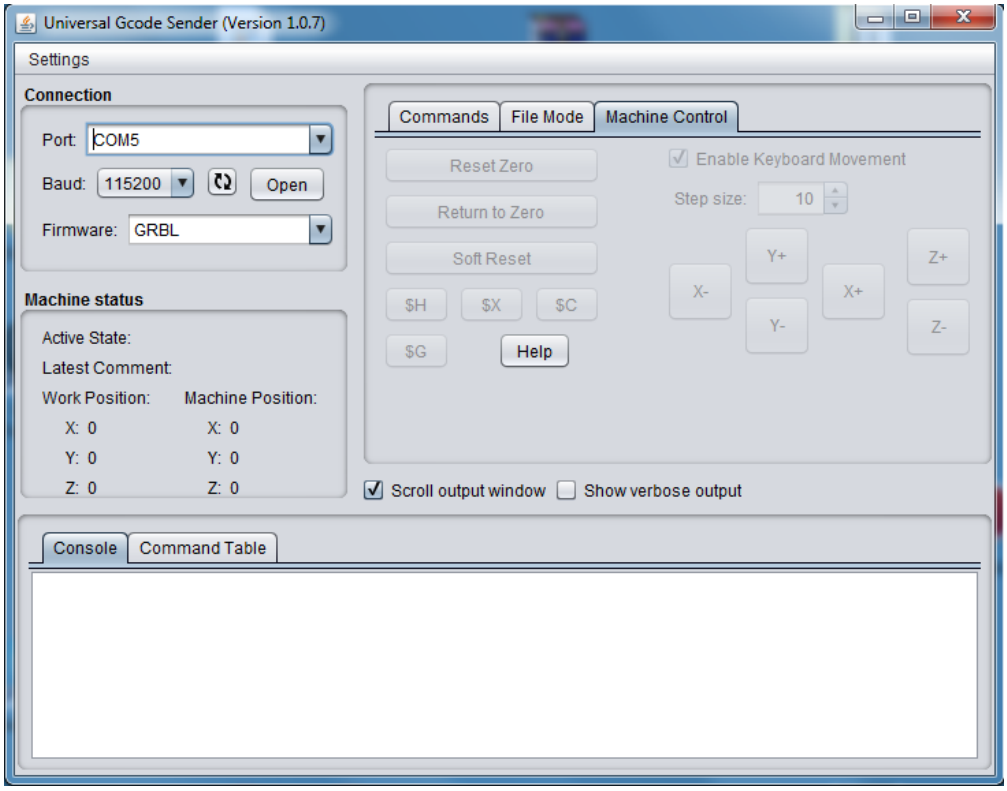
To send commands to your CNC/3-axis stepper motor driver board you need a g-code sender to send command and instruction.

Download the Universal-G-Code-Sender.

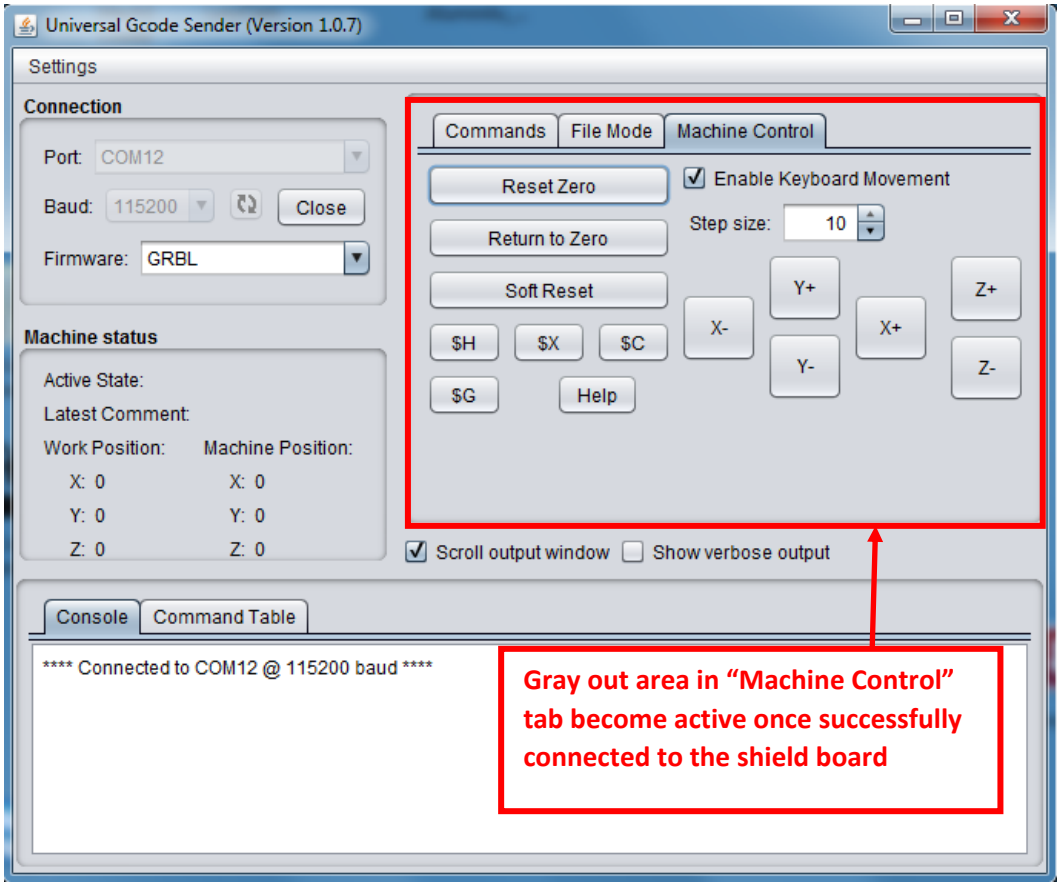
Universal GcodeSender is a Java based GRBL compatible cross platform G-Code sender. Use this program to run a GRBL controlled CNC machine. Once downloaded, connect the UNO board USB to your computer and take note of the COM port it is connected to.

Run the *start-windows.bat* or the *JAR* file directly.

You will be presented with the following screen:



Click the “Machine Control” tab. Select the “Port” number which your Arduino board is attached to. Select 115200 as the Baud rate as this is the speed configured in “GRBL” firmware. Click “Open” button to start the communication for controlling this shield board.



Now, you are ready to control the XYZ stepper motor connected to shield board. Click X+ or X- for example, the stepper motor attached to X terminal will turn in forward or reverse direction.

This will complete our initial setup for driving 3-axis stepper motor connected to this shield board with "GRBL" loaded to Arduino controller board using "Universal G-Code Sender" user interface.

5. Recommended Driver Board & Accessory

- CNC/3-Axis Stepper Motor Driver Shield for Arduino Board
- A4988 Stepper Motor Driver Module
- GT2 Pulley 5mm Bore
- 5mm to 8mm Motor Shaft Coupler

HandsOn Technology provides a multimedia and interactive platform for everyone interested in electronics. From beginner to diehard, from student to lecturer. Information, education, inspiration and entertainment. Analog and digital, practical and theoretical; software and hardware.

open source hardware

HandsOn Technology support Open Source Hardware (OSHW) Development Platform.

Learn : Design : Share

www.handsontec.com

The Face behind our product quality…

In a world of constant change and continuous technological development, a new or replacement product is never far away – and they all need to be tested.

Many vendors simply import and sell wihtout checks and this cannot be the ultimate interests of anyone, particularly the customer. Every part sell on Handsotec is fully tested. So when buying from Handsontec products range, you can be confident you're getting outstanding quality and value.

We keep adding the new parts so that you can get rolling on your next project.

Breakout Boards & Modules

Connectors

Electro-Mechanical Parts

Engineering Material

Mechanical Hardware

Electronics Components

Power Supply

Arduino Board & Shield

Tools & Accessory

Quick Reference

NEMA size 17 1.8° 2-phase stepper motor

IMS INTELLIGENT MOTION SYSTEMS, INC.

Schneider Electric

Notes and Warnings

Installation, configuration and maintenance must be carried out by qualified technicians only. You must have detailed information to be able to carry out this work.

- Unexpected dangers may be encountered when working with this product!
- Incorrect use may destroy this product and connected components!

For more information, go to www.imshome.com

Specifications

1.5 Amp motors		Single length	Double length	Triple length
Part number		**M-1713-1.5** • *(1)*	**M-1715-1.5** • *(1)*	**M-1719-1.5** • *(1)*
Holding torque	oz-in	32	60	75
	N-cm	23	42	53
Detent torque	oz-in	1.7	2.1	3.5
	N-cm	1.2	1.5	2.5
Rotor inertia	oz-in-sec^2	0.000538	0.0008037	0.0011562
	kg-cm^2	0.038	0.057	0.082
Weight	oz	7.4	8.1	12.7
	grams	210	230	360
Phase current	amps	1.5	1.5	1.5
Phase resistance	ohms	1.3	2.1	2.0
Phase inductance	mH	2.1	5.0	3.85

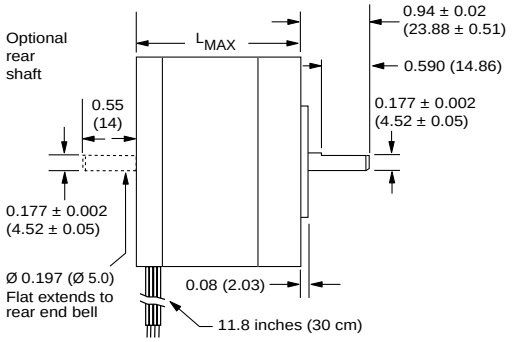
(1) Indicate S for single-shaft or D for double-shaft. Example M-1713-1.5S

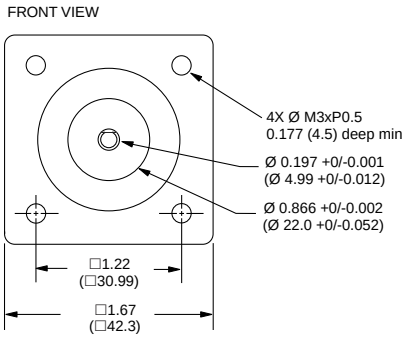
Wiring and Connections

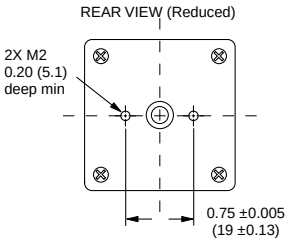
Signals and wire colors	
Phase A	Red
Phase /A	Blue
Phase B	Green
Phase /B	Black

Mechanical Specifications

Dimensions in inches (mm)







Motor stack length inches (mm)	Single	Double	Triple
LMAX	1.34 (34.0)	1.57 (40)	1.89 (48)

Part Numbers

Example:	M	-	1	7	1	3	-	1	.	5	S				
Stepper motor frame size **M-17** = NEMA 17 (1.7"/42 mm)	**M**	**-**	**1**	**7**	1	3	-	1	.	5	S				
Motor length **13-** = single stack **15-** = double stack **19-** = triple stack	M	-	1	7	**1**	**3**	1	.	5	S					
Phase current **1.5** = 1.5 Amps	M	-	1	7	1	3	-	**1**	**.**	**5**	S				
Shaft **S** = single, front shaft only **D** = double, front and rear shafts	M	-	1	7	1	3	-	1	.	5	**S**				
Optional optical encoder *(1)* **ES** = Single-end **ED** = Differential	M	-	1	7	1	3	-	1	.	5	**E**	**S**	**1**	**0**	**0**
Line count 100, 200, 250, 400, 500 or 1000 *(2)*															

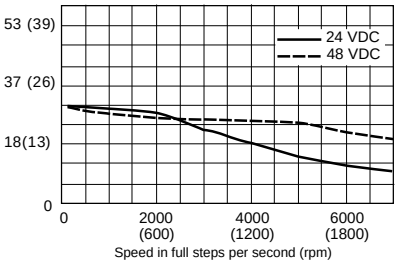
(1) An encoder replaces the shaft designator in the part number.
(2) All encoders have an index mark, except the 1000 line count version.

Torque-speed performance

Measured at 1.5 Amps RMS

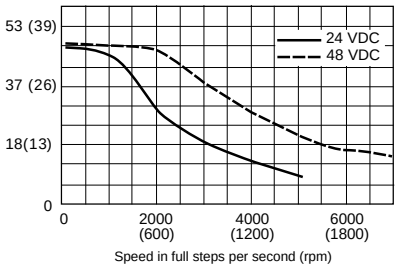
M-1713-1.5

Torque in oz-in (N-cm)



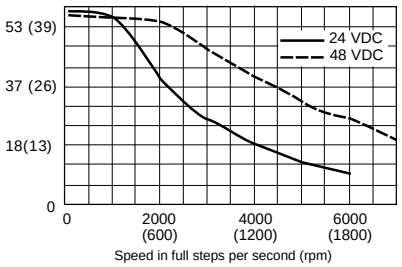
M-1715-1.5

Torque in oz-in (N-cm)



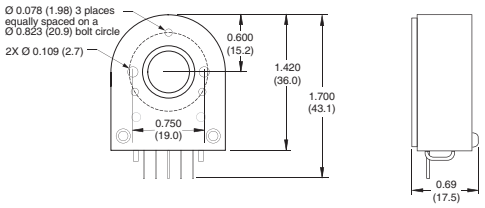
M-1719-1.5

Torque in oz-in (N-cm)



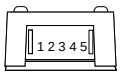
Optical Encoder Option

Dimensions in inches (mm)



Connectivity

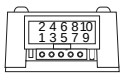
single-end encoder



	wire	*function*
1	Brown	Ground
2	Violet	Index
3	Blue	Channel A
4	Orange	+5 VDC input
5	Yellow	Channel B

optional interface cable available: ES-CABLE-2

differential encoder

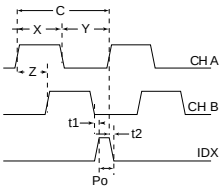


pin	*function*	*pin*	*function*
1	no connect	6	Channel A+
2	+5 VDC input	7	Channel B –
3	Ground	8	Channel B+
4	no connect	9	Index –
5	Channel A –	10	Index +

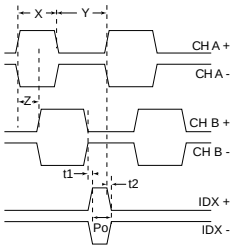
interface cable included

Timing

single-end encoder

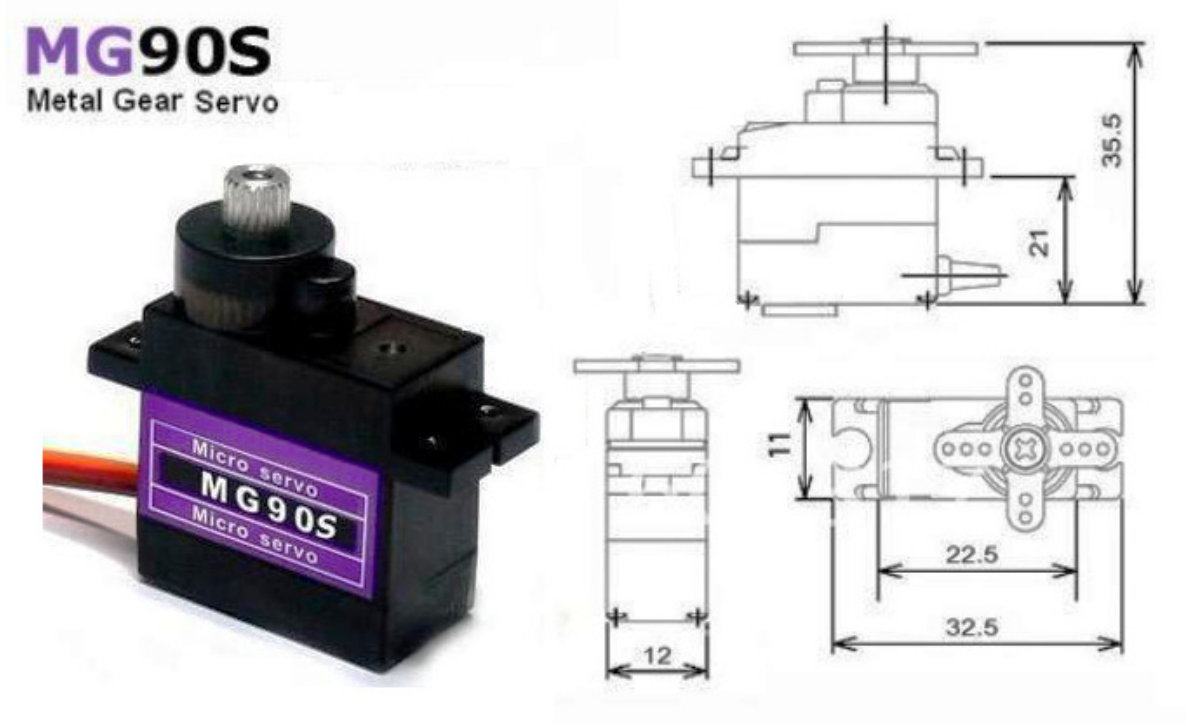


differential encoder



Parameter	Symbol	Min	Typ	Max	Units
Cycle error			3	5.5	°e
Symmetry		130	180	230	°e
Quadrature		40	90	140	°e
Index pulse width	Po	60	90	120	°e
Index rise (after Ch A or B rise)	t1	-300	100	250	ns
Index fall (after Ch A or B fall)	t2	70	150	1000	ns

C One cycle: 360 electrical degrees (°e).
X/Y Symmetry: the measure of the relationship between X and Y, nominally 180°e.
Z Quadrature: the phase lead or lag between channels A and B, nominally 90°e.
Po Index pulse width, nominally 90 °e.
NOTE: Rotation is as viewed from the cover side of the encoder.



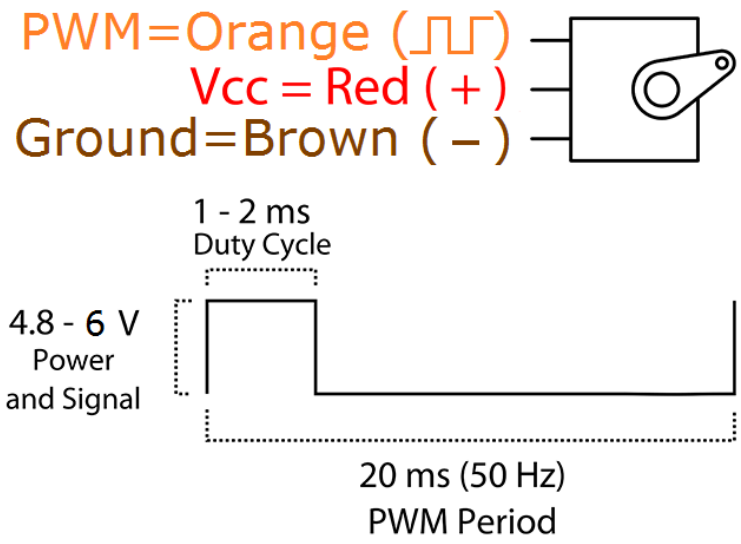
MG90S servo, Metal gear with one bearing

Tiny and lightweight with high output power, this tiny servo is perfect for RC Airplane, Helicopter, Quadcopter or Robot. This servo has *metal gears* for added strength and durability.

Servo can rotate approximately 180 degrees (90 in each direction), and works just like the standard kinds but *smaller*. You can use any servo code, hardware or library to control these servos. Good for beginners who want to make stuff move without building a motor controller with feedback & gear box, especially since it will fit in small places. It comes with a 3 horns (arms) and hardware.

Specifications

- Weight: 13.4 g
- Dimension: 22.5 x 12 x 35.5 mm approx.
- Stall torque: 1.8 kgf·cm (4.8V), 2.2 kgf·cm (6 V)
- Operating speed: 0.1 s/60 degree (4.8 V), 0.08 s/60 degree (6 V)
- Operating voltage: 4.8 V - 6.0 V
- Dead band width: 5 µs



Position "0" (1.5 ms pulse) is middle, "90" (~2 ms pulse) is all the way to the right, "-90" (~1 ms pulse) is all the way to the left.

Bibliography

[1] P. A. S. da Rocha, R. D. de Silva e Souza, and M. E. de Lima Tostes. "Prototype CNC machine design". In: *2010 9th IEEE/IAS International Conference on Industry Applications - INDUSCON 2010*. 2010, pp. 1–5.

[2] Kajal J.Madekar, Kranti R. Nanaware,Pooja R. Phadtare, and Vikas S. Mane. "Automatic mini CNC machine for PCB drawing and drilling". In: *International Research Journal of Engineering and Technology (IRJET)* 3.2 (Feb. 2016).

[3] SATHYAKUMAR. N. *A Build-Your-Own Three Axis CNC PCB Milling Machine*. Teaching Learning Centre for Design, Manufacturing Education, Indian Institute of Information Technology, Design, and Manufacturing-Kancheepuram. July 2016.

[4] *ARDUINO UNO REV3*. URL: https://store.arduino.cc/usa/arduino-uno-rev3.

[5] *Arduino UNO + Arduino CNC Shield V3.0+A4988 Installation Guide*. Apr. 2017. URL: https://osoyoo.com/2017/04/07/arduino-uno-cnc-shield-v3-0-a4988/.

[6] Dan Rock. *VREF adjustment A4988*. URL: https://e3d-online.dozuki.com/Guide/VREF+adjustment+A4988/92?lang=en.

[7] Peter. *The difference between unipolar and bipolar stepper motors*. June 2018. URL: https://techexplorations.com/blog/arduino/blog-the-difference-between-unipolar-and-bipolar-stepper-motors/.

[8] Zen Toolworks. *Arduino Grbl CNC Tutorial - 01*. Feb. 2019. URL: https://www.youtube.com/watch?v=diqG9fbcuMs.

[9] David L. Jones. *PCB Design Tutorial*. second. David L. Jones, June 2004.

Acknowledgements

We are thankful to our college Vivekanand Education Society's Institute of Technology for considering our project and extending help at all the stages needed during our work of collecting information regarding our project.

It gives us immense pleasure to express our deep and sincere gratitude to Dr.(Mrs.) Asawari Dudwadkar (Project Guide) for her kind help and valuable advice during the development of project synopsis and for her guidance and suggestions.

We are deeply indebted to Head of the Electronics Department Mrs. Kavita Tiwari and Principal Dr. (Mrs.) J. M. Nair for giving us this valuable opportunity to do this project. We express our hearty thanks to them for their assistance without which it would have been difficult in finishing this project and project review successfully.

We convey our deep sense of gratitude to all the teaching and non-teaching staff for their constant encouragement, support and selfless help throughout the project work. It is our great pleasure to acknowledge the help and suggestions, which we received from the Department of Electronics Engineering.