Rapid Prototyping of Rapid Prototyping Machines

By

Ilan Ellison Moyer

Submitted to the Department of Mechanical Engineering in
Partial Fulfillment of the Requirements for the
Degree of

Bachelor of Science

at the

Massachusetts Institute of Technology

May 2008

Signature of Author: _____
Department of Mechanical Engineering
May 13th, 2008

Certified by: _____
Neil A. Gershenfeld
Professor of Media Arts and Sciences
Director, Center for Bits and Atoms
Thesis Supervisor

Accepted by: _____
John H. Lienhard V
Professor of Mechanical Engineering
Chairman of the Undergraduate Thesis Committee

Rapid Prototyping of Rapid Prototyping Machines

By

Ilan Ellison Moyer

Submitted to the Department of Mechanical Engineering
On May 15th, 2008 in Partial Fulfillment of the
Requirements for the Degree of Bachelor of Science in
Mechanical Engineering

ABSTRACT

Rapid prototyping tools empower individuals to create almost anything. Unfortunately, these tools are still far too expensive for personal ownership. The do-it-yourself community has responded with a slew of home-made rapid prototyping machines, but development times are slow because of the complexity of the necessary control system and the need to design the mechanical elements from scratch. This thesis seeks to address both of these issues. A control system is developed which treats the machine as a distributed Internet Zero network controlled by a software virtual machine with the benefits of simplified configuration and greater flexibility. A low cost circuit board milling machine, built as the test bed for this distributed controller, is described in detail. Finally, a parametrically designed XY table is introduced as a prototype for a universal machine axis and a first step towards the creation of reusable machine designs. These contributions will hopefully aid in accelerating the development of new rapid prototyping machines.

Thesis Supervisor: Neil Gershenfeld
Title: Professor of Media Arts and Sciences

ACKNOWLEDGEMENTS

I would like to thank Professor Neil Gershenfeld for allowing me to share in the incredibly exciting experience of being part of his research group for the past year, and for providing me with unerring guidance over the course of this thesis. It is always exciting to start on a project slightly dubious of its direction and to then become more and more excited as the full picture becomes clear. Thanks also to David Kopp of Schnieder Electric for his mentorship and friendship, and his technical support with the Internet Zero components of this project. Kerry Lynn of Cisco Systems is the reason I joined the Physics and Media group originally, and I thank him for his continued support since that pivotal day. Also, I would like to acknowledge all the Physics and Media graduate students for their willingness to provide help regardless of their workload or the time of day or night.

A special thanks to Ken Stone at the MIT Hobby Shop for his friendship throughout my tenure at MIT, and of course the constant guidance he has provided me on nearly every project I've been involved with these past four years. The Hobby Shop has been an invaluable resource during my time at MIT and has made all the difference. Much of the hardware for this thesis project was fabricated at the MIT Hobby Shop.

Finally, I would like to thank my parents for their love and support these past four years... and the 18 years prior as well.

TABLE OF CONTENTS

Abstract..... 2

Acknowledgements 3

Introduction 6

Background 8

A Virtual Machine Controlling a Low Cost Circuit Board Milling Machine over an IP Network 14

- Control System Problems 14
- A Virtual Machine and Distributed Network Architecture 14
- Design Overview 16
- Chassis 18
- X and Y Axis Overview 19
- X and Y Axis Guides 19
- Virtual Guides 20
- Error Mapping Using Virtual Guides and Feedback Loops 21
- X and Y Axis Transmissions 21
- Z Axis Overview 26
- Z Axis Guide 27
- Z Axis Transmission 28
- Virtual Transmissions 30
- Virtual Motors 31
- Virtual Axes – a Recap 31
- Table 31
- Spindle 31
- Distributed Controller Network 33
- Virtual Stepper Motors 37
- Virtual Motor Controller 38
- Concluding Remarks 39

A Parametrically Designed XY Motion Stage..... 40

- Introduction 40
- Design Overview 40
- Axis Overview 42

Guides 42
Transmission 43
Towards a Reusable Design 44
Concluding Remarks 48
Conclusions 49
References 50

INTRODUCTION

Rapid prototyping has gained widespread industrial acceptance as a means of quickly and economically producing small quantities of physical objects. In addition to its commercial applications, rapid prototyping tools have the potential to drastically influence the ways people create and their reasons for doing so. Digital fabrication promises individuals means of creating complex objects with virtually no prerequisite skill. This not only fosters creativity by removing the time-cost of taking design risks, but also promises freedom from the one-size-fits all paradigm of mass-production. If it is easy to make something which exactly fits your needs, why settle for a sub-optimal commercial product?

The bottleneck which prevents individuals from reaping the fruits of modern rapid prototyping technology is its high cost of commercially available tools. Prices range from the thousands to the hundreds of thousands of dollars, depending on the type of machine and its capabilities. In response to this market void, many hobbyists have decided to build their own – a task much easier said than done. The control systems available today are tough to configure and limited in their flexibility. Also, the from-scratch mechanical design of a rapid prototyping machine can prove daunting to those without a formal mechanical engineering education. Unlike the extensive code libraries which programmers use to share their efforts with others, machine designs are typically done in a way which is not reusable. These difficulties conspire to ensure that the process of making a rapid prototyping machine is anything but rapid.

This thesis seeks to alleviate the some of the hardship which has retarded the development of new rapid prototyping machines. First, a new control system paradigm is introduced which makes it easy to configure and expand the capabilities of a machine. Instead of being rigidly defined by the control system, the machine tool is treated as a distributed network of addressable internet nodes which receive commands from a virtual machine running on a computer. The Internet Zero (I0) protocol, developed at the MIT Center for Bits and Atoms, is used to establish this network. Figure 1 on the following page shows an entire linear motion axis complete with a stepper-motor actuated capstan drive system and an embedded I0 motor controller. This is part of a low cost PCB mill which was designed and built as a test bed for the distributed network and virtual machine control system.

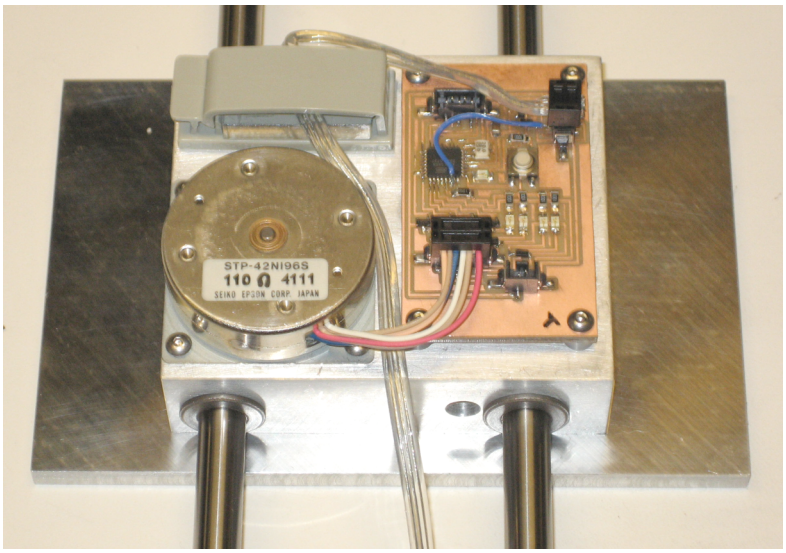


Figure 1: Linear Motion Stage with an Integrated Networked Controller

A parametrically designed XY axis was also developed as an exploration into the feasibility of reusable mechanical design. Instead of needed to design rapid prototyping machines from scratch, parametric design software can be used to capture the logic behind a design. The idea is to allow flexibility in adjusting engineering parameters such as travel and stiffness without any additional design effort.

It is the hope of the author that contributions of this thesis, namely a virtual machine controlling a real machine over a distributed network, and a prototype of reusable mechanical design, will aid the future rapid prototyping machine builder in both accelerating and relieving some of the hardship of their task. The faster machines can be built, the sooner the rapid prototyping can begin!

BACKGROUND

The drive to create is the inescapable human characteristic which has propelled us from the stone age to a modern era saturated with technology. Along the way, the paradigm of commercially "making things" has changed. What used to be a personal pursuit carried out by skilled craftsmen is now the vastly coordinated and segmented effort of mass production. This shift has improved many aspects of life, and is in many ways necessary to support the growing complexity of modern products. Much more design effort and capital can be invested in creating an object when the cost is distributed among millions of units. Unfortunately, the same aspects which empower mass production have had a negative impact on the creative powers vested in individuals. Making things on a professional level *demands* expensive equipment and knowledge generally conferred only by an engineering degree. This is of course a relative assessment. Anyone with woodworking equipment and skill can make professional tables, just as they were made 100 years ago. But asking a person to single-handedly produce a laptop computer is currently beyond reasonable expectations.

However, the desire to create – an impulse of the individual mind– is as strong as ever. And the unsatisfying one-size-fits-all philosophy of mass production has only amplified the urge. An exciting new set of technologies developed for industry is slowly being hijacked for this purpose.

Companies in the development phase preceding mass production and the individual maker face similar issues. Before committing to producing a million copies of a design, it is imperative that small quantities of prototypes are generated and validated. Production machinery, whose operation relies upon economies of scale, is impractical for the task. Thus was born the field of *rapid prototyping* (RP). While the term typically evokes mental images of three-dimensional printers, the underlying spirit can be expressed simply: the automated creation of a physical object from a digital representation. This definition implies a process, which is depicted in Figure 2.

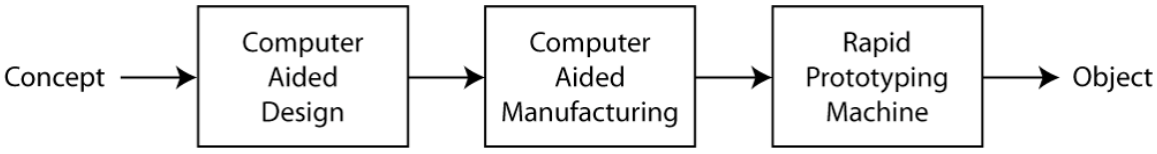


Figure 2 : The Rapid Prototyping Process

The first step in the process is creating the digital (i.e. mathematical) representation of a concept. This is accomplished using a computer software package known as a *computer aided design* (CAD) tool. Most modern CAD

tools include fancy visualization features which render the object exactly as it will appear in real life.

In order to understand the middle step, *computer aided manufacturing* (CAM), it is first necessary to skip ahead to the final step in the process. Rapid prototyping machines interact with physical matter in either an additive or subtractive manner to achieve a desired shape. Many tools have been created towards this purpose, each with a different modus operandi. CNC milling machines such as the Hass MiniMill shown in Figure 3a uses a rotating cutter to remove material from a piece of stock. Smaller versions like the Roland Modela (Figure 3b) can be used to create circuit boards by selectively removing the copper cladding of copper-laminated PCB material. Laser cutters (Figure 3c) and waterjet cutters (Figure 3d) utilize focused light energy and high pressure streams of water, respectively, to trace two-dimensional shapes out of sheets of material. The waterjet cutter in particular can cut through 8 inches of hardened steel plate. The quintessential rapid prototyping machine is the three-dimensional printer. This class of machine builds up an arbitrary 3D shape one layer at a time. The way each layer is build varies across several available technologies: *stereolithography* (SLA) uses a laser to harden liquid polymer, *fused deposition modeling* (FDM) deposits streams of plastic into the desired pattern, and inkjet-style 3D printers selectively squirt material where it is needed (Figure 3e).

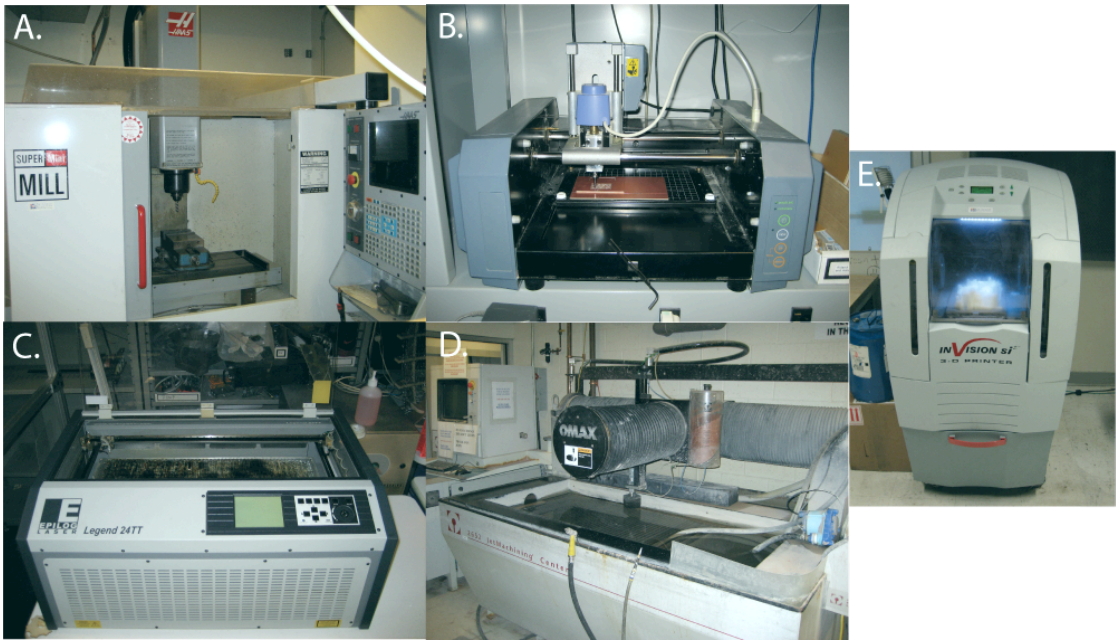


Figure 3: A Posse of Rapid Prototyping Machines: A.) CNC Milling Machine, B.) Desktop Modeler, C.) Laser Cutter, D.) Waterjet Cutter, E.) 3D Printer

The operation of every rapid prototyping machine is dependent on a single technology, *computer numerical control* (CNC), to electronically control the relative position between its "business end" and a work-piece. It is here that computer aided manufacturing comes into play. Once a part has been described mathematically, a software CAM package couples this information with knowledge of the machine to generate a sequence of moves which will result in the desired physical shape. These moves are interpreted and acted on by the machine's CNC controller.

While the rapid prototyping field conceptually answers the prayers of the creative individual, machines on the market today are priced out of reach. The least capable of the previously described RP machines, the Roland Modela, costs $5,000. Laser cutters and 3D printers go for $20,000, larger CNC routing machines for $10,000, and waterjet cutters cost $150,000. One solution, which has been developed by the MIT Center for Bits and Atoms, is the worldwide establishment of creative centers known as *Fab Labs* [1]. These facilities, one of which is shown in Figure 4, provide members with access to many of the RP tools mentioned above. Though the program is growing rapidly, it is unable to even approach meeting the demand.

Figure 4: Fab Lab Boston

Another solution which has found popularity among the DIY community is, unsurprisingly, home-brewed rapid prototyping machines. What exactly is entailed in this endeavor? Figure 5 presents a generalized model of the traditional DIY machine architecture.

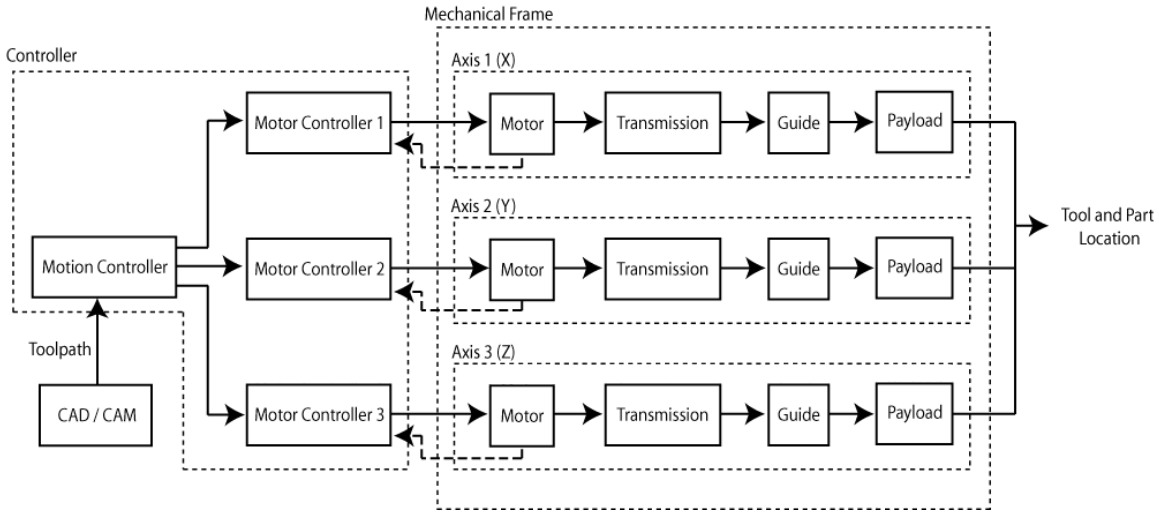


Figure 5: A Generalized DIY Machine Architecture

Just like their commercial brethren, DIY RP machines function according to one overarching principle: precisely moving their effector, be it a polymer nozzle or a spinning cutter, with relation to the part being fabricated. To accomplish this a series of axes are required – each responsible for a single vector of motion. Most RP machines utilize three orthogonal linear axes, though more advanced tools might require rotational axes as well. An axis is composed of four components: a payload, a guide, a transmission, and a motor. The payload is what is being moved; this could be a table supporting the part, an effector such as a spindle or nozzle, or even another axis. The guide provides mechanical support to the payload and constrains its movement along the vector of the axis. Typical guides include precision-ground rod, linear bearing rails with recirculating ball bearings, and dovetailed ways. Motors are the source of mechanical power which moves the payload. The most common motors found on DIY machines are the economical rotary stepper motors, which move in precise rotational increments in response to electrical activation of their armatures. The task of converting the rotational motion of the motor into movement of the payload along the guide is performed by the transmission. Transmission elements for linear axes include leadscrews, ballscrews, capstan drives, and rack-and-pinion drives. Each type of transmission has pros and cons in terms of force and resolution amplification, maximum load, backlash, mechanical efficiency, and cost.

The control system of a RP machine is responsible for interpreting and acting upon movement (i.e. position and feedrate) commands generated by CAM software. This is typically accomplished in two steps. First, a motion controller decomposes each movement command into its components along the axes of the machine. In the case of stepper motors, these decomposed commands are then further broken up into a series of synchronized step and

direction commands which are sent to the individual motor controllers, one per axis. The motor controllers then apply electrical current to the armatures of the axis to achieve the desired movement. Motion controllers may either be stand-alone units or software packages running on a personal computer. Motor controllers are always external units owing to their need to source large amounts of electrical power, and may come with several controllers per box. A popular open-source motion controller program is furnished by the *Enhanced Machine Controller* (EMC) project, while stepper motor controllers are available from many hobby and industrial suppliers.

It should now be apparent that a broad scope of knowledge is necessary to design a rapid prototyping machine. The mechanical design alone requires either an engineering degree or extensive experience to pull off optimally. The control system is slightly more modular, but also necessitates in-depth research and quite a bit of parts hunting. Both aspects present bottlenecks to the development of new DIY RP machines.

In the spirit of the open source movement, several collaborative projects have sprung up online which seek to pool knowledge, skills, and time towards the design of DIY rapid prototyping tools. One shining example is the Fab@Home project, whose 3D printing “fabbers” (shown in Figure 6) are capable of producing arbitrarily shaped forms out of plastic and can be assembled by anyone with a laser cutter.

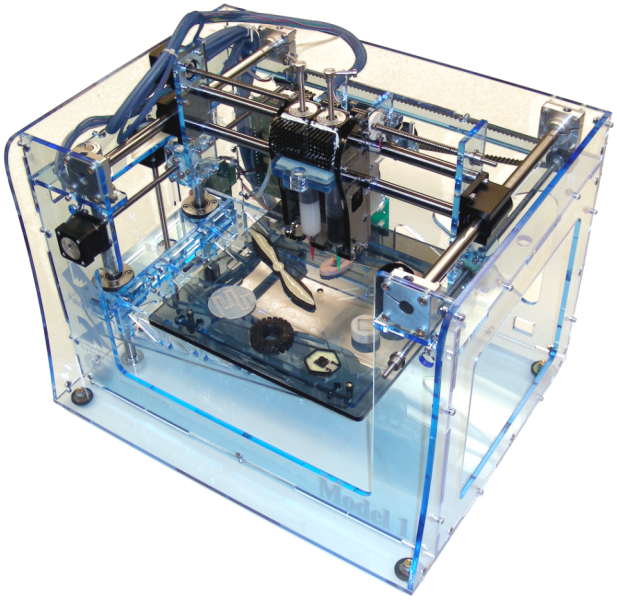


Figure 6: The 3D Printing "Fabber" from the Fab@Home Project

While projects like Fab@Home certainly benefit the DIY community by yielding designs for RP machines, they do not fix the underlying problems posed by the difficulty of designing a machine oneself. After all, a machine which is ideal for the needs of one individual might not satisfy those of another. It is ironic that the bottleneck restraining the proliferation of rapid prototyping tools is the same problem which those machines seek to alleviate in the aforementioned one-size-fits all market of mass-production.

The following pages seek to answer these problems by presenting new means of expediting the process of making rapid prototyping machines.

A VIRTUAL MACHINE CONTROLLING A LOW COST CIRCUIT BOARD MILLING MACHINE OVER AN IP NETWORK

Control System Problems

Perhaps the most daunting aspect of building a rapid prototyping machine is the control system. A typical control system has been briefly described and modeled schematically in the background section. Several issues make this standard architecture sub-optimal for machine development:

- The motion controller must be carefully configured with the parameters of the machine.
- The motion controller executes rigidly coded routines which cannot be easily modified by the user.
- Expensive interface cards are often necessary between a computer running motion control software and stepper motor controllers.
- The number of motors is limited.
- Accessory inputs and outputs, such as those used for hand encoder wheels or limit switches, are pre-prescribed and limited.
- Large bundles of cables must be run between the stepper motor controllers and their respective axes.

A Virtual Machine and Distributed Network Architecture

A new control system paradigm has been designed and is presented on the next page as Figure 7. The primary differences between this architecture and that of Figure 5 are the use of a virtual machine as the controller and the means of communication between the virtual and physical machines. Just as a rapid prototyping machine is built from components, a virtual machine is constructed from modules of code, or objects, which behave like the components of the real machine. For example, the guides of a machine's axes constrain the motion of the payload to a specific direction. Virtual guides contain a mathematical vector which returns to the controller the three-dimensional movement of the payload in response to linear motion along the guide. It is even possible that the same CAD software which generates real components will one day simultaneously generate "virtual" components. By building the machine virtually in the controller, configuration is automatic. And by coding the controller in an easy-to-learn programming language like Python, modifications are easily implemented.

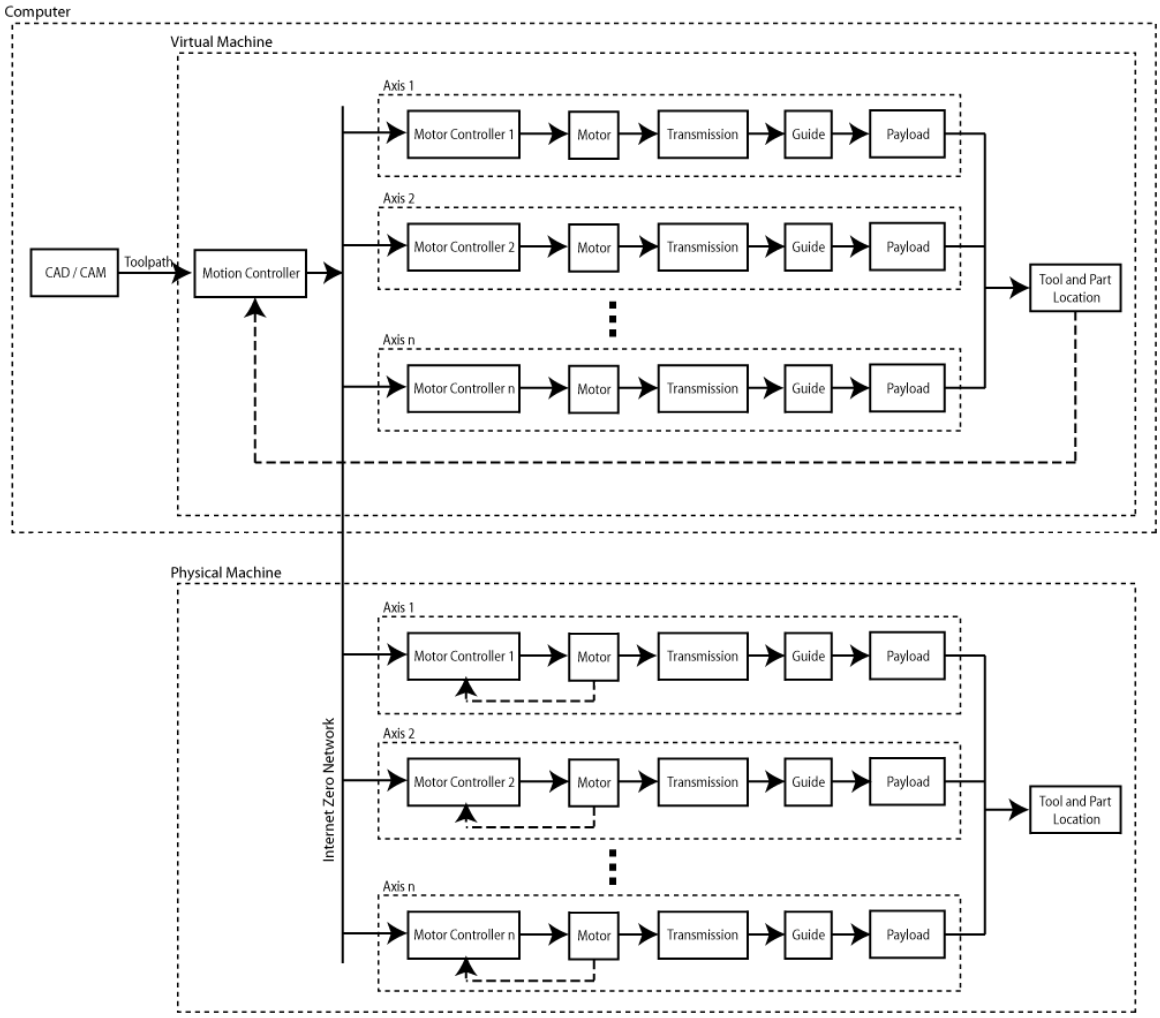


Figure 7: Virtual Machine Controller Architecture

The connection between the real and virtual machines is established using internet protocol over an Internet Zero (I0) link developed at the MIT Center for Bits and Atom [2]. Internet Zero is an extremely low cost (~$2/node) communications system which uses timed electrical pulses to transmit data.

Each motor controller has its own I0 node with a unique IP address. If the motor controller is mounted directly to the axis it controls, only three wires need to be run to the controller: two for power and one for data. Figure 8 is a photograph of an I0 stepper motor controller which was manufactured using a Roland Modela RP tool in a Fab Lab.

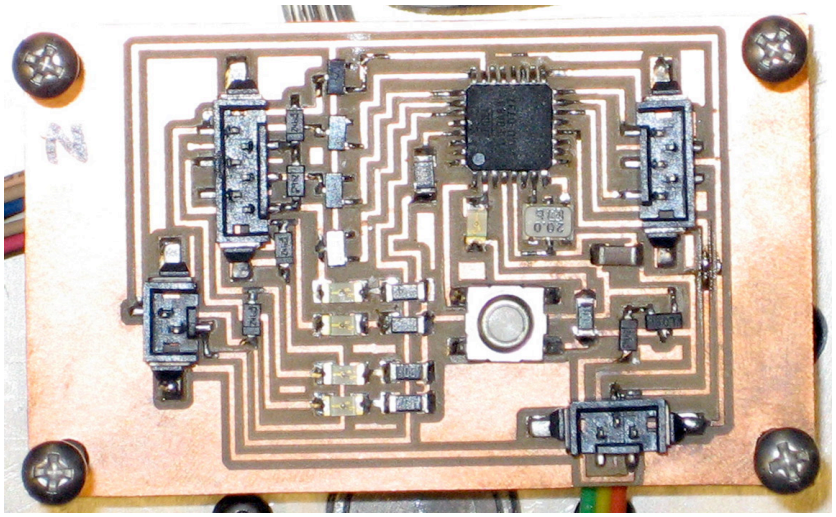


Figure 8: Internet Zero Stepper Motor Controller

Because the connection to the control system is over a network, an arbitrary number of nodes can be attached. This permits adding as many accessories as is desired, where each accessory has a virtual analog in the controller which specifies its behavior.

Design Overview

A low cost circuit board milling machine has been designed and constructed as a test bed for the virtual machine controller. Due to the intimate conceptual connection between the virtual machine and the real machine, both will be described in parallel.

The overarching principle driving the mechanical design of this particular rapid prototyping machine is cost – both in terms of the necessary components and the time required for its manufacture. Commercially available circuit board milling machines cost upwards of $2000, while the present design can be had for around $150 worth of parts and material. What made this possible was defining from the get go the bare minimum requirements of a circuit board milling machine. The following observations were made:

1. X and Y axis positional resolutions are not critical – 0.001" is sufficient.
2. X and Y axis positional accuracy and repeatability IS important for creating thinly spaced traces.
3. Cut depth is not critical because the process of milling traces is inherently two-dimensional. This implies that alignment between all

three axes is not important – particularly in directions which couple with tool height.

4. The use of small end mills (~1/64" - 1/32") at high spindle speeds (~10KRPM) generates very small cutting forces. The stiffness of the machine can be assigned accordingly.

These observations and careful analytical design have resulted in the machine shown in Figure 9.

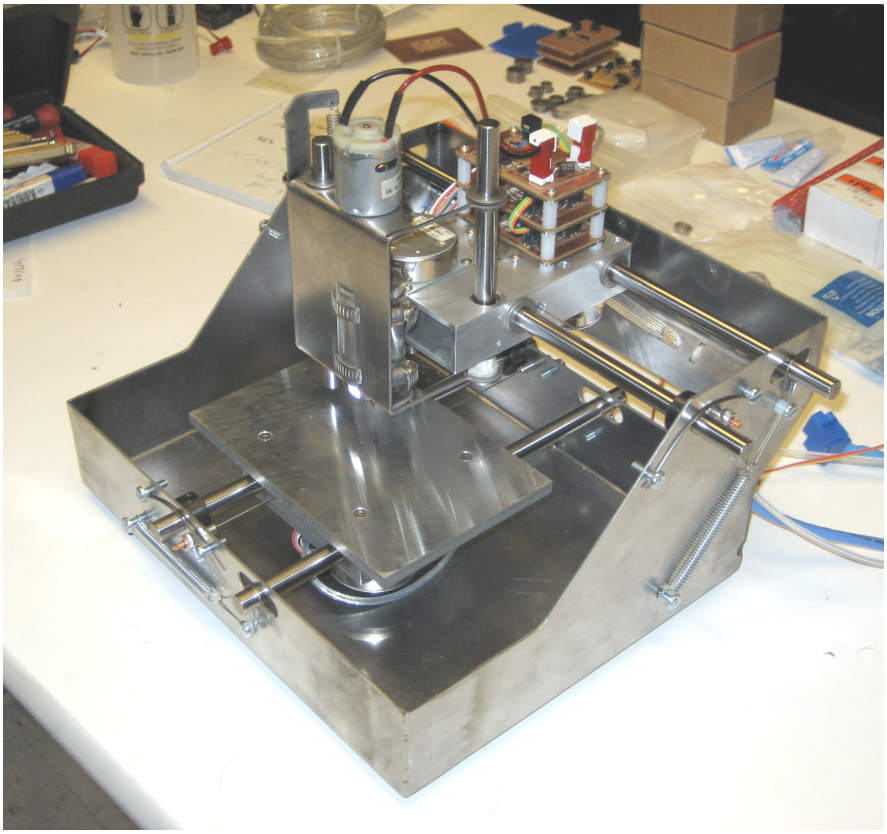


Figure 9: A Low Cost Circuit Board Milling Machine

Several themes should be noted. First, bent sheet metal is used in all of the structural components of the machine (i.e. chassis and Z axis frame) because of its low cost and ease of mass-production. Box extrusion, which lends itself nicely as a bearing support, is utilized for both the x and y axis carriages. Figure 10 presents a CAD model of the machine with its major features labeled.

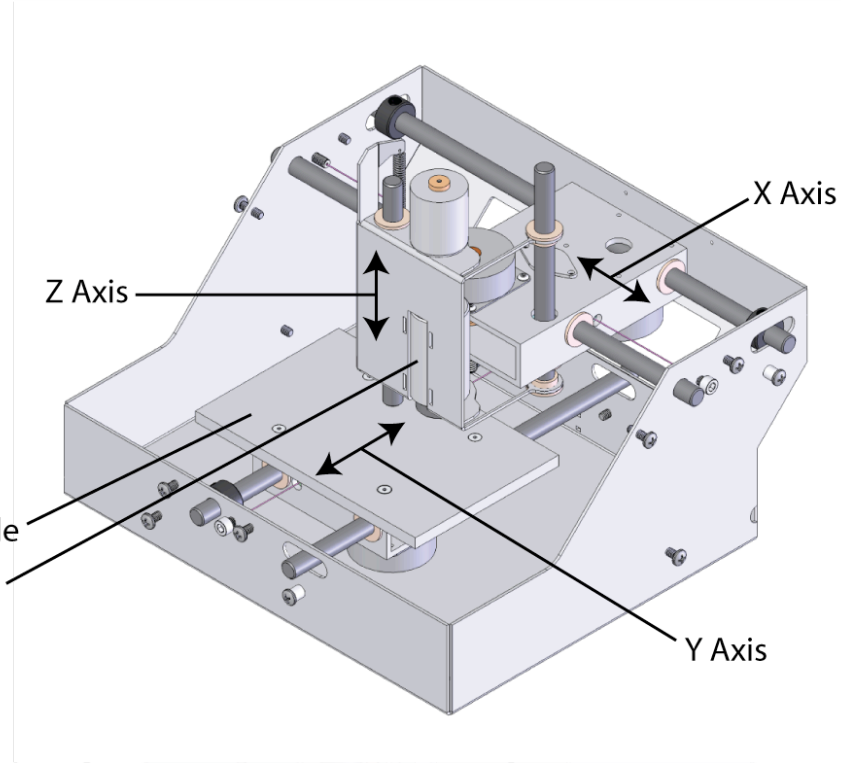


Figure 10: Axes of Motion

An 1/8" shank carbide tool is mounted in the spindle, and a copper-clad board is adhered to the table using double-stick tape. The rest of the machine serves to precisely control the relative position of these two elements.

Each machine element will now be described in detail, along with its accompanying virtual machine.

Chassis

The mechanical frame of the machine is constructed from a single bent piece of 0.06" thick mild steel. The prototype chassis was cut on a waterjet similar to that shown in Figure 3d and bent on a hand brake, but a mass-produced version would be stamped and CNC bent. The corners were also welded to increase rigidity. Interested hobbyists who find themselves without a waterjet cutter could still use a band saw and drill press to cut out the frame, and an inexpensive brake to perform the required bending. Advantages of this chassis design are the low cost and minimal number of parts. The major downside is the inaccurate nature of the bending process. Because both axes rely on the chassis to establish their vectors of motion, bending errors will affect the mechanical accuracy of the machine. Fortunately, however, the only misalignments which are influenced by bending accuracy are in a

direction coupled to the Z axis. It has already been discussed that maintaining a consistent distance between the tool and the part is not critical for the process of circuit board milling.

X and Y Axis Overview

The X and Y axes are identical in principle. Both are made of aluminum box extrusion carriages gliding on a pair of precision-ground steel shafts. A capstan-and-cable drive system is powered by stepper motors mounted directly on the moving carriage. The internet zero control boards are also mounted on the carriage, meaning that only three wires need to be run to the stationary chassis per axis. High flex ribbon cable is used for this purpose. A cast and fly-cut aluminum work table is mounted to the Y axis carriage. The X axis box extrusion overhangs its bearing rods on one side and is integral to the Z axis assembly.

X and Y Axis Guides

The X and Y axis guides of the machine consist of 3/8" diameter precision-ground and hardened steel shafts with enough length to permit a work envelope of 4" x 6". This working area was chosen because it matches a commonly available size of copper-clad board and is adequate for most projects. The chosen diameter gives an overall stiffness between the tool and table of roughly 800kN/m, or 0.0005" at a cutting thrust force of 2 lbs. Unlike the XY stage of the previous section, these two axes are decoupled from each other. The configuration depicted by Figure 10 significantly simplifies the design by establishing geometric and functional symmetry between the axes.

Riding on the rails is a carriage made from aluminum box extrusion. Each rail passes through two bronze bushings which are press fit into the extrusion.

While the purpose of a guide is to constrain motion along a vector, machine designers quickly learn that it is possible for a guide to provide *over constrai*nt. Generally speaking, over constraint occurs when an object has less degrees of freedom than constraints. When applied to linear guides, poor manufacturing tolerances may lead to over constraint if both rails attempt to guide the carriage along different vectors. An economical solution to this problem has been adapted from the Roland Modela, and is presented on the next page as Figure 11.

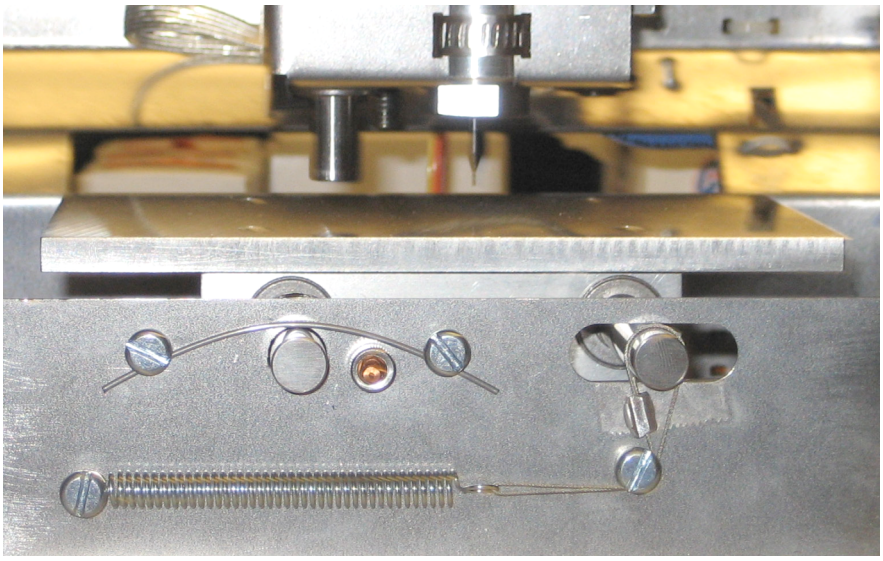


Figure 11: Method of Mitigating Over-Constraint

The position of one rail is exactly defined by its mounting conditions: bent pieces of piano wire act as springs to preload the ends of the shaft against V-grooves cut into the sheet metal chassis. It is this rail which constrains the vector of motion of the carriage. The second rail rests against horizontal slots in the chassis. Spring-tensioned cables apply a downward preload force to the ends of the rail while providing little resistance to lateral movement. In this way the second rail acts only to prevent rotation of the carriage about its axis of movement without attempting to also specify its linear path.

It should be noted that although this design allows for inconsistent heights of the chassis sides, over constraint may still occur if the rails are not parallel in the supported direction. The author does not anticipate this to be a problem in a mass-produced model, as CNC bending equipment is quite good at making parallel bends.

Virtual Guides

Each virtual guide is a programming object with a 3 x 1 unit vector attribute which describes the mechanical constraint imposed by the real guide. For example, a perfectly aligned x-axis constraint vector would be [1 0 0]. If 0.05" of z-axis coupling is present across a 1" span of travel, the constraint vector would be [0.999 0 0.05]. Guide objects contain a method whose input is the distance traveled along the guide and whose output is the resulting displacement in 3D space. The internal calculation is simple: $\Delta = sC$, where Δ is a displacement vector of the form $[\Delta_x \ \Delta_y \ \Delta_z]$, s is the travel along the guide,

and C is the constraint vector. Motion from the virtual transmission thus results in three-dimensional movement of the payload.

Error Mapping Using Virtual Guides and Feedback Loops

One useful application for virtual guides is error mapping. If guide misalignment can be accurately modeled, a feedback loop can be implemented in the virtual machine which compensates for systematic errors in the real machine. This is included in the schematic of Figure 7 as a signal path leading from the "part and tool location" box back to the motion controller.

A virtual feedback loop has been implemented in the low cost circuit board mill's controller as a means of correcting for manufacturing inaccuracies. This same technique could also be used to map other systematic errors like thermal growth or structural deflection. One could imagine a series of I0 enabled temperature and strain sensor modules scattered throughout a real machine which constantly update the virtual machine.

X and Y Axis Transmissions

Both the X and Y axes utilize capstan drives to convert the rotation of a stepper motor into linear movement along the guides. Capstan drives are a form of friction drive in which a cable is wrapped around a rotating shaft, or capstan. Rotation of the capstan results in linear advancement of the cable. Figures 12 shows a view inside the box-extrusion carriage. Box extrusion is perfect for this application because it houses the entire transmission system internally and allows the capstan to be supported by two bushings.

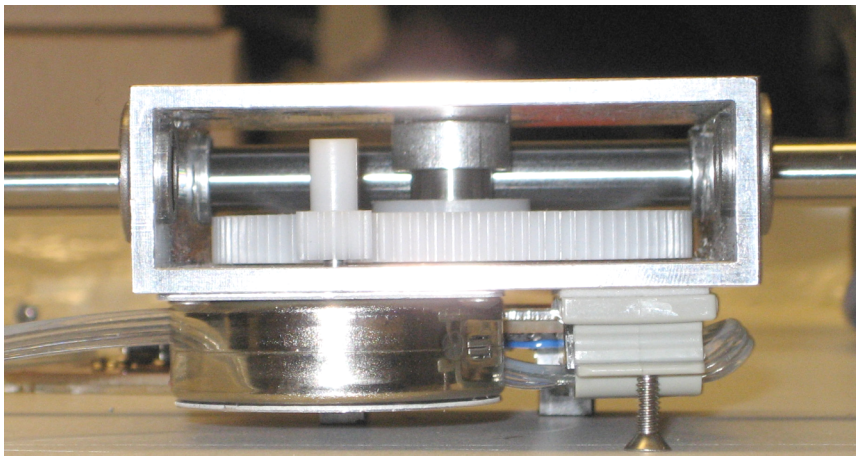


Figure 12: X and Y Axis Power Transmission

A small molded Delrin spur gear is glued to the output shaft of a cheap ($1.35) stepper motor, and mates to a larger spur gear which has been press-fit onto the capstan. The mounting configuration of the capstan and larger spur gear is shown schematically in Figure 13.

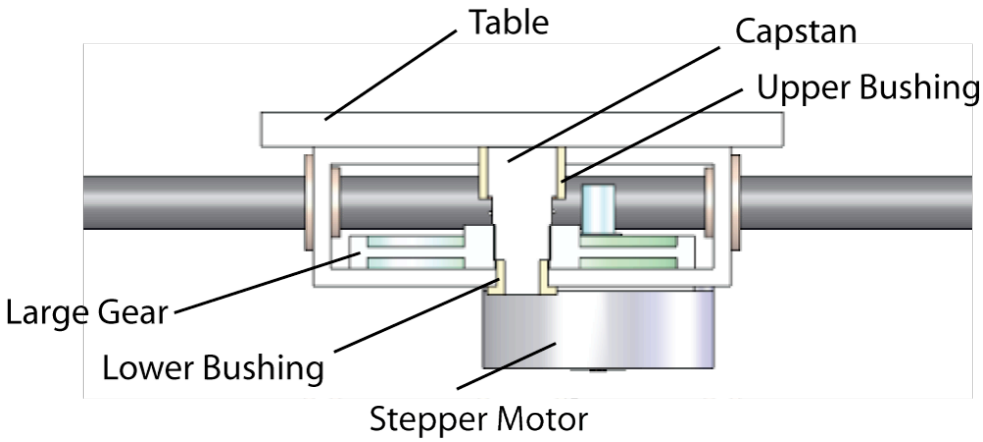


Figure 13: Capstan Configuration

The capstan is supported on both ends by bronze bushings which are press-fit into the box extrusion. The lower bushing is flanged, and fits comfortably within the bore diameter of the large gear. The upper bushing is unflanged. The small length of exposed capstan is turned to the precise diameter required to achieve 0.001" of motion per step of the stepper motor. A top section view of the capstan drive is presented by Figure 14, followed by a detailed description of the drive's underlying workings.

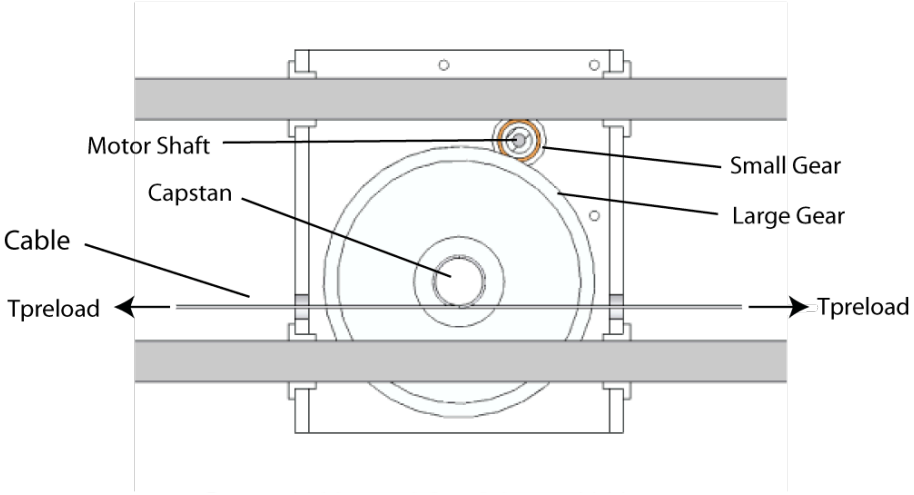


Figure 14: A Top Section View

The stepper motors which drive the X and Y axes were selected because of their compact size, low cost, and unipolar phase wiring (which simplifies the

design of the motor driver circuitry). Unfortunately, their step size is sub-optimally large at 3.75 degrees per step. A gear train is therefore necessary to increase the stepping resolution of the drive system. Several problems result. First, the largest single-stage gear ratio which can be fit within a 1"x3" box extrusion (an ideal size based on available options) is roughly 9:1. This means that 720 steps of the motor will result in one full revolution of the capstan. In order to achieve a linear resolution of 0.001", the capstan diameter cannot exceed 0.23". There is a general design principle for capstan drives which says that the cable diameter must be 25 times less than capstan diameter to prevent excessive fatigue of the cable [3]. The cable diameter required for this system must therefore be less than 0.009"! Steel cable of this size has a breaking strength of only 10 lbs. Prof. A. H. Slocum, in his book *Precision Machine Design,* recommends a factor of safety of 10 between the maximum anticipated load and a cable's rating breaking strength [3]. To understand analytically why the resulting maximum working load of 1 lbs is not nearly enough, it is important to examine the physics of a cable wrapped around a capstan.

The operation of a capstan drive in fundamentally based on friction between the capstan and the cable. Equation 2 is the coulomb model of static friction, and gives some intuition into the important parameters.

$$F = \mu N \tag{2}$$

F is the frictional force, μ is the coefficient of friction between the capstan and the cable, and N is the normal force acting between the capstan and cable. The greater the friction force, the more traction the drive system can apply to the cable to generate propulsion. μ is a constant which is approximately 0.15 for plastic-coated wire [3]. The final variable is the normal force. By wrapping the cable around a circular shaft and placing it under tension, a component of the tension acts radially (i.e. normal to the interface.) Therefore the cable must be pre-tensioned for capstan to exert traction on the cable. This insight is where the direct applicability of equation 2 ends.

A generalized capstan equation exists which fully describes their behavior [3]:

$$\theta_{\max} = \frac{Log_e \frac{T_{preload} + T_{drive}}{T_{preload}}}{\mu} \tag{3}$$

Two tension forces and a coefficient of friction determine the wrap angle necessary to prevent slipping of the capstan. $T_{preload}$ is the static pretension

externally applied to the cable, and T_{drive} is the force the capstan applies to the cable which causes the carriage to move on the guides. Preload tension is applied to the cable by two vented socket head cap screws which are screwed into threaded holes in the chassis – the cable passes through the vents and is retained by copper stop sleeves.

Figure 15 below shows a force balance acting on a rotating capstan.

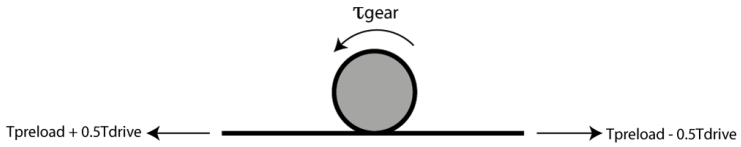


Figure 15: Capstan Force Balance

The torque applied to the capstan by the gear, τ_{gear}, results in a drive force equal to the product of the applied torque and the capstan radius. This force manifests itself as a tension difference between the sections of cable on either end of the capstan. Thus, the section of cable which is pulling the carriage has a net tension of $T_{preload} + 0.5T_{drive}$, and the slack section has a net tension of $T_{preload} - 0.5T_{drive}$. The difference in tension between these sections (because they act in opposite directions) equates to a net force of T_{drive}.

For a given coefficient of friction, there is a wrap angle θ_{crit} at which preload tension will be entirely lost in the slack section of cable at exactly the same load at which slip occurs. Preload is lost in the slack section when $T_{preload} = 0.5T_{drive}$. Substituting this into Equation 3 yields:

$$\theta_{crit} = \frac{Log_e 3}{\mu} \tag{4}$$

For a coefficient of friction of 0.15, θ_{crit} evaluates to a wrap angle of 7.32 radians or 1.2 turns, which is independent of both the pretension and the drive force. In the present design the cable is wrapped around the capstan just once to ensure that slip always occurs before preload loss.

Given a wrap angle of 6.28 radians (1 turn = 2π radians), it is now possible to calculate the pretension necessary to prevent slip at a given drive load by rearranging Equation 3 into Equation 5 below.

$$T_{preload} = \frac{T_{drive}}{(e^{\mu\theta} - 1)} \tag{5}$$

It should now be clear why a cable with a working load of 1 lbs is hardly sufficient for milling even a soft material like copper-clad board. The

maximum cable tension was shown earlier to be $T_{preload} + 0.5T_{drive}$. Setting this expression equal to 1 lbs and substituting this and the known values θ (6.28) and μ (0.15) into Equation 5 gives a maximum driving force of approximately half a pound. Much of the available working tension is used by the pretension force to establish traction. One half pound of cutting force was deemed insufficient during the design process based on the author's personal experience with hand tools.

The solution arrived at is to half-step the motor. Half-stepping is when two phases of the stepper motor are energized simultaneously to achieve an average position between two detents [4]. By decreasing the step size of the motor by half, the capstan and cable diameter can be doubled. A 0.018" steel cable has a breaking strength of 40 lbs, which translates to a maximum working load of 4 lbs. Equation 5 and the equality 4 lbs = $T_{preload} + 0.5T_{drive}$ indicate that the maximum drive force now available is roughly 2 lbs, which is a 4-fold increase.

One problem still remains. At their shafts, the motors can generate 300 g-cm of torque at a running voltage of 12V (they are rated at 24V and 600g-cm, but a 12V computer power supply is much more economically obtainable.) This translates through the 9:1 gear train to 2.7 kg-cm of force at the capstan. If the capstan has a diameter of roughly 0.45 inches, the maximum force which can be exerted on the cable should the carriage become jammed is much larger than the maximum working load at approximately 10 lbs. The solution to this problem – which makes the drive system possible – is to utilize the capstan as a torque-limiter. By adjusting the pretension in the cable adequately using Equation 5, slip can be induced at a maximum working load of 2 lbs. This would of course cause the machine to lose its position, but this is preferable to a cable snapping. For a slipping load of 2 lbs, the necessary pretension is 1.27 lbs. One method for accurately setting this pretension is to use an instrument tuner and to pluck the cable as it's being tensioned. The natural frequency of a cable can be found using Equation 6, and is dependent only on the tension in the cable and its length [5].

$$\omega_n = \sqrt{\frac{4T_{preload}}{0.37m_{cable}}} \tag{6}$$

It is up to the reader to determine the mass of the cable used for each axis and the length between the capstan and cable anchor point.

Z Axis Overview

The Z axis is responsible for controlling the height between the work piece and the tool spindle. Figure 16 below is an annotated picture of the Z axis assembly.

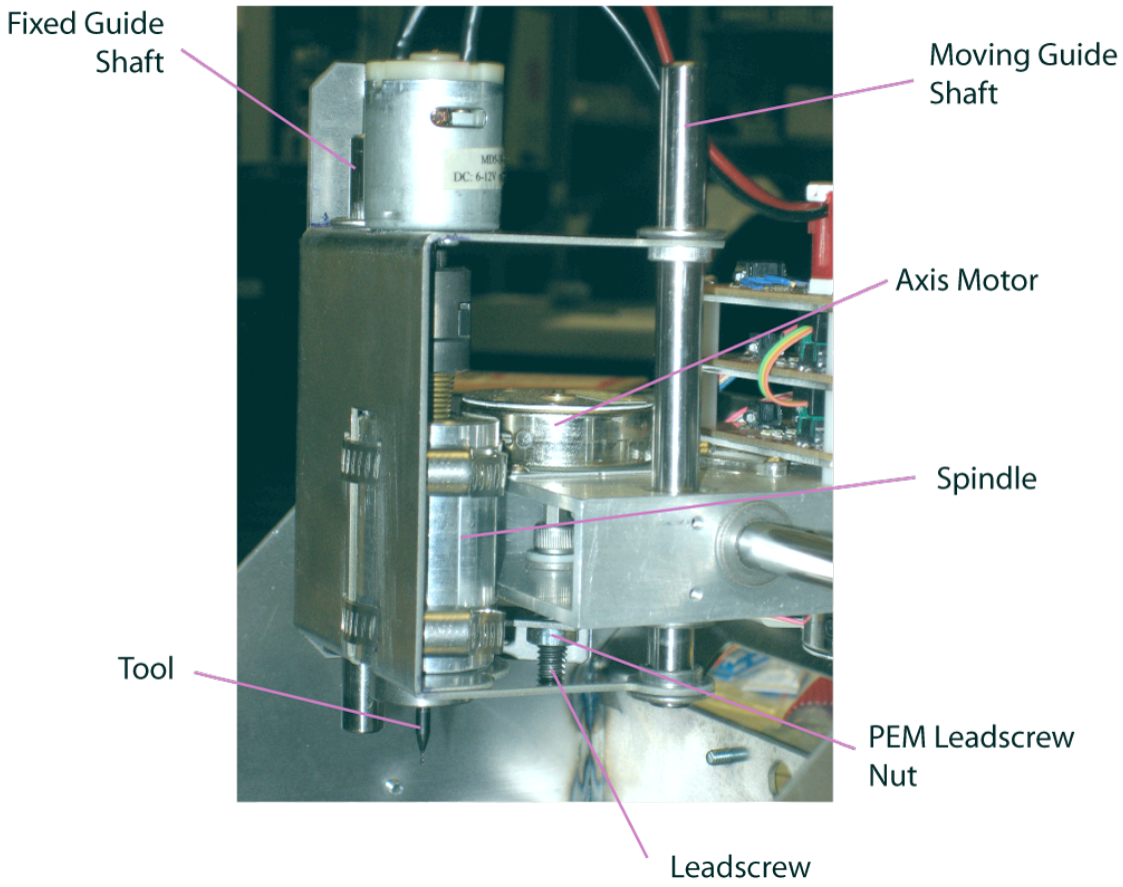


Figure 16: The Z Axis Assembly

As was mentioned earlier, the Z axis carriage is made from 0.06" mild steel which was waterjet cut to shape and then bent. Unlike the X and Y axes, a leadscrew is used to actuate movement in the Z direction. This choice was made for two reasons. First, a leadscrew lends itself best to the task of attaching the Z axis directly to the same box extrusion which houses the X axis components. Second, the higher resolution offered by a leadscrew over a capstan (roughly a factor of 4) opens up the possibility of using error mapping to compensate for systematic errors in the other two axes.

Z Axis Guide

Unlike the X and Y axes, the Z axis guide represents an exact-constraint design. This means that the position of the Z axis carriage is exactly defined by its bearing surfaces and the leadscrew. The Z axis guide configuration is shown in Figure 17.

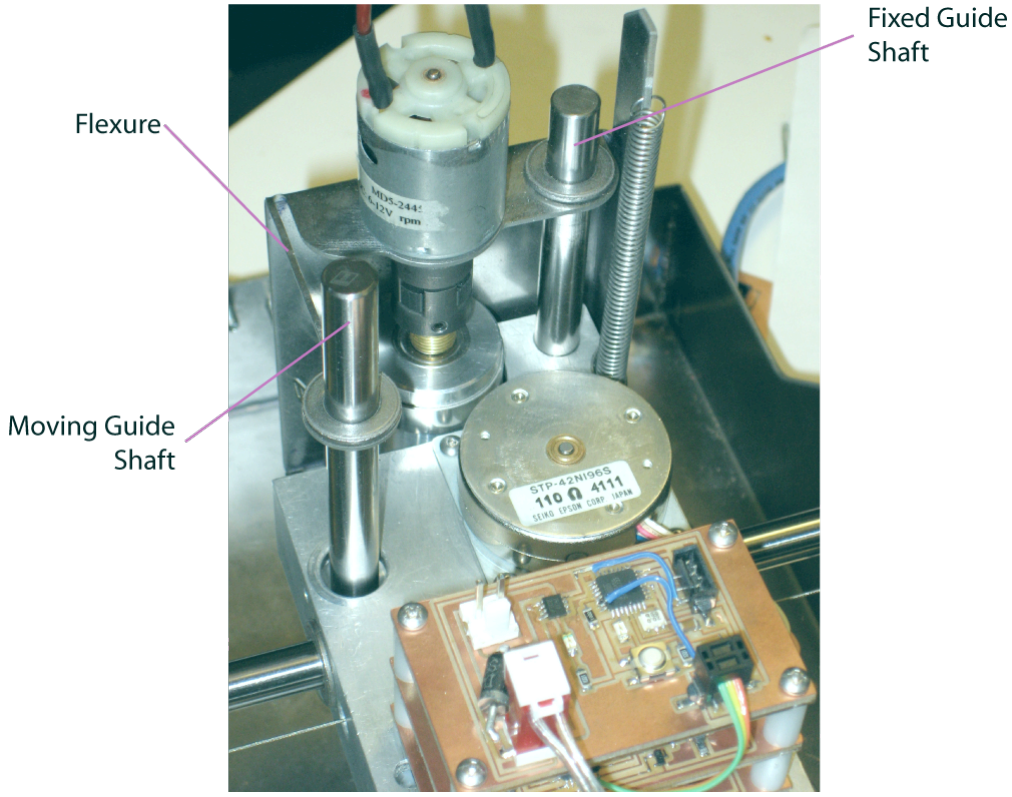


Figure 17: Z Axis Guide Configuration

One of the guide shafts is fixed to the X axis box extrusion by passing it through a reamed hole in the extrusion and then retaining it with two flat-end setscrews. It is this rod which defines the vector of motion of the carriage. Twin bronze bushings are pressed into flanges in the sheet-metal carriage and provide a bearing surface with the fixed rod. The moving guide shaft is fixed at both its ends to flexures on the Z carriage and slides in a single bushing which is press-fit to the lower surface of the X axis carriage. The flexures provide a high stiffness in their axial direction – the only direction needing constraint which the fixed rod does not address – while still flexing in the tangential direction. An exact-constraint design is crucial for this axis because of the inaccuracy of the bent sheet metal carriage.

It might be puzzling at first why bushings are used in the flexures if the rod is fixed to them. Bushings are a cheap way of increasing the thickness of the flexures where the rod attaches, thereby prevent the flexure from pivoting at the joint with the rod. This increases the stiffness of the flexure along the axis of the rod by a factor of 2 which helps protect the flexures from plastically deforming if the rod is hit in its only unconstrained direction. The reason this works is because the bending moment at the root of the flexure, where yielding first occurs, is now also supported by the joint between the flexure and guide shaft. Intentional jamming promoted by a dab of high-viscosity superglue is enough to lock the moving shaft to the flexure bushings. The glue should be removable with a cyanoacrylate solvent, although this has not yet been tested.

Z Axis Transmission

Controlled motion of the Z axis is achieved using a leadscrew. Figure 18 gives an overview of the present implementation.

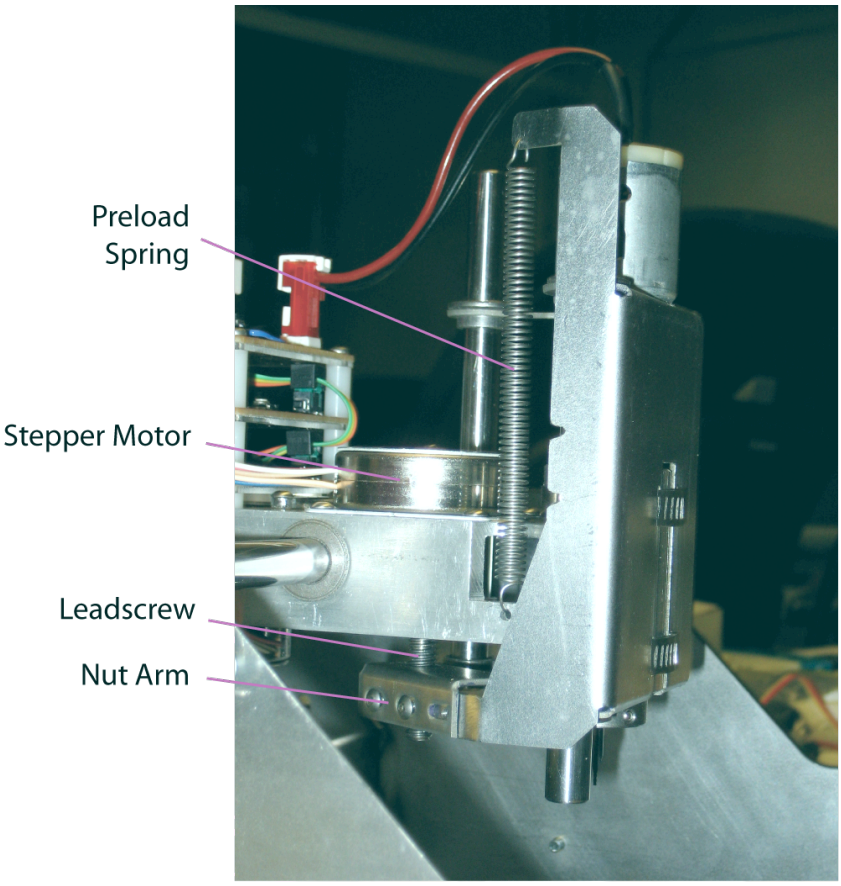


Figure 18: Z Axis Transmission Configuration

A stepper motor mounted to the X axis carriage directly drives a socket head cap screw acting as a low cost leadscrew. This leadscrew applies thrust to the Z axis carriage via a PEM nut attached to the nut arm. A PEM nut is a form of threaded insert which can be press-fit into sheet-metal. A spring stretched between the Z axis carriage and X axis box extrusion assists gravity in preloading the carriage against the leadscrew. This eliminates backlash between the leadscrew and nut.

One of the challenges of using bent sheet-metal for the carriage is aligning the nut with the leadscrew. Instead of directly mounting the PEM nut to the nut arm, it is mounted to a small steel L bracket (shown in Figure 19).

Figure 19: The PEM Nut Pressed into an L Bracket

Slots in the L bracket, through which mounting screws pass, permits adjustment along the axis of the nut arm. The ductility of the arm itself allows repositioning in the perpendicular direction. These two modes of adjustment suffice to exactly position the nut underneath the leadscrew.

Figure 19 on the next page shows how the stepper motor is coupled to the leadscrew. An unthreaded female 3/16" hex standoff is mounted with superglue to the shaft of the stepper motor. The leadscrew, a stainless steel ¼-20 socket head cap screw with a key size of 3/16", rotates inside a bronze bushing. A low-friction washer acts as a thrust bearing between the stainless steel and the bronze bushing. Rulon J (reinforced Teflon) is the preferred material for this bearing because of its unique property of having a lower static than dynamic coefficient of friction. This prevents slip-stick which otherwise tends to stall out the motor. Because there is no reduction between the motor and the leadscrew, rotational friction has a more significant affect

than on the X and Y axes. One countermeasure is to "power-step" the motor, which provides 1.4 times the torque but consumes twice as much power [4].

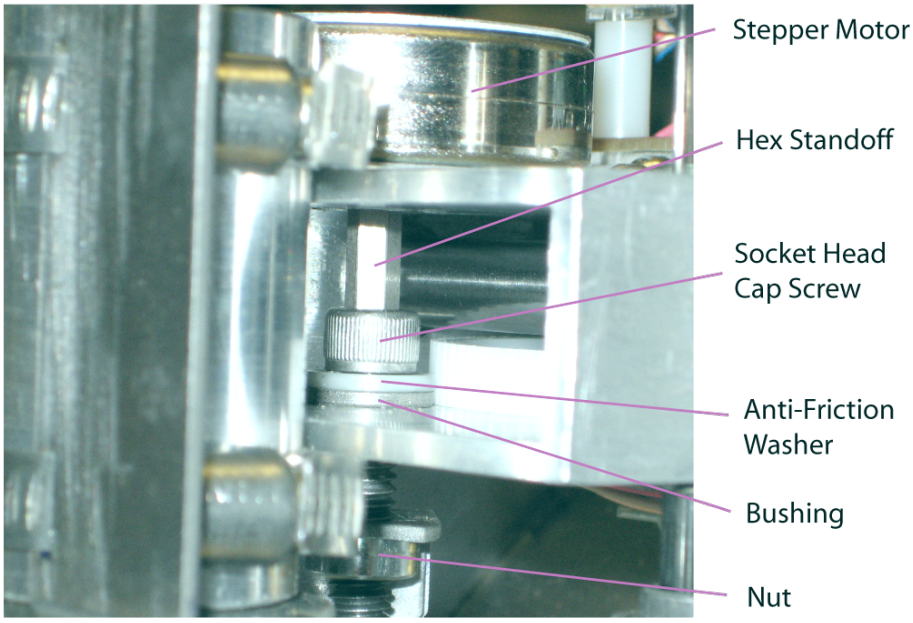


Figure 20: Leadscrew Coupler

One danger of the present coupler design is the danger of overloading the motor's internal bearings. There is a slight gap between the hex standoff and the bottom of the screw's socket. If for some reason the preload force on the carriage is overcome, the screw will begin acting in reverse and may apply an axial thrust to the motor shaft. One solution is to place a loosely fitting spacer between the head of the screw and the top of the X axis box extrusion. This would transfer the thrust load to the box extrusion instead of the motor shaft.

Virtual Transmissions

Despite the complexity of the mechanical transmissions, the virtual transmissions are incredibly simple. Each transmission object contains a single attribute, the overall reduction ratio between the rotation angle of the motor and the resulting linear travel, and a method which multiplies this attribute by an input angle to return a linear distance of travel along the guide.

Virtual Motors

The physical motors were discussed in detail previously. Virtual motors are programming objects which, like the virtual transmissions, contain a single attribute and a single method. The attribute is the rotation angle per step, and the method multiplies a step input to return a resulting angular displacement of the motor shaft.

Virtual Axes – a Recap

Virtual motors, transmissions, and guides can be chained together to tell the controller the movement of an axis in 3D space in response to motor steps. It is in this way that the virtual machine is assembled to behave similarly to the real machine.

Table

The work table is a 4" x 6" x 0.25" plate of MIC 6 precision machined cast aluminum. While relatively expensive, this Alcoa product saves much of the time and frustration inherent in attempting to obtain a flat surface on a thin plate of extruded or rolled aluminum. These cheaper alternatives have surface stresses which cause them to distort when fly cut. The table costs around $6 in materials – a price which is easily justified by the savings in manufacturing time.

Spindle

The spindle's purpose is to rotate the cutting tool at high speeds while simultaneously providing rigid support in the transverse and axial directions. A cross-sectional view of the spindle is shown in Figure 21 on the following page. An effort was made to design all of the manufactured spindle components, namely the shaft and housing, as parts which could be turned on a lathe without requiring further machining. Also, all of the precision diameters (with the exception of the tool bore) are machinable in a single setup. These decisions were made as a way of reducing manufacturing cost and complexity. It is for this reason that the reader will notice unusual bearing selections, such as smaller bearings being used for the larger diameter sections of shaft.

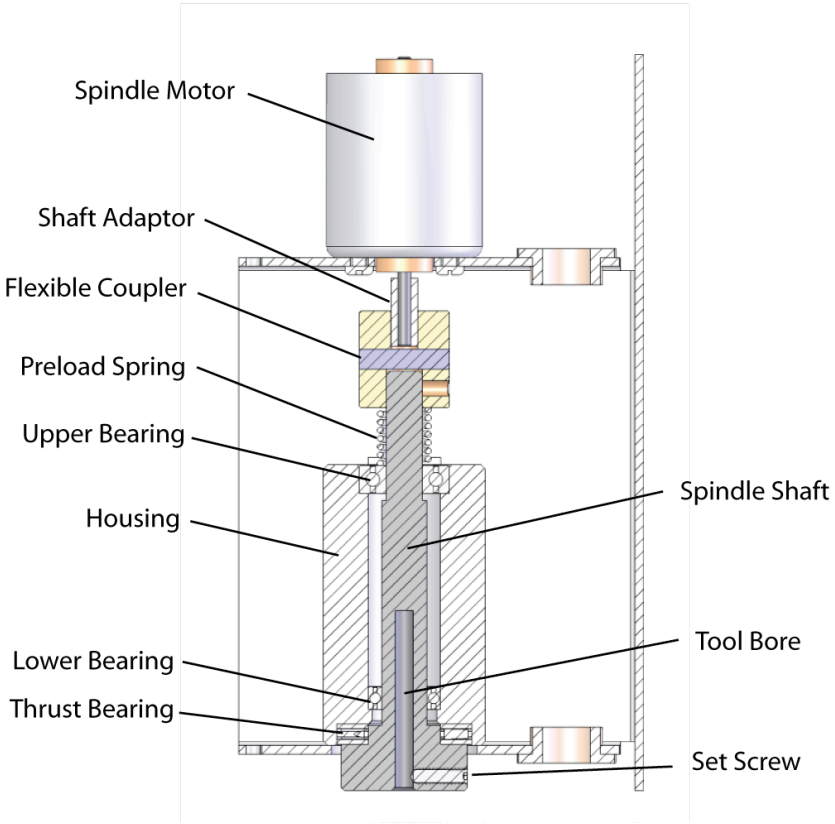


Figure 21: Spindle Cross-Section

The steel spindle shaft is supported by three bearings inside an aluminum housing. Two of these bearings are ABEC-5 radial bearings which are lightly press-fit into the housing, and the third is a needle-roller thrust bearing. The purpose of the thrust bearing is to give the spindle stiffness in the axial direction. A flexible shaft coupler is used to attach the spindle shaft to the motor while compensating for misalignments caused by manufacturing tolerances. The flexible coupler is a spider type: a Buna-N rubber spider rests between fingers protruding from two steel disks attached to the motor shaft and spindle shaft. It should be noted that an adaptor sleeve has been press-fit onto the motor shaft because a coupler disk with the proper bore size could not be obtained. A spring sandwiched between the upper bearing and the spindle shaft's coupler disk (which is held in place by a set screw) applies a preload force of several pounds to the thrust bearing. The purpose of this preload is not to increase the stiffness of the bearing – it merely maintains contact between the shaft and the thrust bearing. A 1/8" hole is reamed down the center of the spindle shaft to receive a carbide end mill. A set screw retains the tool in place. The set screw is located in the largest diameter of the spindle shaft – both the set screw length and shaft diameter were chosen so that the set screw fully fill the hole when tightened. This balances the spindle and reduces vibrations.

The spindle motor is a low-cost DC blow dryer motor (~$2), and is able to propel the spindle at speeds exceeding 15,000 RPM. More than sufficient torque is available to overcome the low cutting forces generated by a small-diameter tool rotating at high speeds and cutting through only several thousandths of an inch thick copper. Close-up photographs of the spindle are shown in Figure 22.

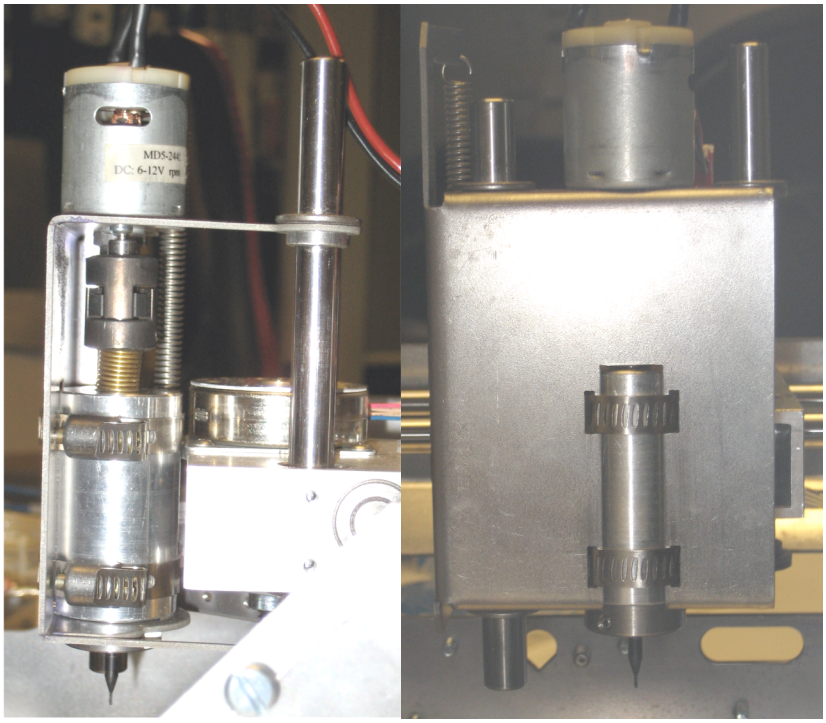


Figure 22: The Spindle

As can be seen in the photographs, hose clamps are used to attach the spindle to the Z axis carriage. This has proven to be economical and incredibly rigid. A slot in the carriage aligns the spindle.

Distributed Controller Network

The introductory section of this project described the distributed-controller paradigm for which this low cost circuit board mill acts as a test bed. Figure 8 of that section showed a photograph of one of the stepper motor control boards. Figures 21, 22, and 23 on the next page indicate the locations of all of the distributed controllers and support circuitry on the machine.

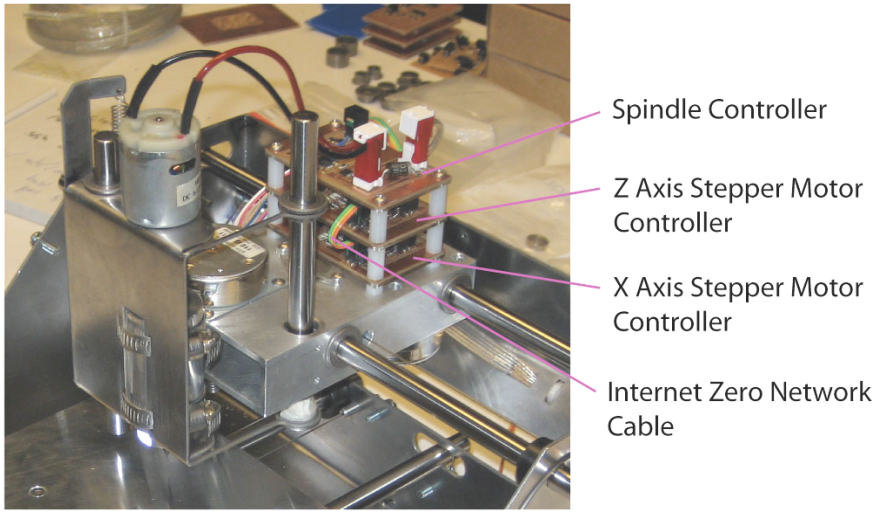


Figure 23: X Axis, Z Axis and Spindle Distributed Controllers

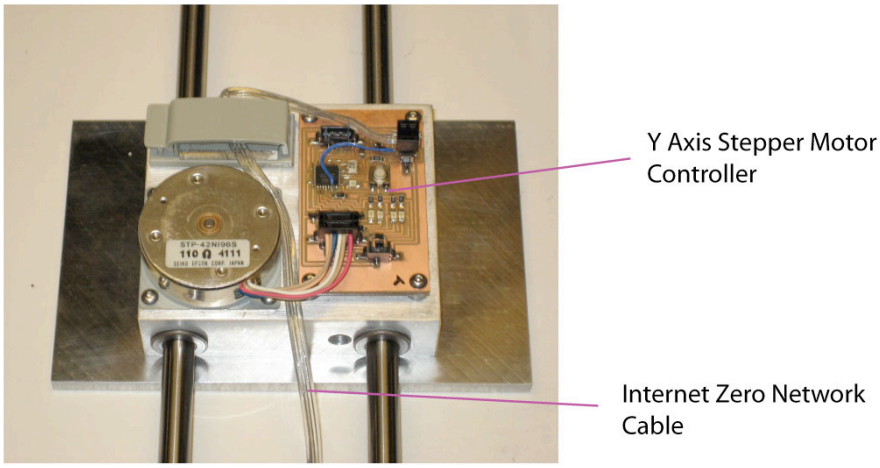


Figure 24: Y Axis Distributed Controller

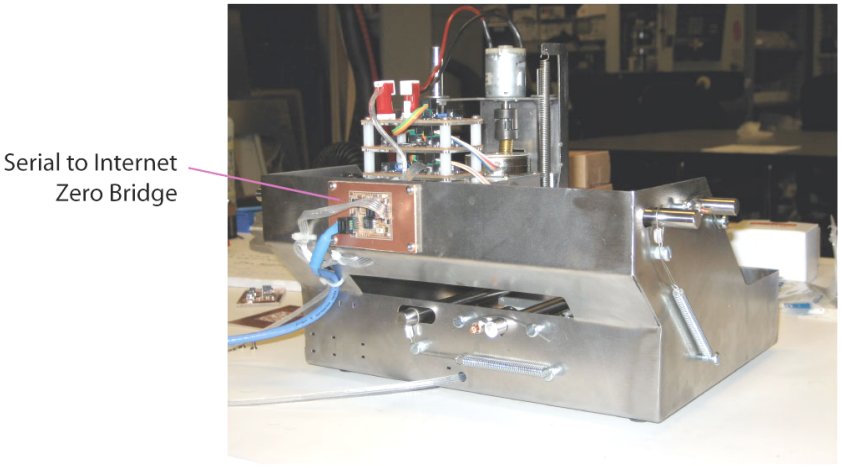


Figure 25: Serial to Internet Zero Bridge

A total of four networked nodes are present on the machine: one for each stepper motor axis and a separate controller for the spindle. The network is shown diagrammatically in Figure 26 below.

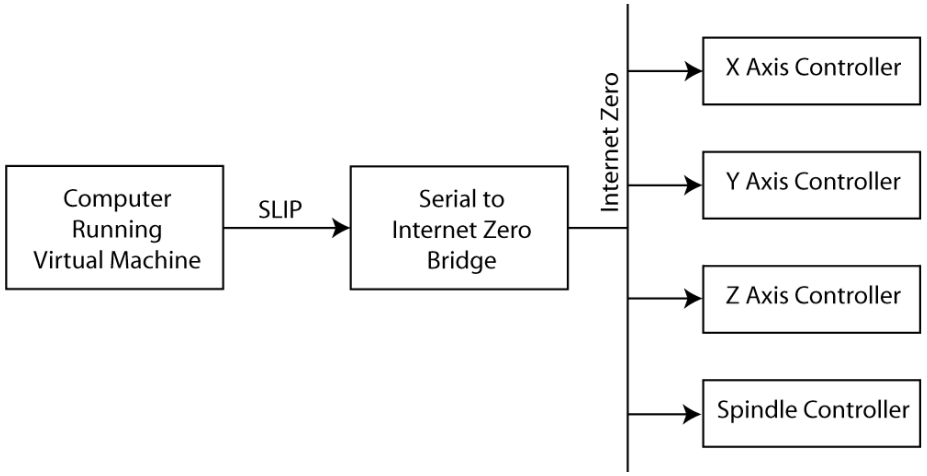


Figure 26: Distributed Control Network, As Implemented in the Current Project

Commands originate at the virtual machine, and leave the computer as IP packets using the *serial line internet protocol (SLIP).* A dedicated circuit called the bridge (Figure 25) converts SLIP packets into I0 packets which then enter the I0 network. The distributed controllers listen to these packets, and will act upon any packet bearing their IP address in its destination field. All of the Internet Zero communications code and the design of the bridge module was done by David Kopp of Schnieder Electric in collaboration with the MIT Center for Bits and Atoms.

Just like their more complex cousins, I0 nodes have both IP addresses and internal ports. Different commands are distinguished by the controllers based on the port to which they are addressed.

The spindle controller has a single port – the "set spindle speed" port. A single byte of information sent to this port sets the speed of the spindle anywhere in the range from 0 (fully off) to 255 (fully on.) The spindle controller has a PWM circuit which is able to modulate the speed of the spindle motor according to this setting.

The stepper motors are slightly more complicated. Several commands must be issued in order to establish synchronized motion between several motors – a necessity for generating motion at angles. The first command, sent to the "setup" port, tells each controller the number of steps to take and how long to wait between steps. Any number of steps between 1 and 65536 can be requested. The time between steps is based on a three-stage internal counter and can thus vary by a range of 256^3. This flexibility is necessary to support

the wide dynamic range of feedrates required when generating sloped motion. A full setup command consists of 5 bytes in all – 2 bytes specify the number of steps and 3 bytes specify the time between steps. The second command, sent to the "sync" port, is a multicast packet which causes all axes to begin their previously configured motions. Multicast packets have a unique specifier which addresses them to all nodes on the network. A uniquely addressed packet is called a unicast packet. A stop command is also available which pauses the motion of the axes. Figures 27, 28, and 29 shows close-up photographs of the bridge and controller boards. All of the internet zero nodes are controlled by Atmel AVR series ATMega88 microcontrollers running at 20 MHz.

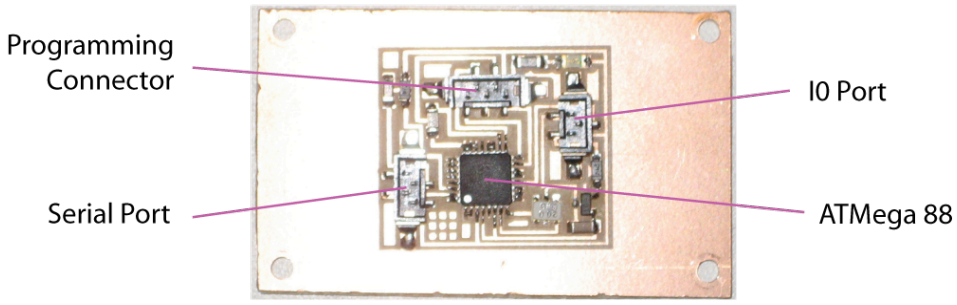


Figure 27: Bridge Module

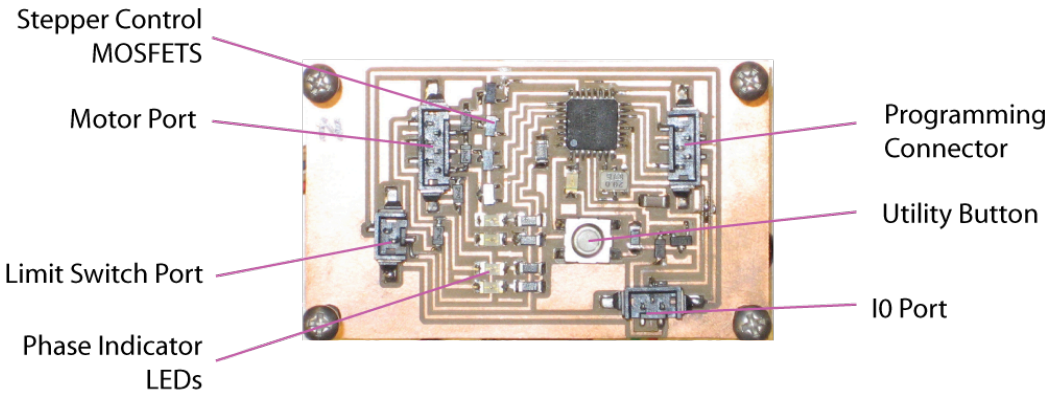


Figure 28: Stepper Motor Control Node

The stepper motor controller board operates by selectively activating phases of the stepper motor by sinking current through an array of flyback-protected

mosfets. LEDs are provided on each phase for the purposes of debugging. Any step pattern can be programmed into the onboard microcontroller, including power stepping and half-stepping sequences. The advantages of these alternative patterns have been discussed in both the X/Y and Z axis transmission sections.

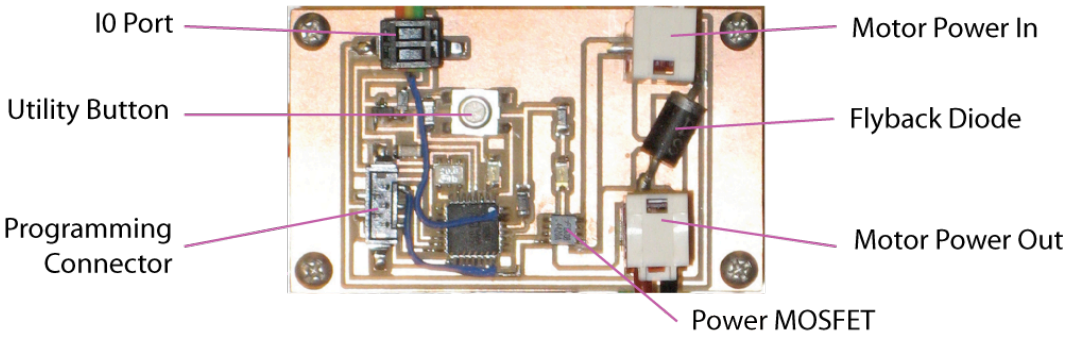


Figure 29: Spindle Control Node

The spindle controller has a single large MOSFET used to control the current running through the spindle motor. A separate high-current power plug is provided because the higher gauge I0 cables are insufficient for the 6A peak peak current draw of the motor.

Both the stepper motor and spindle control boards have "utility" buttons which can be used to initially communicate the node's IP address to the virtual machine. Two setup options are possible: the node generates a randomized IP address on startup and the utility button is used to send this address to the virtual machine, or the IP address is hard-coded into both the node and the virtual machine. The former is more appealing in a commercial setting, but the hard-coded option was used for the purposes of prototyping the system.

These boards were made using the same process that the low cost circuit board mill replicates. It could be interesting to use this project create its own circuit boards.

Virtual Stepper Motor Controllers

Internal to the real motor controllers are several pre-scaler and counter settings which center the dynamic range of the possible times between steps. The virtual stepper motor controller contains a method which accepts a requested time-per-step and returns a 3-byte threshold value which, when compared to the internal 3-stage counter running in the real motor controller,

generates a motor step. This 3-byte threshold can then be sent to the motor controller as the 'time to wait between steps' value on the setup port.

Virtual Motion Controller

The crux of the virtual machine is the motion controller. While this is being described as "virtual", there is no physical analog. The virtual motion controller is responsible for interpreting the list of movement commands output from the CAM software and generating motion commands which are sent out over the I0 network.

The first step of this process is to determine how far each axis has to move based on the difference between the machine's current position and the coordinates of the desired location. This is done in conjuncture with a error-mapping feedback loop wrapped around the virtual guides. The resulting linear axis movements are then processed through the respective virtual transmissions and motors to obtain a necessary number of steps. Each movement command also includes an overall feedrate which is decomposed onto the directions of the machines axes. The overall 3D distance of the move is calculated using the Pythagorean Theorum recounted by Equation 7.

$$d = \sqrt{\Delta_x{}^2 + \Delta_y{}^2 + \Delta_z{}^2} \tag{7}$$

Dividing this distance by the feedrate yields an overall time for the move. Because all of the axis must arrive at their final destination simultaneously, the move time is universal for all axes. Dividing the move time by the number of steps gives a time-per-step for each axis, which is then fed into the virtual stepper controllers to get a 3-byte value suitable for transmission over the I0 network to the real stepper motor controllers.

Once all of the stepper motors have been properly set up, a multicast packet is sent to the sync port to initiate movement of the real machine.

If the movement list includes spindle on and off commands, these would be sent to the spindle controller at the appropriate times. Otherwise, the spindle is turned full-on when the sequence of moves is initiated, and turns off when finished.

Concluding Remarks

A control system has been described in which a virtual machine controls a real machine over a distributed network. This new architecture aims to reduce the time necessary to implement control systems in both DIY and commercial rapid prototyping machines. Other benefits have been discussed, such as the ease with which systematic error compensation can be implemented using virtual control loops, unlimited expandability, and the reduced wiring complexity of the distributed network.

The mechanical design of a low cost circuit board mill has also been discussed in detail. While this design effort was motivated by the need for a test bed, several innovations have resulted which may enable the creation of a new breed of low-cost rapid prototyping machines. Capstans have been used as traction drive systems with integral torque-limiters based on controlled pretension. Methods of mitigating the manufacturing errors inherent in sheet-metal construction were explored, and the collection of solutions developed (and borrowed) enabled the use of low-cost sheet metal in the design. Box extrusion was also used as a way of reducing the number of components necessary to implement bearing supports, and reducing the overall material cost and weight. Finally, an exact constraint Z axis design was generated.

A PARAMETRICALLY DESIGNED XY MOTION STAGE

Introduction

One of the major hurdles in the development of rapid prototyping machines is the synthesis of a mechanical design. I believe that modularity and parametric design provide a solution. As was shown in figure 2, RP machines can be modeled as a collection of axes which enable relative motion between an effector (such as a tool spindle) and a workpiece. Each axis is a module which can share many common design elements with not only the other axes of the machine, but the axes of many potential rapid prototyping machines. If the dissimilarities can be defined mathematically according to engineering principles, it may be possible to create a single "universal" design which can be custom-tailored for unique applications merely by changing several parameters in a CAD tool.

Just as software engineers create libraries of code which they share, the goal of this project is to create a mechanical design as a 'function' which returns a valid solution in response to engineering parameter inputs. Designs created in this way are inherently reusable because they fully capture the engineering logic driving their geometry.

This section will present an XY table, one of the most common axis configurations found in RP machines, whose design is driven by three engineering parameters: the stiffness (i.e. resistance to deflection) of the stacked guides and the travel of each. This was accomplished using the SolidWorks CAD tool linked to Microsoft Excel design tables. A bill of McMaster-Carr source-able materials is automatically generated as well.

Design Overview

In order to make the XY table useful to the DIY community, several constraints were imposed upon the design:

1. All parts and material are available online from McMaster-Carr.
2. Aluminum box extrusion and architectural channel are used whenever possible to reduce cost and fabrication effort.
3. Only manual machine tools are to be used in fabrication.

Figure 30 is a photograph of the final design. It should be noted that the X and Y axes are conceptually identical except for the dimensions and payload elements. The Y axis payload is the X axis, while the X axis supports a table constructed from aluminum architectural channel. This could of course be modified to fit a specific application. Also, the two axes are coupled by a single box extrusion which in the spirit of modularity can be though of as two extrusion segments welded together.

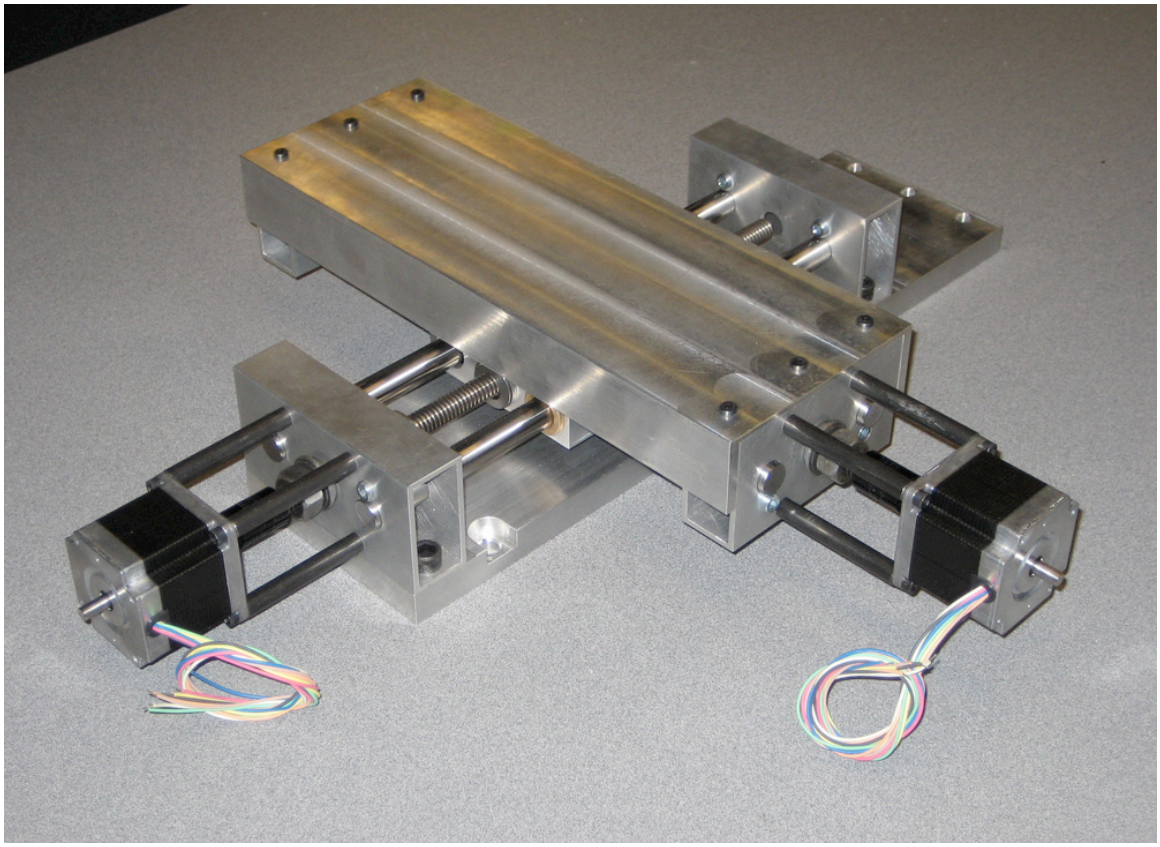


Figure 30: The Finished XY Motion Stage

Axis Overview

Figure 31 depicts schematically the design of a single axis and points out some of its key features.

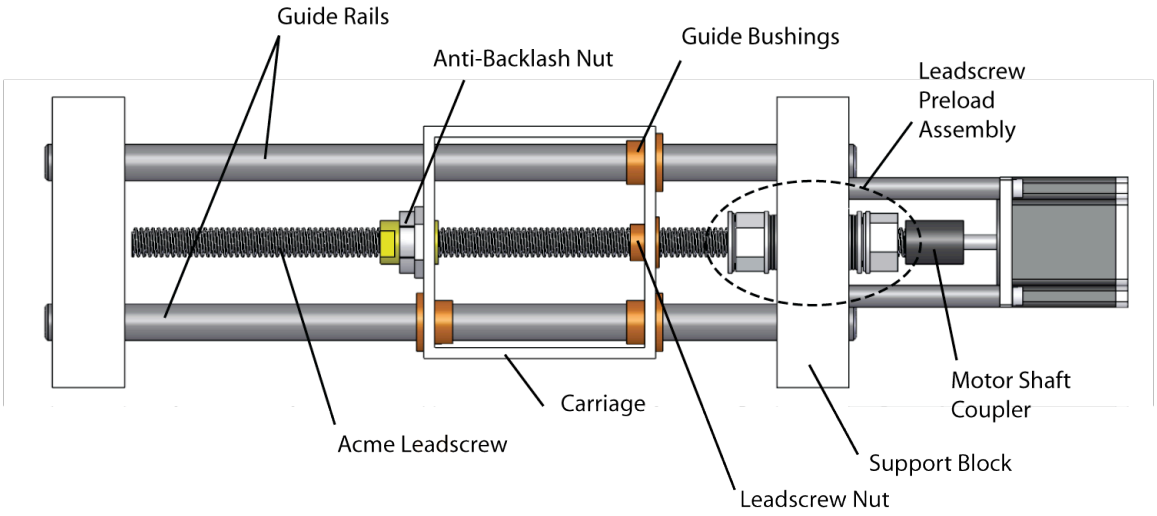
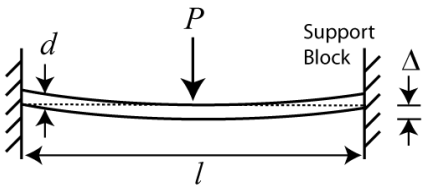


Figure 31: Parametrically Designed RP Machine Axis

Guides

The structural component of the axis is formed by two precision ground rods supported on both ends by aluminum box extrusion. The length and diameter of these rails directly influence the stiffness of the axis according to the standard bending equation for a beam with two fixed ends. This is given by equation 8 [6]. It should be noted that the support blocks prevent rotation of the beam about its transverse axes – hence the use of this particular equation.



$$K = \frac{P}{\Delta} = 3\pi E \frac{d^4}{l^3} \qquad (8)$$

In equation 8 above: K is the stiffness of the axis, P is the applied load, Δ is the deflection of the axis, E is the Young's Modulus of the precision shafts (190 GPa for steel), d is the shaft diameter, and l is the length of shaft between the support blocks.
The use of box extrusion for the support blocks posed a problem of restraining the support rails. Figure 32 shows the solution arrived upon. National Pipe Thread plugs are placed directly above the entry holes of the guide rails on both sides of each support block. Because the plugs are tapered, tightening

them into their threaded holes generates a stress field which locks the guide rails into place. This prevents lateral motion of the rods and also preloads them against the support block – increasing the stiffness of the joint.

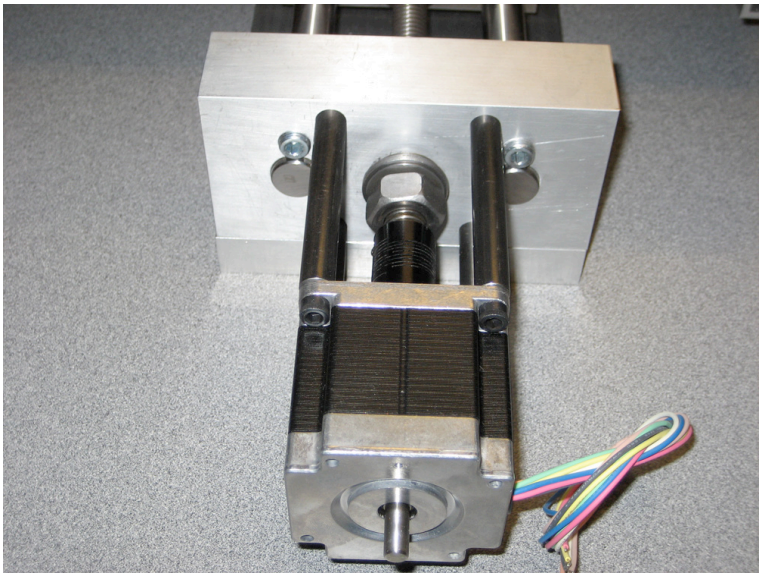


Figure 32: NPT Plugs Used To Retain Linear Guide Shafts

Transmission

Rotary motion of the stepper motor is translated into linear motion of the carriage by an ½-10 acme leadscrew. Backlash is removed from the system by using two leadscrew nuts. One nut is rigidly fixed to the carriage, while the other is externally threaded with a different pitch (16TPI) than the acme nut. The difference in pitches allows the backlash to be taken up by turning the threaded nut against the leadscrew. A locknut is used to secure the threaded nut after adjustments are completed. It should be said that this aspect of the system has not yet been successfully tested. The motor is engaged to the leadscrew using a helical beam shaft coupling. The preload assembly, shown in Figure 33 on the next page, constrains the leadscrew axially.

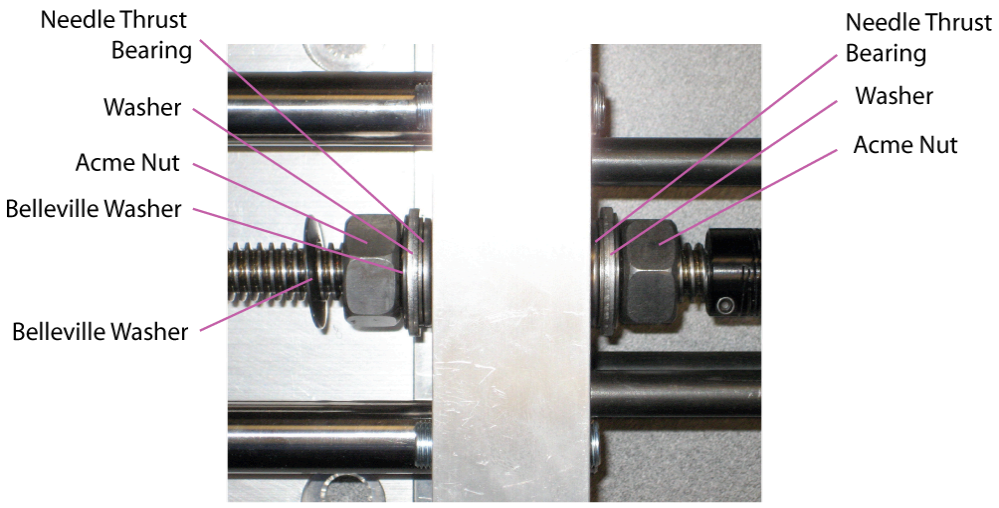


Figure 33: Leadscrew Preload Assembly

High thrust forces are generated at the carriage by the mechanical reduction of the leadscrew. Assuming a 10% leadscrew efficiency, a stepper motor capable of 136 oz.-in. of torque can generate over 50 lbs of thrust through a ½-10 leadscrew. Needle roller bearings were chosen to minimize rotational friction under the potentially high thrust loads, and are placed on the leadscrew on both sides of the motor support block. A Belleville washer is then placed on one side of the support block, and the entire assembly is sandwiched by two Acme nuts held in place by Loctite. The purpose of the Belleville washer is to permit the application of preload to the bearings in a controlled manner – the bearings are very stiff on their own and a small rotation of the Acme nuts could easily overload them without a compressive washer present. As long as the preload exceeds the maximum force on the assembly, the leadscrew will have an axial support stiffness equal to that of the support block. An additional Belleville washer is floating on the carriage side of the assembly to help reduce the impact caused by an accidental collision.

Towards a Reusable Design

At this point, the author would like to draw a distinction between two types of parametric design. The first might be called 'engineering driven parametric design', and is concerned with driving geometric dimensions to achieve engineering performance goals. The guide shafts are an example of this – in fact, the only example present on the XY stage. The diameters of the guide shafts are picked based on the engineering requirement of overall stage stiffness using Equation 8.

The other type of parametric design could be called 'geometrically driven parametric design'. Examples of this include the size of bushing used in the carriage, which must adapt based on the diameter of the guide shafts. And when the bushing sizes change, the reamed holes into which they're press-fit must also adapt. Etc, etc…

Figure 34 shows how the engineering parameters are entered into the design system.

	A	B	C	D	E
1	**INPUTS**				
2			UNITS	METRIC	UNITS
3	TABLE STIFFNESS	004E+06	N/m		
4	X TRAVEL	6	in	0.1524	m
5	Y TRAVEL	6	in	0.1524	m
6					
7					
8					

Figure 34: Engineering Parameter Inputs

It is also possible for the two design modes to become mixed. For example, the shaft lengths are based on both a performance parameter, stage travel, and geometric dimensions such as the widths of the carriage and support blocks. In practice the shaft lengths must be computed before the shaft diameters can be calculated using Equation 8.

Figure 35 is a screenshot showing how non-parametric dimensions like the carriage and support block widths are provided manually.

		UNITS	METRIC	UNITS
CARRIAGE				
CROSS-SECTIONAL WIDTH	4	in	0.1016	m
WALL THICKNESS	0.1875	in		
X AXIS CENTERLINE	0.625	in		
Y AXIS CENTERLINE	0.625	in		
WALL-TO-SHAFT CLEARANCE	0.0625	in		
X AXIS SHAFT SEPARATION	2.75	in		
Y AXIS SHAFT SEPARATION	2.75	in		
SHAFT-TO-SHAFT CLEARANCE	0.125	in		
OVERALL LENGTH	2.125	in		
ENDBLOCKS				
WIDTH	1.25	in	0.03175	m
SHAFT OVERHANG	0.125	in		
X AXIS LENGTH	4.5	in		
Y AXIS LENGTH	7	in		
TABLE				
LENGTH	13.75	in		
BASE				
OUTSIDE SUPPORT BLOCK SEP.	13.75	in		
L-BRACKET CLEARANCE	0.25	in		
OVERALL LENGTH	18	in		

Figure 35: Non-Parametric Dimensions

Figure 36 is another screenshot showing Equation 8 being evaluated based on both engineering parameters and geometric dimensions. The result is a minimum shaft diameter which satisfies the requirements of Figure 34.

	A	B	C	D	E
7					
8	X AXIS WORKING LENGTH	0.254	m	10	in
9	Y AXIS WORKING LENGTH	0.254	m	10	in
10					
11	X AXIS MINIMUM DIAMETER	0.013831972	m	0.544565829	in
12	Y AXIS MINIMUM DIAMETER	0.013831972	m	0.544565829	in
13					
14	X AXIS STIFFNESS	005E+06	N/m		
15	Y AXIS STIFFNESS	005E+06	N/m		
16	ACTUAL TABLE STIFFNESS	003E+06	N/m		
17					

Figure 36: Evaluation of Beam Bending Formula

Although Equation 8 is continuous, the available components are not. If a given stiffness calls for a shaft diameter of 0.694", the design software know that the smallest available shaft which meets this criterion is 0.75". Microsoft Excel's "lookup" function was used extensively to find components in a table which met the engineering requirements imposed upon them. Figure 36 shows the lookup table used to select a shaft diameter which satisfies the results of Figure 36.

18	**SUITABLE COMPONENTS**									
19		DIAMETER	UNITS	METRIC	UNITS	LENGTH	UNITS	METRIC	UNITS	McMASTER #
20	X AXIS PRECISION SHAFT	0.625	in	0.015875	m	14	in	0.3556	m	6061K111
21	Y AXIS PRECISION SHAFT	0.625	in	0.015875	m	14	in	0.3556	m	6061K111
22										
23										
24										
25	**AVALIABLE COMPONENTS**									
26										
27	McMASTER CARR PART #s	0.875	0.75	0.625	0.5	0.375	0.25			
28	30	**6061K338**	**6061K545**	**6061K123**	**6061K79**	**6061K82**	**6061K81**			
29	24	**6061K337**	**6061K54**	**6061K151**	**6061K53**	**6061K52**	**6061K51**			
30	20	**6061K456**	**6061K448**	**6061K441**	**6061K433**	**6061K425**	**6061K417**			
31	18	**6061K336**	**6061K44**	**6061K141**	**6061K43**	**6061K42**	**6061K41**			
32	16	**6061K455**	**6061K447**	**6061K439**	**6061K432**	**6061K424**	**6061K416**			
33	15	**6061K334**	**6061K345**	**6061K122**	**6061K335**	**6061K325**	**6061K315**			
34	14	**6061K454**	**6061K446**	**6061K438**	**6061K431**	**6061K423**	**6061K415**			
35	12	**6061K333**	**6061K34**	**6061K131**	**6061K33**	**6061K32**	**6061K31**			
36	10	**6061K114**	**6061K113**	**6061K112**	**6061K109**	**6061K108**	**6061K107**			
37	9	**6061K332**	**6061K24**	**6061K121**	**6061K23**	**6061K22**	**6061K21**			
38	8	**6061K106**	**6061K105**	**6061K104**	**6061K103**	**6061K102**	**6061K101**			
39	7	**6061K453**	**6061K445**	**6061K437**	**6061K429**	**6061K422**	**6061K414**			
40	6	**6061K331**	**6061K14**	**6061K111**	**6061K13**	**6061K12**	**6061K11**			
41	5	**6061K452**	**6061K444**	**6061K436**	**6061K428**	**6061K421**	**6061K413**			
42	4	**6061K451**	**6061K443**	**6061K435**	**6061K427**	**6061K419**	**6061K412**			
43	3	**6061K449**	**6061K442**	**6061K434**	**6061K426**	**6061K418**	**6061K411**			
44										
45										
46										
47	McMASTER CARR COST	0.875	0.75	0.625	0.5	0.375	0.25			
48	30	**26.88**	**21.88**	**17.46**	**14.31**	**13.27**	**11.21**			
49	24	**22.6**	**18.62**	**14.31**	**11.27**	**10.86**	**8.06**			
50	20	**20.35**	**17.19**	**13.2**	**10.36**	**9.84**	**8.3**			
51	18	**16.95**	**13.94**	**10.95**	**8.42**	**7.97**	**6.68**			
52	16	**15.97**	**13.77**	**10.57**	**8.29**	**7.87**	**6.63**			
53	15	**15.28**	**11.88**	**9.02**	**7.22**	**6.64**	**5.57**			
54	14	**14.26**	**11.79**	**9.08**	**7.25**	**6.9**	**5.81**			
55	12	**11.98**	**9.3**	**7.2**	**6.23**	**5.31**	**4.47**			
56	10	**9.96**	**8.43**	**6.48**	**4.8**	**4.54**	**4.04**			
57	9	**8.97**	**6.94**	**5.4**	**4.25**	**3.97**	**3.53**			
58	8	**7.38**	**6.24**	**4.8**	**3.84**	**3.64**	**3.38**			
59	7	**6.98**	**5.88**	**4.54**	**3.89**	**3.9**	**3.88**			
60	6	**5.99**	**4.64**	**3.58**	**3.42**	**3.26**	**2.74**			
61	5	**4.99**	**4.21**	**4.22**	**3.9**	**3.88**	**3.88**			
62	4	**3.98**	**3.87**	**4.33**	**3.9**	**3.88**	**3.88**			
63	3	**3.88**	**3.89**	**4.31**	**3.9**	**3.88**	**3.88**			
64										

Figure 37: Shaft Lookup Table

In addition to finding components which satisfy the engineering parameters, lookup tables are also used to automatically generate a bill of materials. Figure 37 shows an automatically generated BOM for the XY stage. The McMaster-Carr order which delivered components for the prototype stage of Figure 30 originated from this spreadsheet.

	A	B	C	D	E	F
1	LINE #	McMASTER #	QUANTITY	UNIT COST	EXTENDED COST	DESCRIPTION
2	1	6061K111	2	$3.58	$7.16	PRECISION SHAFT 0.625in. DIAMETER x 14in. LONG
3	2	6061K111	2	$3.58	$7.16	PRECISION SHAFT 0.625in. DIAMETER x 14in. LONG
4	3	9440T26	4	$1.94	$7.76	Dry-Lube SAE 841 Bronze BUSHING 0.625 ID 0.75 OD
5	4	9440T26	4	$1.94	$7.76	Dry-Lube SAE 841 Bronze BUSHING 0.625 ID 0.75 OD
6	5	4638K511	8	$0.14	$1.12	HEX SOCKET PIPE PLUG 1/8" NPT
7	6	99030A305	1	$17.80	$17.80	ACME LEADSCREW 1/2" - 10
8	7	5909K31	4	$2.53	$10.12	THRUST BEARING CAGE ASSEMBLY 1/2" ID
9	8	5909K44	8	$0.85	$6.80	THRUST BEARING PRECISION WASHER 1/2" ID
10	9	9712K74	6	$0.40	$2.41	BELLEVILLE WASHER 0.505" ID 260 LBS
11	10	91083A033	8	$0.04	$0.35	STEEL WASHER 17/32" ID 1-1/16" OD 0.074" THK
12	11	94815A107	4	$2.22	$8.88	ACME HEX NUT 1/2"-10
13	12	89955K45	1	$8.31	$8.31	STEEL STRUCTURAL TUBING 0.375" OD 0.209" ID 6' L
14	13	91251A360	8	$0.28	$2.21	SOCKET HEAD CAP SCREW 10-32 x 3.000"
15	14	6208K13	2	$24.00	$48.00	HELICAL SHAFT COUPLER 0.25" - 0.25"
16	15	6338K461	2	$0.92	$1.84	SAE 841 BRONZE BUSHING 0.375" ID 0.625" OD
17	16	98812A053	1	$6.74	$6.74	THREADED BRASS ROD 3/4"-16 12"L
18	17	94830A436	2	$2.74	$5.48	FLEXLOC LOCKNUT 3/4"-16
19	18	6546K261	1	$5.63	$5.63	ALUMINUM TUBE 4"x4" 0.1875" WALL THK 12"L
20	19	6546K261	1	$15.11	$15.11	ALUMINUM TUBE 1.25"x2.5" 0.125 WALL THK 36"L
21	20	1630T14	1	$13.25	$13.25	ALUMINUM CHANNEL 5" BASE 2.25"LEG
22	21	6381K534	2	$2.34	$4.68	SAE 660 BRONZE BUSHING 0.625 ID 0.75 OD 1.123"L
23	22	8975K493	1	$31.50	$31.50	ALUMINUM BAR 0.75"x5"x36"
24	23	91251A621	4	$0.17	$0.69	SOCKET HEAD CAP SCREW 3/8-16 x 0.625"
25	24	92220A172	6	$0.15	$0.89	LOW HEAD CAP SCREW 10-32 x 0.375"
26					$221.64	

Figure 38: An Automatically Generated Bill of Materials

In addition to a BOM, engineering drawings are also automatically generated by SolidWorks.

Figure 39 shows the solid model which is updated by the design spreadsheets in response to changes in the engineering parameters.

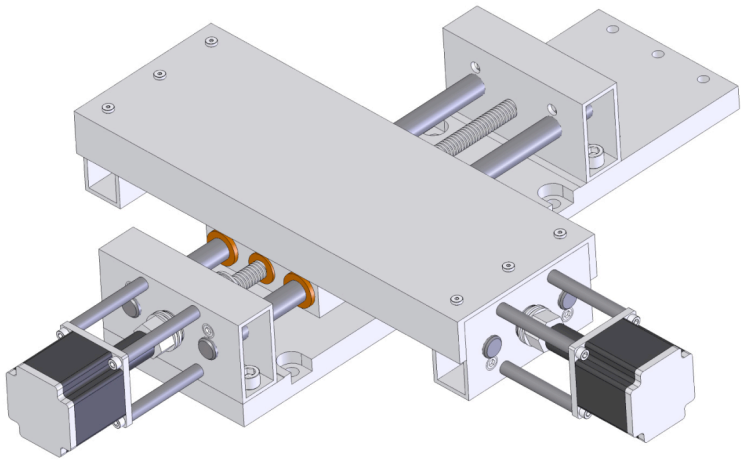


Figure 39: Engineering Driven Parametrically Designed Solid Moel

Concluding Remarks

Engineering driven parametric design is a valuable but under-used tool for making mechanical designs reusable. This project has demonstrated that the method can be successfully used to design and create a practical and semi-universal component of a rapid prototyping machine.

One of the major difficulties in the design process has been the interface between Microsoft Excel and the CAD program SolidWorks. CAD manufacturers might consider integrating an engineering design tool into their packages in addition to the geometric design tools already present. The interface might be something like this: "engineering blocks" containing commonly used equations such as those for beam bending or torsional shaft yielding could be strung together visually on a worksheet. Inputs could be picked depending on which parameters were to be driven. For example, Equation 8 could be expressed as a block where the user picks either stiffness or shaft diameter as the input. The output of the blocks could be displayed on the worksheet or used to drive dimensions in the solid model.

CONCLUSIONS

The overwhelming sense of resonance between the world and the mind triggered by the act of making things can be a powerful force of positive world change. I would posit, without offering any proof, that the pivotal contributions made by individuals to humanity were driven by excitement rather than self-sacrifice. By empowering ordinary people to create nearly anything they can envision and to experience the accompanying exhilaration, the field of rapid prototyping has the potential to foster creativity and innovation on a never-before-seen scale.

If rapid prototyping is to live up to its promise, it must exceed the bounds of industrial prototyping shops and enter the home. A first step in this process is to enable interested individuals to make their own rapid prototyping machines. This thesis has sought to lower the barrier-to-entry by introducing several ways of rapidly prototyping rapid prototyping machines. A new control architecture, consisting of a real machine controlled by a virtual machine over a distributed network, was introduced. Benefits include easier configuration and greater flexibility. A method of creating reusable mechanical designs was also demonstrated, which will hopefully aid machine designers in creating useful "libraries" of mechanical designs in the same way that libraries of code save programmers from duplicating efforts.

It is my sincere hope that the work contained in this document will help give the field of rapid prototyping a shove along the path towards ubiquity.

REFERENCES

1. Gershenfeld, N.: "FAB: The Coming Revolution On Your Desktop – from Personal Computers to Personal Fabrication." Basic Books. (2005).

2. Gershenfeld, N. and Krikorian, R.: "The Internet of Things." Scientific American 291, pp. 76-81 (October, 2004).

3. Slocum, A.H.: "Precision Machine Design." Society of Manufacturing Engineers. (1992). pp. 691-693.

4. Jones, Douglas W.: "Control of Stepping Motors: A Tutorial." http://www.cs.uiowa.edu/~jones/step/index.html . (1995). Accessed May, 2008.

5. Avallone, E.A. and Baumeister, T. : "Mark's Standard Handbook for Mechanical Engineers." McGraw-Hill Professional. Second Edition. (1998).

6. Oberg, E. and Jones, F. : "Machinery's Handbook." Industrial Press Inc. Twenth Sixth Edition. (2000). pp. 244.