SHERLINE PRODUCTS
INCORPORATED 1974

Operating Instructions for the Sherline Vertical Milling Machine CNC System

Sherline Linux (Ubuntu v. 12.04) with LinuxCNC version 2.6.11, updated 10/9/19

P/N 8540/8541, 8020/8021, 8600/8601, 8620/8621

EMC2 is now referred to as LinuxCNC

PRECAUTIONS

1. Do not connect or disconnect stepper motors when the driver box is powered up. Always turn off the power to the driver box before plugging unplugging a stepper motor. Improperly connecting or disconnecting a motor can damage it.
2. Before turning on the computer and booting up LinuxCNC, make sure the power switch for the stepper motors on the side of the computer (or on the 8760 driver box) is in the "OFF" position. After LinuxCNC is fully loaded it is then OK to turn on power to the stepper motors. Controls within LinuxCNC prevent overloading of the power supply when multiple motors are powered up at the same time, but without this safeguard motors or drivers can be damaged.
3. Before turning the computer on for the first time, confirm the 115/230 Volt power switch on the back of the computer is in the correct position. If it must be changed, note that there is a second power selection switch inside the computer case on the driver board power supply that must also be changed. Contact Sherline before removing the tamper-proof label to open the case or you may void the warranty. (The spindle motor will automatically adapt to any current from 100 to 240 VAC, so no transformer or switching is needed for it.)
4. If all stepper motors stop running at the same time, turn off power to the stepper motors (switch on side of computer or back of 8760 driver box), wait a few seconds and turn power back on. A protection circuit was introduced on 2009 and later driver boards that will shut down power to the stepper motors in case of an electrical surge or anomaly. Turning power off and back on should reset the circuit. Individual replaceable fuses on each axis were added in October, 2011. See the troubleshooting guide for replacement instructions.

NOTE: Effective January 2009, Sherline computers no longer come with a floppy drive. A USB flash drive is included instead to facilitate file transfers.

Finding the most current instructions

The most up-to-date version of these instructions can always be found on the Sherline website at www.sherline.com/cnc-instructions. The following instructions refer to systems utilizing the Ubuntu version of Linux and the LinuxCNC g-code control program. Older instructions for using the Debian version of Linux 4.xx and the EMC distributed by Sherline starting January1, 2005 or the initial versions of 2.xx Redhat Linux will remain posted there as well.

Why we switched to Ubuntu and LinuxCNC

The Debian version of Linux, along with the EMC (Enhanced Machine Controller) software has been distributed by Sherline since January, 2005. It is thoroughly debugged and stable, and if you only use a mill, it will still work fine. We have decided to offer the Ubuntu version of Linux primarily because it now supports a broader range of hardware. Its interface will also be a little more comfortable for those familiar with Windows. For example, you now double-click on files to open them instead of single clicking and there is no longer a "mount" and "unmount" command for using a USB flash drive or disc drive. Just click on the icon and it opens as it would in Windows. Small differences aside, it also allows us to upgrade from EMC to LinuxCNC. The graphic interface of LinuxCNC again looks very much like the original EMC, but now includes a lathe-specific program in addition to the mill program. It also supports several additional G- and M-codes. We feel these advances make it worthwhile for us to change over, and we think you will appreciate the additional features as well. As before, the operating system and control program are provided free of charge to Sherline CNC users. The most important advancement in LinuxCNC for Sherline users is LinuxCNC allows the use of sub programs!

CNC for Lathe Users

Purchasers of Sherline CNC lathe systems can also use these instructions, as the basics of CNC apply regardless of the machine being run. Keep in mind that lathe operations will require somewhat different g-code simply because of the way parts are made on a lathe. While primarily written for the mill user, the instructions on how to write g-code are more or less universal and can be applied to a lathe or a mill, keeping in mind the way each machine works. More

specific instructions on the use of LinuxCNC and also the lathe portion of the program can be found in the LinuxCNC user manual. A copy of the manual is part of the software pre-loaded onto your Sherline CNC computer and can be found under the main menu at *Applications>CNC>LinuxCNC User Manual*. It is also available on-line at http://linuxcnc.org/docs/2.6/pdf/LinuxCNC_User_Manual.pdf.

Use of non-Sherline Programs

Sherline cannot be responsible for the support of software or hardware products not designed or sold by Sherline. For questions on Windows® CAD, CAD/CAM, control or free utility programs, please contact the provider of the software. It would be impossible for Sherline to guarantee that g-code generated by any particular CAD/CAM program will run on EMC or LinuxCNC without modification. A thorough knowledge of g-code programming will always be useful to eliminate problems caused by post processors.

Safety Rules for Power Tools

1. **Know your power tool**—Read the owner's manual carefully. Learn its application and limitations as well as the specific potential hazards peculiar to this tool.
2. **Ground all tools**—If a tool is equipped with a three-prong plug, it should be plugged into a three-hole receptacle. If an adapter is used to accommodate a two-prong receptacle, the adapter wire must be attached to a KNOWN GROUND. Never remove the third prong.
3. **Keep guards in place**—and in working order.
4. **Remove adjusting keys and wrenches**—Form a habit of checking to see that keys and adjusting wrenches are removed from the tool before turning on any machine.
5. **Keep work area clear**—Cluttered work areas and benches invite accidents.
6. **Avoid a dangerous work environment**—Do not use power tools in damp or wet locations. Keep your work area well illuminated.
7. **Keep children away**—All visitors should be kept a safe distance from the work area.
8. **Make your workshop "kid-proof"** —with padlocks, master switches or by removing starter keys.
9. **Do not force a tool**—Do not force a tool or attachment to do a job for which it was not designed. Use the proper tool for the job.
10. **Wear proper apparel**—Avoid loose clothing, neckties, gloves or jewelry that could become caught in moving parts. Wear protective headgear to keep long hairstyles away from moving parts.
11. **Use safety glasses**—Also use a face or dust mask if a cutting operation is dusty.
12. **Secure the work**—Use clamps or a vise to hold work when practicable. It is safer than using your hand and frees both hands to operate the tool.
13. **Do not overreach**—Keep your proper footing and balance at all times.
14. **Maintain tools in top condition**—Keep tools sharp and clean for best and safest performance. Follow instructions for lubrication and changing accessories.
15. **Disconnect tools**—Unplug tools before servicing and when changing accessories such as blades, bits or cutters.
16. **Avoid accidental starting**—Make sure the switch is "OFF" before plugging in a power cord.
17. **Use only recommended accessories**—Consult the owner's manual. Use of improper accessories may be hazardous.
18. **Turn the spindle by hand BEFORE switching on the motor**—This ensures that the workpiece or chuck jaws will not hit the lathe bed, saddle or crosslide, and also ensures that they clear the cutting tool.
19. **Check that all holding, locking and driving devices are tightened**—At the same time, be careful not to over tighten these adjustments. They should be just tight enough to do the job. Overtightening may damage threads or warp parts, thereby reducing accuracy and effectiveness.
20. **Don't use your lathe for grinding**—The fine dust that results from the grinding operation is extremely hard on bearings and other moving parts of your tool. For the same reason, if the lathe or any other precision tool is kept near an operating grinder, it should be kept covered when not in use.
21. **Don't let long, thin stock protrude from the back of the spindle**—Long, thin stock that is unsupported and turned at high RPM can suddenly bend and whip around.
22. **Wear your safety glasses**—Foresight is better than NO SIGHT! The operation of any power tool can result in foreign objects being thrown into the eyes, which can result in severe eye damage. Always wear safety glasses or eye shields before commencing power tool operation. We recommend a Wide Vision Safety Mask for use over spectacles or standard safety glasses.
23. **Checking/changing computer power supply voltage settings**—Sherline attempts to ship each CNC computer set to the proper voltage for the customer's country. Customers outside the USA should check this setting to confirm it is correct before plugging in the computer for the first time. If you need to change the voltage setting, there are two switches. You must change both the 115V/230V power switch on the back of the computer and the switch on the side of the driver board power supply inside the computer case. For proper procedure, see Part I, System Components and Connections further on in these instructions.
24. **Static electricity can damage your computer**—If the case must be opened, be sure you are properly grounded before touching any components inside your computer. A spark of static electricity can damage delicate circuits

in some components in your computer. Either wear an approved static electricity grounding strap around your wrist or touch the case of the computer with one hand before touching any components with your other hand.

25. **Be sure you know what will happen BEFORE pushing the [START] button**—Run your program in [BACKPLOT] mode before running the actual part to make sure it will run the path your are expecting. Make sure your machine is in the proper home position before starting the operation.

Electrical Connections

The power cord supplied is equipped with a 3-prong grounding plug that should be connected only to a properly grounded receptacle for your safety. Should an electrical failure occur in the motor, the grounded plug and receptacle will protect the user from electrical shock. If a properly grounded receptacle is not available, use a grounding adapter to adapt the 3-prong plug to a properly grounded receptacle by attaching the grounding lead from the adapter to the receptacle cover screw.

Current Requirments

The electrical circuit designed into the spindle motor speed control of your lathe or mill reads incoming current from 100 to 240 volts AC and 50 or 60 Hz and automatically adapts to supply the correct DC voltage to the motor. As long as you have a properly wired, grounded connector cord for your source, the machine will operate anywhere in the world without a transformer. This has been true for all Sherline machines built since 1994.

Electronic Filter for CE Approval

Sherline now offers an in-line electronic filter between the DC motor and speed control and the incoming AC current that suppresses emissions to meet CE standards. All machines shipped to countries requiring CE approval will be supplied with these filters already installed and shipping boxes properly marked. This adds an additional $45.00 to the cost of the lathe or mill, but it does allow us to apply the CE label to our product. If a machine is purchased without this filter, we cannot guarantee that it will be able to make it through customs in the subject country. Included with the installed filter is a certificate of CE compliance. See www.sherline.com/45500 for more details on the filter.

Stepper Motor Cord Precautions

Do not disconnect the cable at the stepper motor end unless absolutely necessary. The connector was designed for limited use and can be easily damaged. When connecting and disconnecting the motors, always use the round plugs that connect to the driver box cables. Do not pull on the wires to unplug the connectors—Grip the plugs themselves. We also recommend use of a plastic tie wrap in in place of one of the motor mounting screws to provide strain relief on the cable to protect this connection.

Lubrication

CNC machines require more attention to keeping the leadscrews properly lubricated than manual machines. When you crank the handles by hand you can quickly realize that a slide is going dry because you can sense the results of the extra drag. A CNC machine will not complain and will always work with the instructions that you gave it. You tell it to run, and it will die trying to please you. The Y- and Z-axis screws can be easily oiled, however, the X-axis screw needs special attention. You can't see it because it is located under the mill table. The best way I have found to lubricate it is to move the mill table all the way to the right, put some oil on your fingertips and transfer it to the leadscrew. As the table is moved back to the left, oil will find its way to the internal brass nuts. Remember, that any type oil is far better than none.

Starting in early 2010, all Sherline cnc mill saddles were fitted with an oil reservoir that feeds lubrication to the X and Y leadscrews. This should be refilled every few hours during cnc operation. (See www.sherline.com/oiler)

The slides also require constant oiling. I believe these oiling procedures should be done after every four hours of operation. For more on lubrication see page 6 of the Sherline Miniature Machine Tools Assembly and Instruction Guide (7th ed) that came with your mill.

Decimal display and machine accuracy

The display panel allows dimensions to be entered to four decimal places in order to maximize accuracy of the internal computer calculations. This is not meant to imply that you should expect to achieve machine output to that degree of accuracy. In reality, you should expect a quality machine in this price range to produce repeatability accuracy of 0.02 mm with 1mm pitch leadscrews or 0.001 with 20 TPI leadscrews.

CAUTION—Manual Adjustments Using the Handwheels

MANUAL MODE: A DC motor acts like a generator when cranked by hand. When using the machine extensively in full manual mode you must first turn off the power supply to the motors and then physically disconnect the stepper motors from the driver board cables to prevent the generated voltage from damaging components on the board. Cover the ends of the motor connecting plugs with masking tape to protect them from debris. When reconnecting, make sure the plugs are clean and fully seated. Constant plugging and unplugging is not good for the connections, so avoid doing so unless absolutely necessary.

CNC MODE: When operating in CNC mode you do not need to disconnect the stepper motors when hand cranking slowly and over short distances to manually position the slides. The motors are locked up when power is on, so the power switch to the drivers must be in the OFF position. For longer positioning travels it is easier to use the Jog Mode and move to approximate position electronically. Final adjustment can then be made with the handwheels (with power turned OFF). To keep from generating a potentially harmful back-current to the driver board, do not turn the handwheels manually faster than about 1 rotation per second when the stepper motors are connected to the driver board.

CNC System Warranty

If within one year from the date of purchase of a new Sherline CNC system any tool or component fails due to a defect in material or workmanship, Sherline will repair or replace it free of charge. In addition, it has always been our policy to replace at no cost all parts, regardless of age, which are determined to have been incorrectly manufactured or assembled and have failed due to this cause rather than because of improper use or excessive wear caused by continuous use in a production environment. In cases such as this, Sherline will inspect the machine or part and will be the sole judge of the merit of the claim.

Freight charges for returning a machine are not covered. **Save the original packaging material to use if the machine or computer must be returned to the factory for service**. Sherline will not participate in insurance claims where the sender didn't make an honest effort to package the items to prevent damage caused by normal shipping stresses or repair said damage without charging the sender.

Merchandise that has been abused, misused, improperly maintained or insufficiently lubricated is not subject to warranty protection. Proof of date of purchase and dealer name may be required for service under this warranty. **NOTE: WD-40 is for rust protection, it is NOT a suitable lubricant!**

Specific exclusion for wear due to production use

Although many Sherline tools have found there way into production environments, they have been designed for intermittent use in home shops. It would be unreasonable to expect a Sherline machine to stand up to the rigors of continuous production use. Continuous use of stepper motor driven leadscrews in a production environment can introduce wear that would be impossible to produce manually. Therefore, slides, gibs, leadscrews, leadscrew nuts and bearings that are subject to high wear due to continuous usage are not covered under this one-year warranty.

Non-warranty service and machine tune-ups

Sherline machines are easy to service, and all replacement parts are available from the factory. However, if in the future you wish to return your machine for any reason to have Sherline repair or adjust it, we will be glad to do so. Our shop rates are very reasonable and our production people have all the proper tools and fixtures to repair or service your machine quickly and return it to factory new condition.

Returning machines or parts for warranty or non-warranty repair

Before returning any tools or parts for repair, please call first and obtain a Return Material Authorization (RMA) number so that we can process your order immediately upon receipt. Remove all non-essential parts from the machine to save shipping weight and return only the portion to be repaired. Wrap and package all parts securely so they cannot move around in shipment. We recommend that heavy parts be double-boxed. Include a brief written description of the desired repair and your return address and phone number or email address inside the box so we can contact you if questions arise. Repairs are normally return-shipped the next workday after they are received.

Technical Support

If you have a physical or electrical problem with a machine, component or accessory manufactured by Sherline, please feel free to contact us at 1-800-541-0735, 1-760-727-5857 or sherline@sherline.com. If you have technical questions regarding the functions Linux or EMC that are not answered in these instructions, the Linux group maintains an excellent website at www.linux.org that has a search function and a lot of helpful information. Remember that Sherline hasn't charged you a single penny for the Sherline version of the LinuxCNC program even though we have spent a considerable amount of money and time adapting the software to the Sherline mill. Our CNC instructions also include a free basic course in the use of g-code. Beyond that, we recognize that many of our customers are new to machining and to CNC, and we will do our best to put callers on the right track to the information they need to get their new CNC system up and running. Learning to use g-code and learning to cut metal beyond what we provide in our instruction manuals must ultimately be the responsibility of the machinist, but we will help where we can.

A Short History of CNC

In order to shorten the instructions and get right to the operation of the machine, this introduction has been moved to the Sherline website at http://www.sherline.com/CNChistory. If you are new to CNC we urge you to read this introduction first, as it will give you a good background on how we got to where we are today and why the instructions are presented the way they are. The following instructions are divided into Part 1, which covers the basics of connecting your computer and understanding the LinuxCNC interface and Part 2, which provides workbook examples to walk you through the basics of using g-code.

Part 1

A different way of learning

Long ago I became aware of the success rate of people who, when left alone with their video recorder and an instruction manual, failed to pass the test and had to stay up late to turn on their recorder to record a show they wanted to save. From their own statistics you would think manufacturers would have attempted to teach in a different manner, but they don't. The products get more complex, making the instructions more complex, and the poor consumers of these products are at a loss. I don't teach using these methods because they obviously don't work. With my instruction you won't be sure if you're reading a novel or an instruction manual, but I'm sure you'll be able to program and make a part when you get to the end.

The first instructions I completely wrote for my tools (I had written instructions about radio control equipment and rules for RC events in the past) were how to cut a screw thread using a rather crude device I designed (but which works well) for a Sherline lathe. I was surprised how few calls on how to use it came in after I sold my first 200, because it is a bit of a complicated thing to do. I then started to wonder if anyone was using it, and you'll never know how relieved I was when a customer came in with a bunch of parts that he made with threads on them.

I knew I was on to something and have always written in this style ever since because it works. It had to work because at that time I was the only one at my company who could answer questions of this type, and I didn't have the time. The fact that I'm a self taught person probably has something to do with it. I even went on to write a couple of books in this style that were well received.

A new challenge

I remember getting a call from Sears about my folksy style of writing that didn't follow Sears' instruction guidelines about the screw cutting attachment. I asked him how many calls they had received about how to cut threads using it. He answered, "None." I asked him who there could answer a technical question of this type. He answered, "Nobody." I said, "With this being the case, don't you think you should leave well enough alone?" I never heard from him again. Right now, I'm looking at these instructions as one of the biggest challenges I've ever taken on as an author, and I don't plan to fail. The average instruction manual written on this subject is two to three hundred pages of very technical reading. You'll never crack a smile reading one of these manuals. I'll do it in less than fifty.

The only students I hired who learned CNC had good teachers, not good training manuals. In fact, I can't think of any employee in the last 30 years that I've ever hired who learned CNC programming and how to run these complex machines on his own. You will! I had to learn on my own because I was so broke after I bought my first NC I was out of choices. I learned by staying up late, working my ass off and making costly mistakes. I would have given my left you-know-what if I had something like this to read rather than reading the junk written by professors trying to prove how smart they were to other professors rather than attempting to teach their own students.

Sherline customers want to build components

I saw the customer who was going to read these instructions as a person who wasn't particularly interested in computers or gadgets and was only interested in making parts. I believed they were curious about CNC and wanted to learn about it without spending too much money. They probably haven't tried to learn how to do something this complicated in the last 20 years. Remember that CNC is a new way of doing and thinking about the machining process, and I believe I'll help get your brain in tune with what you are about to learn with my ramblings.

Learning while you work

A student of this class could be already working at the trade. Words of advice for this group: You can't afford to have idle thoughts as you load parts on machines that other people set up and wrote the programs for unless you are satisfied to load parts for the rest of your life.

First study the mechanical part of the setup; things like how the part is held and the type of cutting tools used and how long they last. You have to show some initiative and start analyzing the program you are running and ask intelligent questions. Notice, I said intelligent questions. Bosses are usually smarter than you think they are, and they easily know when some when clown is pulling their chain. Don't ever make suggestions unless you are sure you are right. A better way to make a suggestion of this type is to ask them just why the job is set up or run the way it is rather than your way. That way you're not threatening them with your question, and they'll appreciate a chance to teach you if you are wrong. If you find out you were wrong, it gives you a way out by saying that you were sure that they must have had a good reason. It wouldn't hurt to let that rule work its way into your personal life.

When you start studying their programs, you'll find them written like a story. Just like a book, the better the author, the easier the program is to read. Each tool used will have its own chapter. (In the real world of machining you'll run few machines without automatic tool changers.) Study these chapters one at a time and watch what the machine does as the program advances. In case you didn't realize it, I'm also instructing all you hobbyist out there on how to organize your programs and setups. Plan them like someone else is going to run the job and pick apart your own work. If you don't, you might find yourself trying to machine your vise off the table.

Getting you into the learning mode

I also felt that people of this type would want to read something about what they are in store for before they started. I wanted my customers to understand just how difficult CNC programming can be to learn at the start and just how interesting it can be at the same time. In other words, I felt "Joe home machinist" doesn't stand a chance unless he's inspired to do so, and in my own way I'm trying right now to inspire you to learn how to program and use these marvelous new CNC machines. I know how easy it is for you to go back to the way you already know or forget the whole idea. I don't want customers who think it'll be easy to learn because they'll be unsatisfied from the beginning and take out their anger on Sherline.

You're about to learn how to control your robot

These aren't instruction about operating a simple single system like a VCR, but instructions about controlling a very complicated device. You are not operators of these marvelous machines; you are masters telling your robot what to do, and you, the controller, have unlimited choices to make. I want to get this concept through to that average customer I mentioned, yet I know that only one out of five professional machinists I know seems to really understand this point.

I'm going to be a student in my own class

I was shocked at just how much I had forgotten about the process of CNC programming. I probably know as much as anybody of what you can do with CNC machines, because I own at least twenty modern machining centers, yet I haven't really written any code in fifteen years. After thinking about it for a couple of days while unsuccessfully attempting to write a simple program I decided this is a good thing. I also decided that I'd learn this from the Linux site on my own and not use the books that were available to me. I believed that if I wrote the instructions as I learned how to program EMC myself I'd write better instructions; after all, all you have to be is one page ahead of your students to teach. Just joking, teachers.

Something for nothing

Sherline Products Inc. is not charging you for this LinuxCNC program, and it is free because of work done by NIST (National Institute of Science and Technology), a government funded organization. The EMC (Enhanced Machine Controller) was written by the NIST and is a very sophisticated program. We all now have the benefit of millions of dollars of programming along with the source code, if needed, simply by asking.

We All Owe Thanks to the EMC Group

At the time this program was written it was thought that only very advanced programmers and scientists would be using a program such as this, and the average person stood little chance of making it functional. Fortunately for us, a lot of very smart people before us, donating their time, started developing a programming interface that allowed both businesses and hobbyists to take advantage of what they had done without being engineers. I'm very pleased with what they have done. This group still works together through the Internet, and help is available through Internet chat groups. Many people have donated much of their time to make this system "user friendly."

The LinuxCNC program operates under the Linux operating system, which is also a free system, and it would be wise to operate it from a dedicated computer for this task. A dual boot system can lead to problems, because Microsoft code has been purposely written to be the dominant control of your computer whether you want it or not. I like the Linux concept of people working together where all have the benefit of any single person's work. For example, these instructions will have a Sherline link placed on the EMC for all to read whether they are Sherline customers or not. It will be my donation towards a successful system available to all. I'm learning how to program EMC by using the internet site put together by dedicated EMC users, so you can also fall back to their site for help. See http://www.linuxCNC.org/ .

This is going to be easy if you do it my way

The Linux operating system works pretty much like all the others. If you can find your way around a Mac or Windows computer, you will have no trouble navigating menus, opening, closing and copying files or saving and deleting items in Linux. The new Ubuntu version is the most Windows-like yet. When you turn on the computer, there will be four icons on the desktop. Depending on whether you have a lathe or a mill, inch or metric, choose the correct icon for your machine and double click on it. In a few seconds you'll get the graphical interface to the LinuxCNC. Well, that was easy. Now, after BS-ing you to death on how hard this was to do I'm going to show you how easy this can be, but only if you do it my way.

Computers have the patience of a saint

If you've had little computer experience, the first thing you will learn is that computers have the patience of a saint. If computers were more human they would probably like to toss their operators out the window as much as frustrated operators want to do the same to them. The only way you can keep a computer from giving you an error message is to not give it errors. The tricky part is computers don't always tell you the error to fix or how to fix it. Computers just tell you that there is an error of a certain type. This can become maddening, and you have to know when to take a break and walk away. Things of this nature are the worst when, for the sake of your own sanity, they should be the best. In many ways, learning programming is similar to learning machining. You can't take short cuts. If you find yourself going in the wrong direction, back up immediately.

Learn machining first

If you know little about machining and little about computers I'd suggest you first concentrate on learning about machining. How can you expect a machine tool and a computer to carry out your commands if you don't know the commands to give? Above all, don't start with a project so complex you don't stand a chance in hell of succeeding. Let each simple problem solved have its own rewards in knowing you just traveled another block further down the CNC machining super highway, route EMC.

NOTE: I have written a book called ***Tabletop Machining*** that is an introduction to metalworking at the small end of the size scale. Details can be found at www.sherline.com/product/5301-tabletop-machining/. If you are new to machining, you will find the information in this book quite helpful.

You will be learning the CNC language of the robot world

The standard "g" code method of programming is to simply replace the manual inputs into a machine (cranking the handles) with the commands that have become industry standards. This is the method you should learn before learning any other method of programming. It is the basic way that CNC machines "talk" to other CNC machines. Even if a simpler system were available today I still wouldn't teach it. It would be similar to teaching a language that few citizens of the world spoke. Programs written using g-codes can be read throughout the world. g-code standards came about very early in the game because this game was controlled by engineers rather that marketing people.

Contouring programs—another world

Programs that allow you to do contouring quickly become very complex. Today there are some affordable programs available for sophisticated hobbyists. These programs can create as many problems as they solve, because of the complexity of each program. Each program has to work with other programs, and each program has its own set of complex rules. If you think you can learn these programs easier than you can solve a couple of trig problems you're in store for a rude awakening, however, if you already know how to use 3D CAD programs you could quickly find these programs very useful. It should also be stated that it is the only way available to create contoured machined shapes.

What it costs in the industrial world

The basic method that these programs work together in the industrial world and the cost of these programs is: 1) The part is designed using a modeling program ($5000 to $10,000), 2) The modeling program is converted to a CNC program (another $5000 to $6,000) and 3) the resulting code is run on a CNC machine starting at ($25,000) that makes the part. The people who can operate these systems

are carefully chosen, trained and well paid. The fact is that any business has no choice except to use the program they purchased, because the programs cost so much and are so very necessary to remain competitive in today's world. They have to spend a great deal of money to accomplish this, while the hobbyist has a choice.

Today's low-cost programs were once complex, expensive programs

The business world understands how complex it is to implement these procedures; the hobbyist doesn't. Because the hobbyist may be purchasing a program that cost thousands of dollars just a few short years ago for a few hundred dollars, the programs themselves haven't become less complicated. They only cost less! In your mind you may be having visions of complex shapes being machined with little effort on your part while your two thousand-dollar investment turns out parts that the old masters of the machine trades would envy. Well my friend it's possible, but it's time to come down to earth and again think about the complexities of doing so.

Cutter compensation—what makes CNC work in a machining environment

I could have sneaked out the back door and only wrote instructions for a couple of simple moves without cutter compensation and let it go at that; but deep down inside, I'd have known I was cheating you out of the real benefits of CNC. I believed that if I started teaching you to use this marvelous programming tool right at the very start it will soon be second nature, and our class will leap-frog ahead of our competition with those thick boring books being taught by instructors I wouldn't let near the least expensive CNC machine in my shop.

If you take the effort to learn this method of programming at the start along with a little simple trig, you'll find it well worth the effort. Cutter offsets are the way you control size with modern CNC machines. When you start making more complex parts and you don't use cutter comp you'll have to change the basic numbers of your programs to make a part on size. Sounds simple enough until you find that every time you correct a number here you end up screwing up another number there. Take my thirty-five years of experience in dealing with CNC and accept this cutter offset stuff as a fact.

I can't help anyone who isn't willing to work hard

Back in ancient times the great mathematician Euclid told a Pharaoh who wanted to learn geometry the easy way that Pharaohs had to walk down the same road as scholars to education. In a sense I'm also stating to you that you will have to learn programming the same way as I'm about to, and at this time I'm only a couple of pages ahead of you. I am certain that if I sat by the phone and answered questions twenty-four hours a day, and I don't plan to, I couldn't help those who aren't willing to put forth the effort it takes on their own to learn to work with these marvelous tools.

Route EMC

This road that you are beginning to travel can become one of the most interesting roads you have every traveled, however, you must take the time to gather the facts that can make a trip so interesting. Instead of seeing great ancient cites on this road we'll be gaining the knowledge that will make our next stop even more interesting and more complex. We'll study a little and then apply these rules to your machine. We will first make simple moves that become more and more complex and at the same time more interesting as we travel on. Eventually we will reach Martinsville, a city on this road, where you will have so many choices as to the direction to travel that you'll be leaving me and heading out on your own to design and build things that you never thought possible in the past. The rules and facts that you will learn on this road will allow you to have a machine at your control that home machinists a few short years ago couldn't dream of. Take the time to enjoy this trip because you're on a road with no end if you have set your goals correctly.

Operating Instructions

Instructions: Mill vs Lathe

These instructions were written primarily for using the CNC system with a mill. A CNC-ready Sherline lathe can also be controlled using the same computer and driver system. You would simply plug the crosslide stepper motor to the X-axis cable and the leadscrew stepper motor to the Y- or Z-axis cable. Writing a program for the lathe is a little different in that you must remember that when you program a distance for the X-axis travel, the part diameter will be reduced by twice that amount. This is because you are taking that dimension off the radius of the part, which is one half of the diameter. Other than that, a little logic on your part should be able to accommodate the differences between the spinning cutting tool/moving part on a mill and a spinning part/moving cutting tool on a lathe. Because of the extra mass of a spinning chuck and part, extra caution should be taken to make sure your program is not going to cause any interference between crosslide or toolpost and the spinning chuck/part once it is running.

The Ubuntu/LinuxCNC version we now distribute does have a lathe-specific program that can be accessed by selecting the appropriate icon on your desktop. They are identified as "Sherline Lathe Inch" or "Sherline Lathe MM." Specific instructions for use of the lathe portion of the program can be found in Chapter 21 by accessing the LinuxCNC User Manual on your Sherline CNC computer from the main menu at Applications>CNC or on-line at http://linuxcnc.org/docs/2.6/pdf/LinuxCNC_User_Manual.pdf.

System Components and Connections

NOTE: Computer styles and specifications change often. The photos that follow are for illustration purposes and may not exactly represent the computer or the components that came with your system.

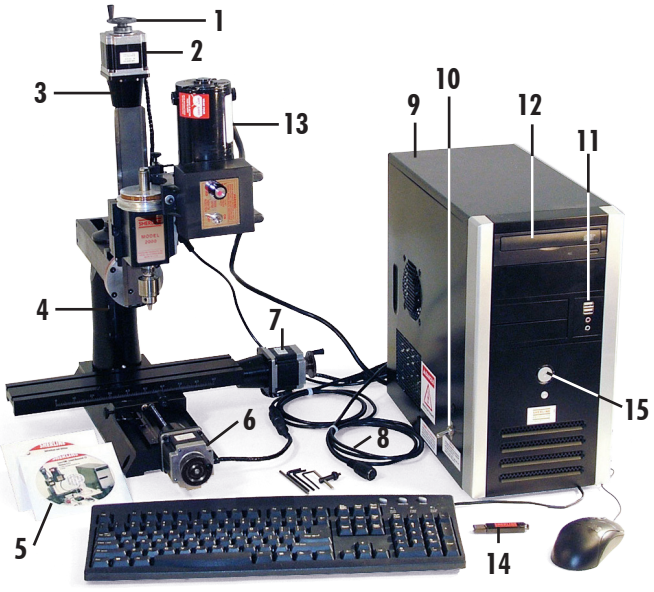


FIGURE 1.1

System components

1) 1-5/8" manual handwheel
2) Z-axis stepper motor
3) Stepper motor mount
4) Sherline vertical mill with standard accessories *(Model 2000 mill shown)*
5) Backup Linux/LinuxCNC installation CD, Sherline instructions CD
6) Y-axis stepper motor
7) X-axis stepper motor
8) Cable for optional A-axis (CNC rotary table) connection
9) Computer with keyboard, and mouse
10) On/Off switch for CNC driver power supply
11) USB drive ports (front)
12) DVD-RW drive
13) DC spindle motor
14) Included USB Flash Drive
15) Power On/Off (Smaller button below is "Restart" button)

FIGURE 1.2

Connections on back of your computer:

1) On/Off switch
2) 115/230V computer power supply voltage selection switch* (Red)
3) Power cord to wall outlet
4) Keyboard connector (Purple)
5) Mouse connector (Green)
6) Ethernet cable connection
7) USB ports, rear panel (more on front panel)
8) Monitor connections (a. HDMI port, b. Serial port [9-pin], c. VGA port [15-pin])
9) Output cables to stepper motors for X-, Y-, Z-, and A-axes
10) Stepper motor driver power indicator light
11) On/Off switch for CNC driver power supply
12) 115/230V driver board voltage selection switch (inside computer case)*
12) USB ports, rear panel (more on front panel)

***NOTE:** *Sherline attempts to ship machines with the voltage setting preset to the proper voltage for the customer's location. Confirm this setting before turning the computer on for the first time. If the setting is not correct, note that there are two switches that need to be changed. The switch for the driver power supply is inside the computer case. Contact Sherline before opening the case to change the voltage or for any reason. Breaking the tamper-proof seal without authorization may void your warranty. Illustrated instructions on how to locate and properly change the setting on this switch can be found at www.sherline.com/resetting-voltage-sherline-computer/.*

Assembling Your System

Getting your system up and running is a simple task. It consists of assembling your mill and connecting it to the computer as follows:

Mill or Lathe

Unpack and assemble your machine according to the instructions included in the Sherline Assembly and Instruction Guide packed with the machine.

Computer connections and starting up

Unpack the computer and enclosed components. Connect the stepper motor cables leading from the back of the computer to each of the three stepper motors. A fourth cable is unused unless you have purchased an optional rotary table that can be connected here as a 4th (A) axis.

- If your keyboard and/or mouse have a USB plug, insert the USB plug into any available USB port at the back of the PC. If you have a keyboard with a purple plug, insert it into the purple connector at the back of the PC (#1 in the previous photo).
- If you have a mouse with a green plug, insert it into the green connector at the back of the PC (#2 in the previous photo).
- Plug your monitor to the blue video connector (6).
- **If not already connected, plug the free end of the 25-pin parallel cable in to the parallel port (4).**
- Insert the female end of the power cord into its receptacle (5) in the computer and plug the male end into a properly grounded wall socket or surge protected power strip.
- Make sure the computer power switch (#3) is turned on. Turn on the computer (press the power switch on the front of the PC) and monitor. (**Note:** Newer computers are powered on and start up from a single button on the front of the case.)
- Leave the power switch to the stepper motors (located on the side of the PC) in the "OFF" position for now.
- Once the desktop icons are displayed, double click the appropriate LinuxCNC icon. There is an LinuxCNC icon for "inch" and "metric" versions of the CNC Mill and Lathe, so choose the appropriate one. Use the program that corresponds with your leadscrews.
- Wait until the LinuxCNC program starts up before turning on power to the stepper motors with the toggle switch on the side of the computer case.

NOTE: If you start the LinuxCNC and run a program and the screen shows the program is running but the stepper motors are not moving, make sure both ends of the 25-pin parallel cable are plugged in as noted above. This is how the computer communicates with the driver box and is the most common technical assistance call we get. The parallel port connection is external and we connect the driver box to the computer through this connector whether the box is mounted externally or built into the computer. **EMC or LinuxCNC will not run with a USB connection**. You can also plug in a printer if need be. Now that most printers use a USB connection, this will probably not be something you need to do unless you want to use an older printer.

Opening the instructions file on the computer

Even if you are not familiar with Linux, you will find the Ubuntu desktop that opens when your computer starts up to be similar to the Windows® or Mac® desktop. These instructions for using CNC are duplicated on the "Utilities and Instructions" CD that came with your machine. There is also a shorter version that is just the workbook portion that includes the examples to learn to use g-code. Insert the CD in the CD/DVD drive and click on the icon that appears on the desktop to show the files and folders on the disk. Double click on a PDF file to open it. To open the HTML version, double click the file icon and then click "Display" in the window that opens. When done, close the instructions and right click on the DVD desktop icon and select "Eject." We suggest you print out the shorter "Print" version of the instructions to use when you are working with the examples.

Included along with the PDF and HTML versions are two MS Word (.doc) files *(CNCInst5.doc and CNCprint5.doc)*. Double clicking on a .DOC file will open it in OpenOffice on your Linux computer or in Microsoft Word if opening it on a Windows computer. The most current version of the instructions can always be found on the Internet at www.sherline.com/cnc-instructions. The PDF files can be opened in both Linux and Windows.

Please read all the instructions before attempting to use the LinuxCNC program to run your mill or lathe. CNC machining is a complicated process, and you will be directed at the proper time to run a program on the machine once you have gained the knowledge you need to do so.

LinuxCNC—Still the latest and greatest

I chose the LinuxCNC program to work with because: 1) The EMC uses coding standards that are used in the modern machine tool world, 2) You will not be dealing with an outdated programming system, 3) The EMC can deal with the cutter compensation (cutter comp) that is needed to make complex parts in an efficient manner, 4) The cost saved by not having to pay for this marvelous programming system will allow you to purchase a separate dedicated computer to drive your mill, and 5) There's great bunch of smart people working on the EMC to constantly improve the system and make it easier to use in the future and remain current.

Basic Operation of the Computer and File Navigation

Booting up

Before turning on the computer, **make sure the 115V/230V switches on the back of the computer and on the side of the stepper motor power supply inside the computer are set to the proper voltage for your local electrical current. The default setting is 115 VAC.** Also, make sure the ON/OFF switch for the stepper motor power supply on the side of the computer is in the "OFF" (down) position. Once the EMC program is fully loaded and running, power

to the stepper motors can be turned on. **Do not turn on power to the stepper motors unless the EMC program is running.** Controls within the EMC prevent overloading of the power supply when multiple motors are powered up at the same time, but without this safeguard motors or drivers can be damaged.

Opening the LinuxCNC Program—Login and Password

When starting up, the computer will boot up without asking you to log in. If you log out of the desktop and log back in and get a login screen, enter **sherline** (all lowercase letters) for the login and **sherline** (all lowercase letters) for the password. Click the [Login] button and the desktop will appear.

To open the LinuxCNC, double click on the desktop icon that says either *Sherline Mill Inch, Sherline Mill MM, Sherline Lathe Inch or Sherline Lathe MM* depending on the machine you will be using. If the icons don't appear, move your cursor to the menu bar at the top of the screen and click on "Applications." In the dropdown menu that appears, navigate to *CNC>LinuxCNC.* In the menu tree that appears, you can select the program you need for mill or lathe, inch or metric. Note that the program that opens from this menu tree is the "Axis GUI," which is slightly different from the "Mini GUI" linked from the desktop icons. The differences between the two are explained in the EMC User Manual which can also be found in the *Applications>CNC* dropdown menu.

Shutting down the computer when done

Before shutting down the computer, close all open windows and applications. Then go to the menu bar at the top of the screen and navigate to *System>Quit*. After clicking on "Quit," you have the choice to "Log Off" and leave the computer running or "Turn off Computer" which will log you off and shut down the computer. There is also a red power button icon in the far upper right corner of the screen. Clicking on this will also offer you the following choices: Log Off, Switch User, Lock Screen, Restart or Shut Down. (If you log off, you will have to use the password "sherline" to log back in.)

Emergency stops

If you see a physical "crash" is about to occur, you can use the mouse and the [Feed Hold] command to stop the program, remove the obstacle and then use [Continue] to continue with the program, but that can take too much time in a panic situation. The fastest way to stop the stepper motors is to turn the power switch on the side of the computer to the "**OFF**" position. The spindle will continue to run, and the machine will have to be re-homed and the program restarted, but turning off power to the stepper motors while they are running will not cause damage. Running a slide until it hits a hard stop should not cause any physical damage either, but power to the motor should be turned off either by stopping the program or turning the stepper motor power supply switch to OFF as soon as possible to prevent possible overheating of the stalled motor. It might save a valuable part that is about to be destroyed.

Transferring g-code files from another computer

1. On the computer you use to generate your g-code, save your g-code program to a CD, DVD or USB flash drive as a text (.txt) file. Program files created in LinuxCNC will automatically be saved with the .ngc extension. Either file format will be recognized by LinuxCNC.
2. Insert the CD, DVD or USB device into your Sherline Linux computer. When you do, an icon for the device will appear on your desktop. When you double click on the icon, a window will be opened showing the contents of that device.
3. On the desktop of your Sherline Linux computer, double click on the folder named "g-code" to open a new window.
4. Click on the file you wish to move, drag it from the CD, DVD or USB window and release the mouse button to drop it into the "g-code" window. You will be offered the choice to "Move Here" or "Copy Here." Use the "Copy Here" command. A copy of the file will now be added to the target folder and the original will remain in the source folder.

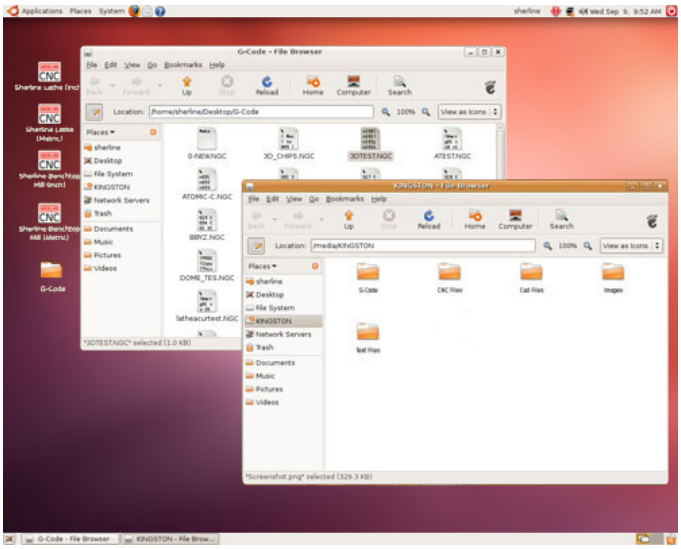


FIGURE 1.3
Files are dragged and dropped in Linux the same as they are in Windows or Mac operating systems. Above is a typical desktop with both the "g-code" folder window and the USB flash drive window open. The various LinuxCNC icons and g-code folder can be seen to the left on the desktop. In the upper left corner is the main menu bar. The desktop background image or "wallpaper" can be the default or an image of your choice.

Saving a file created or modified in LinuxCNC to your "g-code" folder

1. On the top LinuxCNC menu bar, make sure the "Editor" box is checked (filled with color).
2. If the editor window is not displayed as the main window, click on that window to cycle through the checked functions until the window showing the lines of g-code is displayed.
3. The main window will now have a small menu bar above it. *Select File>Save As*. When you do, a window will open.
4. The default directory to save programs should be the "g-code" folder. If it is not, in the file saving window

click on the file icon in the upper right corner of the window and navigate to **home/sherline/Desktop/g-code** to select the g-code folder.

5. In the "Name" bar type in a file name and then click the [Save] button.

You can also save a file directly to a CD, DVD or USB flash drive by navigating to the appropriate drive instead of the "g-code" folder (as in step 4 above) and then naming and saving as in step 5.

NOTE: In the Ubuntu version of Linux it is no longer necessary to unmount the drive when you are done. Simply close the drive's window. If using an earlier version of linux, you must "Unmount" the CD or USB device before removing it. To unmount the drive, right click on the appropriate icon on the desktop (it will have a green triangle at the bottom right corner of the icon) and select "Unmount" from the menu. Once the device is unmounted, the small green triangle will disappear, and you can safely remove the CD or USB device.

What's in the "g-code" folder?

When you open the folder called "g-code" there are already several programs we have pre-loaded that you can run or modify. Here is what they are called and what they do:

0-new.ngc—A blank program ready for you to open and rename to create your own programs

TheGreatRace.ngc—A triangular aircraft race course used as an example in the instruction manual. **Note:** This is the only example given in the Sherline instructions that you don't have to type in yourself because it is lengthy for what it does.

The following programs are written by the EMC group as examples of different types of programming techniques and haven't been fully tested by Sherline. Ray Henry provided the explanations.

3D_Chips.ngc—This little penguin is the EMC mascot. It is written using metric dimension but can be run on any mill. It is a bit too large to run directly on the Sherline mill but is a great sample of contouring that will go quite a ways in foam.

3dtest.ngc—Runs X, Y, and Z axes to draw circles in several planes (on-screen use only, not for cutting an actual part). Run this from near the center of each axis and it draws a circle and the letters of the plane on which the circle lies. This can be handy when you change the angle of view of the plotter, because it will show if you are looking through an axis because the lettering will be backwards or upside down.

bbxy.ngc— (Metric) This program moves the cutter in about a four-inch circle in one direction and then the other. There is an m00 between the direction change so you need to press [Resume] or <s> on the keyboard

bbxz.ngc— bbxz.ngc is the same as bbxy above except that it makes circles in other planes.

bbyz.ngc—bbyz.ngc is the same as bbxy and bbxz above except that it makes circles in other planes.

cds.ngc—Circle Diamond Square (cds) is the original proof that the EMC interpreter could run a milling machine. It is also a bit large for the Sherline mill but works great in the backplot mode. It allows you to change views and look at the various parts of a milled cube.

diag.ngc—Tests movement of each axis using small incremental moves. Hint—You will want to rapid to 0,0,0 before you start this program or have something else to do.

skeleton.ngc—A sample of code used to create a standard starting and ending condition for a program. This does not show much if you run it but you can copy and past it into your programs and then paste your code in the center. That way you can be certain of the state of the machine after each run.

Dome_Test.ngc—Runs X, Y and Z axes to create a slight domed surface.

isd.ngc—As well as nist.ngc and nist2.ngc are variations of a router program that cuts a NIST (National Institute of Standards and Technology) logo. There is an error near the end of the code in one of them. See if you can find it. These take a while to complete.

Opening the new file in LinuxCNC

1. Open LinuxCNC by clicking on the appropriate machine icon on the desktop (Sherline Mill Inch, etc.).
2. Click the [AUTO] tab along upper menu bar.
3. Click the [OPEN] tab on lower menu bar (the opened path should be **home/sherline/desktop/g-code**).
4 Highlight the file you want to open by clicking on it.
5. Click the [OPEN] button.

Saving a g-code file to a CD, DVD or Flash Drive

CD or DVD—The Ubuntu version of Linux includes a free CD/DVD writing utility called Bracero Disc Burning. Here is how to use it:

- Open the program by navigating the main menu to ***Applications>Sound & Video>Bracero Disc Burning***
- Select "Data Project" from the four choices offered
- Either drag the files to be copied into the open window from another window (such as the g-code folder) or use the navigation feature to select and include the files to be copied
- Insert a blank CD or DVD into the drive
- Click the [Burn] button in the lower right corner of the application window
- Close both windows, exit the program and eject the disc.

USB Flash Drive

- Insert the flash drive into an available USB port.

- Click on the flash drive icon that appears on the desktop to open a window.
- Open the g-code folder
- Drag the file(s) to be saved from the g-code folder to the flash drive window. A small menu will appear. Select “Copy Here.”
- Click the “X” in the upper right corner of the USB drive window to close it.
- Remove the drive from the USB port

Familiarizing Yourself with the Control Panel

The Sherline Graphical Interface

By Ray Henry and Joe Martin

Start your engines

After having connected everything as shown in the reference photos shown previously you will need to turn on the computer but not the motor drivers (the drivers are the individual circuitry that control the stepper motors) to see how this system runs. The power switch for the computer is located on the front of the computer. The driver power switch is located on the left side of the computer as indicated in the photo. Up is on, and down is off.

When the computer is turned on the desktop will appear. This will allow you to have access to various Linux programs, however, the only program that Sherline will be using is one of the CNC lathe or mill, metric or inch programs. Choose the program that represents the leadscrews (inch or metric) that are installed in your Sherline mill or lathe. Double click on the appropriate desktop icon and then the LinuxCNC program will open. Unlike earlier versions you will not be asked for a login or password. (**NOTE:** If you log out and log back in you will be asked for a login and password. In machines with EMC or LinuxCNC installed by Sherline the login will be sherline and the password is sherline. If you have done the installation on your own machine, the login and password will be whatever you set up in the initial installation, so do not forget what you chose.)

I had Ray Henry put together the interface specifically for the Sherline CNC. He also wrote much of “The Sherline Graphical Interface” section of the instructions. Many of the features of the standard EMC program were removed for the Sherline interface because they didn’t apply for a mill of this size and type to keep things as simple as possible. Sherline machines do not have limit or home switches and do not have direct control of spindle speeds and such. It is a general motion interface to the EMC. With it you can jog any axis. You can set coordinate systems and tools. You can verify and view the spindle path and motion in a built in program called “Backplot” and dry-run a program before running the machine. The panel is laid out in a sensible manner and is easy to understand.

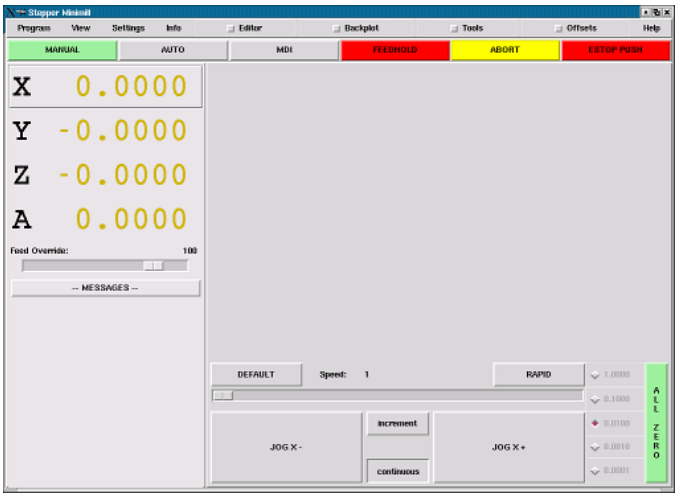


FIGURE 1.4—The control panel screen

The screen is laid out in sections. The first row is the menu bar across the top. Here you can configure the screen to display additional information. The small buttons that are located left center of the command indicate that they control the large blank section of the control panel that is sometimes called a “pop-in.” This will allow you to view the editor, plotter, tool page and coordinate page. If more than one pop-in is active (button shown as red) you can “toggle” between these pop-ins by right clicking anywhere within the pop-in screen with your mouse.

Below the menu line is a line of control buttons. These are the primary control buttons for the interface. Using these buttons you can change mode from [MANUAL] [AUTO] [MDI] (manual data input). You can also use [FEEDHOLD] or [ABORT] to stop or abort a programmed move. When you press the [FEEDHOLD] button it changes color and becomes a [CONTINUE] button. It toggles between running and pausing. Feed Hold has the advantage of being able to restart the program from where you stopped it. When all else fails press a software E-Stop—the button in the upper right labeled [ESTOP PUSH].

There are two columns below the control line. The left side of the screen shows axis position, feed rate override, and any messages that are sent by the EMC to the operator. You can add things like current tool number and length, type of position shown, and offsets in effect by looking under the <View> menu.

Directly below the pop-in area on the right side of the screen is the area that is controlled by the chosen mode. In [MANUAL] mode it’s configured so that you manually control the slide movements. You’ll have your choice of “jog” or “incremental” moves using the same big buttons. In [AUTO] mode it displays a set of buttons that allow you to load a part program, view the current location and [Run] it. In [MDI] mode it displays a line where you can enter your command in standard programming code.

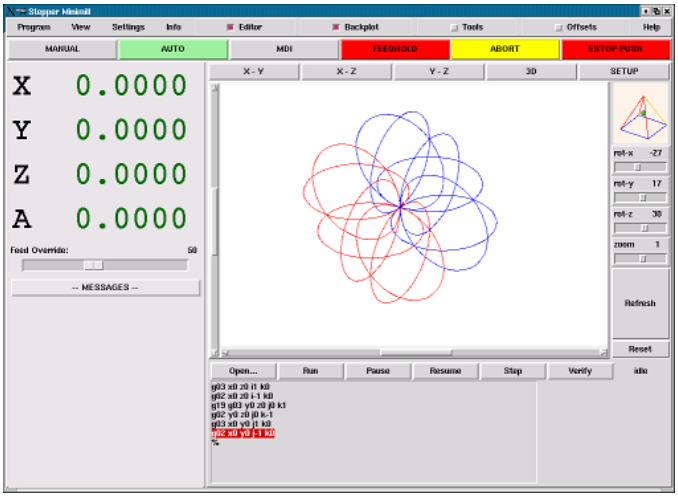


FIGURE 2.1—The screen with BACKPLOT displayed

If you compare the second screen shot with the first you can see that we are in now Auto mode, a program has been run, and the tool path is being displayed using Backplot.

Notice that the display position numbers have changed color. When you start first load the LinuxCNC program, the slide location position will be displayed in gold. This means that a homing routine has not been done yet. Once an axis has been homed, the numbers turn dark green.

To the upper right of the plotting screen is a small window with a pyramid figure that displays the angle from which you are viewing the tool path. When the LinuxCNC is first started, the default view in the Backplot mode is oriented at (X-27°, Y17°, Z30°, Zoom=0). These values can be changed by using the sliders underneath the pyramid to view your program from any angle and at various distances. While running your program in Backplot mode, keep these values in mind, as they will determine how the program is drawn on the screen. While viewing your tool path from beneath, for example, the movements may appear to be opposite of how you have defined them in your program.

Feed Hold and Feed Rate Override

You can operate feed rate override and feed hold in any mode of operation. Override will change the speed of jogs or feed rate in manual or MDI modes. A feed hold will be initiated if you click the [FEEDHOLD] button or if you press the pause button on the keyboard. Once in feed hold, the same actions will return you to the speed the machine was moving before the feed hold.

You can adjust feed rate override by grabbing the slider with your mouse and dragging it along the groove. You can also change feed rate a percent at a time by clicking in the slider's groove. In auto mode you can also set feed override in 10% increments by pressing the top row of numbers on your keyboard. The number 1 will set feed rate override to 10 percent of programmed or jog speed. The number 9 will set 90 percent.

NOTE: Do not enter a value higher than 100 for the Feed Override, as the program will stop and you will get the error message "axis 0* following error." (*The number indicated could be 0, 1, 2 or 3 depending on the axis selected.)

Messages

The message display located under the axis positions is a sort of scratch pad for the EMC. If there are problems it will report them there. If you try to home or move an axis when the [ESTOP] button is pressed, you'll get a message that says something about commanding motion when the EMC is not ready. If an axis faults out for something like falling behind, the message pad will show what happened.

If you want to remind yourself to change a tool for example, you can add a line of code to your program that will display in the message box. Example: (msg, change to tool #3 and press resume) will display "change to tool #3 and press resume" in the message box. The "msg," comma included is the command to make this happen; without "msg," the message wouldn't be displayed in the message box.

To erase messages, simply click the message button at the top of the pad or hold down [Alt] and press [m].

Manual Mode

Manual mode shows a set of buttons along the bottom of the right-hand column. A green button labeled "ALL ZERO" has been added to designate the present position as the home position. I felt that a machine of this type would be simpler to operate if it didn't use a machine home position. This button will zero out any offsets and will home all axes right where they are.

Axis Focus (Active)

Axis focus is important in manual mode. Notice that in Figure 2.2 you can see a line or "groove" around the X-axis position display. This groove says that X is the active axis. It will be the target for jog moves made with the plus and minus jog buttons. You can change axis focus by clicking on any other axis display. You can also change axis focus in manual mode if you press its name key on your keyboard. Case is not important here. [Y] or [y] will shift the focus to the Y-axis. [A] or [a] will shift the focus to the A-axis. To help you remember which axis will jog when you press the jog buttons, the active axis name is displayed on them.

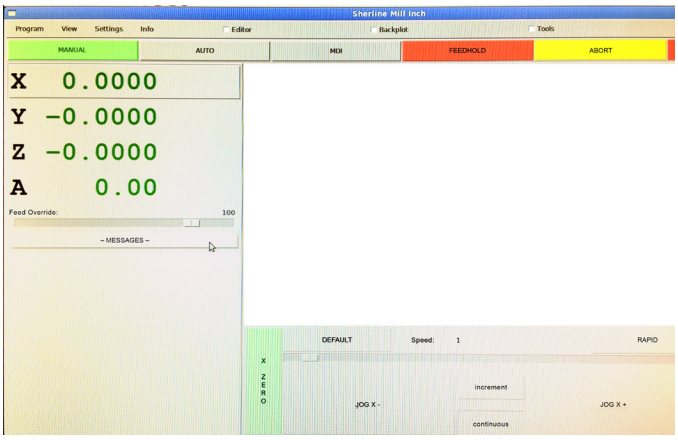


FIGURE 2.2

Jog Mode

The EMC can jog (move a particular axis) as long as you hold the button down when it is set for "continuous," or it can jog for a preset distance when it is set for "incremental." You can also jog the active axis by pressing plus [+] or minus [–] keys on the keyboard. Again, case is not important for keyboard jogs.

The two small buttons between the large jog buttons let you set which kind of jog you want. When you are in incremental mode, the distance buttons come alive. You can set a distance by pressing it with the mouse. You can toggle between distances by pressing [i] or [I] on the keyboard.

Incremental jog has an interesting and often unexpected effect. If you press the jog button while a jog is in progress, it will add the distance to the position it was at when the second jog command was issued. For example, two one-inch jog presses in close succession will not get you two inches of movement. You have to wait until the first movement is complete before jogging again.

Jog speed is displayed above the slider. It can be set using the slider, by clicking in the slider's groove on the side you want it to move toward, or by clicking on the [Default] or [Rapid] buttons. This setting only affects the jog move while in manual mode. Once a jog move is initiated, jog speed has no effect on the jog. As an example of this, say you set jog mode to incremental and the increment to 1 inch. Once you press the [Jog] button it will travel that inch at the rate it started.

Directions of Travel

A note near the end of Part 1 explains how direction of travel is called out in cnc programming. Keep in mind that the + and – directions refer to APPARENT TOOL MOVEMENT in relation to the operator, not direction of table travel.

Keyboard Jogging

When the control is in manual mode you can also control the slides using the keyboard cursor control arrows. The slides will move in the direction the arrows point, and the Z-axis can be raised or lowered using the [Page Up] or [Page Down] keys. The speed the slides move at will be determined by the manual feed rate setting.

Yes, you can move the machine around in manual mode. You can even do some tolerable milling in manual mode, so long as you work a single axis at a time and set jog speed for the feed rate that your tool, machine, and material need, but the real heart of CNC machine tool work is the auto mode.

Auto Mode

Sherline's auto mode displays the typical functions that people come to expect from the EMC. Along the top are a set of buttons which control what is happening in auto. Below them is the window that shows the part of the program currently being executed. As the program runs, the active line shows in white letters on a red background.

The first three buttons, [Open], [Run], and [Pause] do about what you'd expect. [Pause] will stop the run right where it is. The next button, [Resume], will restart motion. They are like "feed hold" if used this way. Once [Pause] is pressed and motion has stopped, [Step] will resume motion and continue it to the end of the current block. Press [Step] again to get the motion of the next block. Press [Resume] and the interpreter goes back to reading ahead and running the program. The combination of [Pause] and [Step] work a lot like single block mode on many controllers. The difference is that [Pause] does not let motion continue to the end of the current block.

Feed Rate Override and Auto Mode

The number buttons along the top of the main keyboard will change feed rate whenever you are in auto mode. Pressing the gravé [`] gets you zero percent feed rate. One through zero gets you ten through 100 percent feed rate. These keys can be very handy as you approach a first cut. Move in quickly at 100 percent, throttle back to 10% and toggle between [Feed Hold] and 10% using the keyboard's pause key. When you are satisfied that you've got it right, hit the zero to the right of nine and go.

Verify

The [Verify] button runs the interpreter through the code without initiating any motion. If verify finds a problem it will stop the read near the problem block and put up some sort of message. Most of the time you will be able to figure out the problem with your program by reading the message given and looking in the program window at the highlighted line. Some of the messages are not real helpful. Sometimes you will need to read a line or two ahead of the highlight to see the problem. Occasionally the message will refer to something well ahead of the highlight line. This often happens if you forget to end your program with a %, m2, m30 or m60.

Restart

To the right of the program window is a space that will show a set of restart functions. These pop in whenever a program is aborted or an EStop happens while the interpreter is running or paused. You can also display or hide these using the view menu.

I've got to warn you that [Restart] can be risky. [Restart] reads down through the program to the restart line. While it does this it sets up its "world model" so that it can resume right where it stopped, but only if the tool is right where it was. Most of the time, when we abort or EStop it's because something went wrong. Perhaps we broke a tool and want to change it. We switch to manual mode and raise the spindle, change tools, and assuming that we got the length the same, get ready to go on. If we return the tool to the same place where the abort was issued, the EMC will work perfectly.

It is possible to move the restart line back or ahead of where the abort happened. If you press the [Back] or [Ahead] buttons you will see a blue highlight that shows the relationship between the abort line and the one on which the EMC stopped. By thinking through what is happening at the time of the restart you can place the tool tip where it will resume work in an acceptable manner. You will need to think through things like tool offsets, barriers to motion along a diagonal line, and such before you press the [Restart] button. A restart is a good time to use feed rate override and the pause key on your keyboard.

MDI

MDI mode allows you to enter single blocks and have the interpreter execute them as if they were part of a program—kind of like a one line program. You can execute circles, arcs, lines and such. You can even test sets of program lines by entering one block, waiting for that motion to end, and then enter the next block.

Below the entry window, there is a listing of all of the current modal codes. You can also write this list to the

message box in any mode if you look under the info menu and select active g-codes. This listing can be very handy. I often forget to enter a g00 before I command a motion. If nothing happens I look down there to see if g80 is in effect. G80 stops any motion. If it's there I remember to issue a block like g00 x0 y0 z0. In MDI you are entering text from the keyboard so none of the main keys work for commands to the running machine. [F1] will Estop the control.

Pop-in Editor

To edit a program that is already open, make sure the "Editor" button is active at the top of the screen. (Clicking on the box next to the word "Editor" will make it turn red, meaning it is active.) Clicking repeatedly on the work screen toggles between any functions that are active (box is red) across the top row. In the Editor window you can edit the g-code for your file. When you open this screen, you will see that a small menu bar opens above the screen with the usual editing functions to be found under "File," "Edit," etc. With these you can copy and paste within the program. The pop-in editor does not allow you to copy and paste between programs.

CAUTION! Be very careful when you highlight text when you are writing a program. **If you accidentally hit any key while a line or lines is selected it will delete the selection**. Ctrl/z and Ctrl/u (press and hold the Ctrl key) will "undo" the last command. Depending how the line(s) were deleted, you may have to press Ctrl/z or Ctrl/u twice. It is NOT a fail-proof procedure, so do not rely on it. Using the *File>Save* command often is highly recommended to save your work periodically. There is nothing worse than typing many lines of code, only to lose them in an instant.

The Editor feature is included because it has the ability to number and renumber lines in the way that the interpreter expects of a file. It will also allow you to cut and paste from one part of a file to another. You can save your changes and submit them to the EMC interpreter with the same menu click. You can work on a file in here for a while and then save and load if mini is in auto mode. If you have been running a file and find that you need to edit it, that file will be placed in the editor when you click on the editor button on the top menu.

Using the Linux text editor

The Linux (Ubuntu) operating system includes a text editor named "gedit." You can find gedit by going to Applications>Accessories>Text Editor. When you start it and select "Open" you can find all your stored files and use *File>Open* to open one. (Double clicking the file name will also open it.) If you highlight a section of text and use the *Edit>Copy* commands you can then open another program and use the *Edit>Paste* commands from the menu to paste the lines of code into the new program. This editor does not have the ability to automatically reload the program into the EMC interpreter after a change. You will need to do that in LinuxCNC by selecting the file after saving it.

Backplot

[Backplot] will show the tool path that can be viewed from a chosen direction. "3-D" is the default. Other choices and controls are displayed along the top and right side of the pop-in. If you are in the middle of a cut when you press one of these control buttons the machine will pause long enough to re-compute the view.

Along the right side of Backplot is the pop-in that you can display with the [SETUP] button. This will show a small graphic that tries to show the angle you are viewing the tool path from. Below it are a series of sliders that allow you to change the angle of view and the size of the plot. You can rotate the little position angle display with these. They take effect when you press the [Refresh] button. The [Reset] button removes all of the paths from the display and readies it for a new run of the program but retains your settings for that session.

Pop-in Tool

The tool page is pretty much like the others. You can set values here and they become effective when you press the [Enter] key*. You can't change tool offsets while the program is running or when the program is paused. The [Add Tools] and [Remove Tools] buttons work on the bottom of the tool list. Once a new tool has been added, you can use it in a program with the usual commands.

*Always use the [Enter/Return] key at the right side of the normal alpha keyboard, not the [Enter] key at the bottom right of the numeric keypad.

Editing the Tools Table—Default numbers must be deleted and replaced with real tool info before proceeding

If you are using the LinuxCNC and using the Sherline Base Page that comes with our disc, then the tooling page is different than the LinuxCNC tooling page. On the Sherline Base Page, when you first install it and click on the Tooling page you will see what I show below. All of the tooling information that is shown on the original page is Sample Info and it doesn't work.

Very Important: If you just edit the Sample info that is already on the tooling page and put in the tool info for the tools that you are using, it will not work! You need to get rid of all of the sample information and start fresh.

Default Tooling Page when first installed:

Tool #	Length	Dia.	Comment
T2	D0.06250	Z+0.100000	{;1/16} endmill
T3	D0.20100	Z+1.27300	{??????}
T99999	Z+0.100000	;BIG	
ADD EXTRA TOOL	REMOVE LAST TOOL		

In order to input any information into the tooling page that will work, you need to remove all of the default tooling information.

To do this you click on "**REMOVE LAST TOOL"** three times. This will get rid of the tooling info for the three tools shown above.

Then you click on "**ADD EXTRA TOOL**", and input your real tool information.

Your new tooling page information should look something like this:

Tool #	Length	Dia.	Comment
#1	-.2	.1875	3/16² Spot Drill
#2	-1.0	.125	1/8² Drill

I don't know where they came up with the default information, but it is for show only. It is a poor example of what should be on the tooling page, and it doesn't work. Again, you need to remove all of it and start fresh.

Limit switches

I felt that limit switches would add an unnecessary cost to the system when using stepper motors with limited torque such as these. They have to be mounted in areas where they are exposed to chips, oil and coolant creating more problems than they eliminate. The Sherline mill is fitted with "hard stop" screws on the X and Y axes that limit travel. These stops are the heads of cap screws fitted to the table and base and positioned to stop a runaway axis without damage. If you make a programming error that would drive the table into a crash situation by over-traveling, the table will encounter these hard stops before damaging the leadscrews and nuts.

You could also locate a home position from these hard stops by stalling the stepper motors against them. Of course you should do this in manual mode and by bringing the table against the hard stop carefully in slow speed and "buzz" the motor against it. I personally would find a different way of locating a home position, using the dials on the handwheels for example, but it is an option for unusual situations

It should also be noted that the hard stop screw would have to be removed from the left underside of the table in order to have the full 9 inches of advertised X-axis travel. With the stop screw in place travel is about 8.65". Note also that Sherline now offers an optional 18" mill table (P/N 54182) with about 13.65" to 14" of X-axis travel and an optional 15" mill column (P/N 45260) that adds 4 more inches to Z-axis travel.

Programming and Operating Your Sherline CNC Vertical Mill

By Joe Martin

Manual vs. CNC

Let's think about what we have to do to write a program for CNC:

1) Create a spot on your storage system.
2) Write a program.
3) Test it for errors using the Backplot program.
4) Dry run the program with the spindle well out of the way so it can't possibly crash.
5) Accurately align the machine with the work so they work in unison when the program is run.

Push the button and have a cup of coffee while your little Sherline just works its butt off for you making good parts.

When you think about this, it has the same steps as to do it manually. Compare:

1) Make some special tooling to hold the special shape. With CNC you can hold raw stock in a vise and machine the shape you need.
2) We'd have to store the special tools in case we wanted to make another one.
3) You'll make a lot of scrap with a manual machine if you don't double check your work before making the first cut, and you will not have a computer to help you as you do with EMC.
4) No matter how you do it, you always have to line up the work to the machine.
5) Don't put the coffee on yet because your real work has just begun while our boy with the CNC is pouring a second cup and making 7th part of 10 pieces. (Oh, how I wish it could be this easy at the start!)

Using "Backplot" to check your program

1) To work with any existing programs we have to be in the [AUTO] mode. With power to the stepper motors turned off [Open] the existing program called "3dtest.ngc." This file comes with EMC.
2) Open the [BACKPLOT] program that can be found on the upper line and a blank section will appear in the Pop-in section. The [Backplot] feature will allow you to visually check a program without actually running the stepper motors. Right clicking your mouse controls the program that is viewed in this section of the control panel. You can toggle between the editor and the backplot easily. We now have the perfect setup to learn CNC and it all fits on a single desktop. You can view and edit your program or view the moves the spindle will make while your program is running.
3) Now that we have a program loaded and the Backplot program running, let's take a break and test our setup. Make sure that [ESTOP] is active with the button highlighted and execute [Run].
4) Observe that the programmed moves can be viewed in the Backplot box and the viewing area can be controlled with a variety of controls. This is very useful in checking out the programs you'll be writing in the upcoming lessons. The line of the program that is highlighted in red in the lower area of the control panel is the line of code that is being processed by the control as the computer runs the program. Note that the EMC program is processing more than a single line of code at the same time; therefore, this line should only be considered "approximate."

How to make a file for our programs

It's possible to create a new file and have it end up in the wrong folder never to be seen again. If you use this method, you will always have your new programs in the proper folder. If you are using a computer supplied by Sherline, you'll find the file called "0-new." The instructions in sections D) and E) below describe how to change "3dtest.ngc" to "0-new." Since Sherline owners will have this file already

provided for them, they can skip sections D) and E) when they come to them and go on to section

1) To create a new file or work with any existing programs remember you have to be in the [AUTO] mode. Notice that a new line of commands appears along the bottom in this mode. The [Open] button is located in the along this row. **Note:** the method I'm describing in A) to E) below is to set up your computer for future use and isn't the necessary steps you'll use in the future.

   A) With power to the stepper motors turned off [Open] the existing program called *3dtest.ngc.**

      *This file comes with EMC. If you are using a computer supplied by Sherline, you should instead open the file called *0-new*. The instructions in sections D) and E) below describe how to change *3dtest.ngc* to *0-new*. Since Sherline owners will have this file already provided for them, they can skip sections D) and E) when they come to them and go on to section 2).

   B) Go to the Pop-in [Editor] command and open along the top row.

   C) The window will display the entire program.

   D) For those of you who do not have a Sherline-supplied computer we are now going to create a file we will call "0-new." (If you have a Sherline computer with the 0-new file already on it, open it and go to section 2) below.) First go back to the [Editor] window by right-clicking your mouse. By having a "0" as the first character of the file name it will always keep this file at the beginning of the program list. This makes it easy to find, because you are going to use it again and again. From the [Edit] window, use [File] then [Save as] (located along the top of the edit window) to save our existing program as *0-new*.

   E) Next, highlight (with the mouse button held down drag the cursor across the entire area you wish to highlight) and delete (press enter) the existing program with your mouse. Replace that program by typing "Save as another file name." This will remind you to only use the *0-new* program to set yourself up for creating a new program. Use the *File>Save* command. Remember, to make any changes permanent you must save the changes before closing the program.

2) We'll now use the [Save As] command to save the *0-new* file with the new name *test-g02*. By using the [Save As] command you also leave the *0-new* program as is to use again. You can create your own files and name these files as you wish from this point on. **Note:** test-g02 is the first program you'll write. Program files are saved in the directory called ***home/sherline/Desktop/g-code***. In the EMC, the programs will be saved here by default.

   NOTE: The [Places] tab of the top menu bar helps you navigate your computer to find the files you are looking for. There is also a search tool in the Accessories menu if you know the file name.

3) You have to get the new program loaded into the control (machine) in order to use it.

4) To accomplish this first use [File] [Open] to open your new "test-g02" program. The program will now be loaded in the control panel with the program name displayed.

All of this BS isn't as bad as it seems, and you will quickly create programs using this information along with "0-new."

Maintaining your EMC programs directory

Many times you will create a number of g-code files that are similar or for testing use. After a while these files can accumulate into a large repository that makes it difficult to find the one you are looking for. If you wish to delete a file, the easiest way to do so is to click the "g-code" folder on the desktop. This is the default location of g-code files created in LinuxCNC. Find the file(s) you wish to delete. Right click on them and then left click on the [Delete] option. This will permanently delete the file selected. Take great caution when deleting files! You do not want to accidentally delete a valued file. It pays to keep a backup of your g-code files on a CD or USB flash drive.

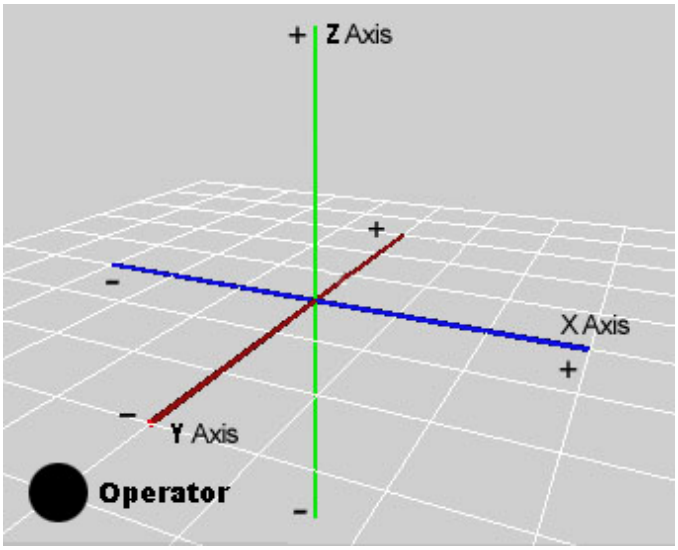


FIGURE 2.3—Mill axis directions in relation to the operator.

Directions of axis movement on a mill

Before we go on, let's be sure we understand the directions of movement of the three axes of a milling machine. When programming g-code you will use the **Cartesian coordinate system**. In relation to the machine operator, the X-axis moves left/right, the Y-axis moves in/out and the Z-axis (spindle) moves up/down.

A tip regarding axis direction

It can be a little confusing for a new machine tool operator, because when you write code you think in terms of the part being fixed and the tool moving, but when you operate most machine tools the tool is fixed and the table moves. When using the jog controls, at first it may seem like the axes are moving in the wrong direction, but when you picture the move in relation to the cutting tool it makes sense. Programming directions relate to APPARENT TOOL MOVEMENT. Therefore, a positive move on the X-axis will result in the table moving to the LEFT (tool appears to move to the RIGHT); a positive move on the Y-axis will result in the table moving TOWARD you and a positive move on the Z-axis will result in the headstock moving UPWARD. (A positive move on the A-axis will cause the rotary table to rotate CLOCKWISE.) This is the way full-size CNC machines work too.

What is the A-axis?

The 4th axis available to you in LinuxCNC is the A-axis. This axis drives a rotary table and is programmed in degrees rather than in linear dimensions. Sherline offers an optional 4th axis rotary table as P/N 8730. It already has a stepper motor mounted and is ready to plug in to the A-axis cable coming from the back of your driver box. A 4th axis can make some operations like milling a large round part or drilling circular hole patterns easier to program. It also allows you to do operations like milling threads or helical gears that would not be possible without CNC because a rotary axis and a linear axis must move at the same time and in the proper relationship to each other to create the desired tool path.

Part 2

Programming Sherline CNC Machines with LinuxCNC

Although there is quite a bit of well-written instruction on how to program on the Linux CNC site, I believe you should only go there to learn beyond the limits of my instructions. Having too many instructors on a subject that you know nothing about isn't a good idea at this time. At the end of my instructions there are lists of the many words and commands available to you taken from the Linux/EMC site, however, there's no reason to remember them now. I want you to learn them as you use them. The commands and words that I'll be using are so basic to writing code that they'll soon become a new language.

UPDATE —July, 2005: **Note:** I recently went back to the EMC website at http://www.linuxcnc.org/ and was very pleased to find many improvements have been made since I first wrote these instructions. It's much better organized now, and I believe after you get the basic concepts from my instructions about how CNC g-codes work, it would be to your advantage to go there and learn about the addition of the many canned cycles that are now available to EMC users. A complete list of g-codes is included in the Ubuntu version of Linux (2009) and can be found under *Applications>CNC* in the main menu or at www.sherline.com/g-code . I take my hat off to the EMC group for making these much needed improvements.

Print a copy of these instructions

Although you could jump between the control screen on your monitor and the instructions, it would be a cumbersome way to work. You can print a copy of these instructions from the CD using any computer with a printer hooked up to it, or you can print a copy from your Linux computer by unplugging the servo driver connector from the parallel port and plugging in a printer instead. In fact, most printers can now be plugged into the USB port in the front of the computer.

To make it easier and to save paper, we have saved just the portion of the instructions that refers to the programming as a separate file you can print out. You have your choice of formats: They are called CNCprint.doc (MS Word) or CNCprint.pdf (Adobe Acrobat Reader). You can also open the instructions on Sherline's website at www.sherline.com/cnc-instructions and print only the pages starting at the beginning of Part 2. (**Note:** Later editions of these files may include a version number in the name; for example, CNCprint5.pdf.)

Your Linux computer already has a program installed that will allow you to read .pdf files by clicking on them. To view .pdf files on your Windows® computer, a copy of the free program Acrobat Reader will need to be installed if it isn't already. Clicking on a .pdf file should cause that program to open automatically. If you don't have it or want to get the latest version, go to www.adobe.com/products/acrobat/readstep.html.

Inch vs. metric dimensions in g-code

Because my instructions have been written using the inch dimension system, work with the inch version of LinuxCNC when using my examples at this time. This will slightly decrease actual table movements on metric machines. The **g20** and **g21** conversion command will only work if you load the LinuxCNC version that is appropriate with your leadscrews. Should you wish to use metric dimensions to program your machine, even though it has an inch leadscrew, you can do so by entering the code **g21** in place of the **g20** code when writing the program for your part. **G20** indicates inch increments, while entering **g21** indicates you will be using metric increments. The program will automatically make the translation for you. You can also use inch dimensions on a machine with metric leadscrews by using the **g20** code instead of **g21** once the proper version of the LinuxCNC has been opened.

If you use the **g20** or **g21** command to change measurement systems, don't forget to clear any previously entered numbers from the tools table. Then re-enter the tool size in the appropriate measurement system for your current program. If you don't, you will end up with a program that has tool offsets entered in a different measurement system than the **g20** or **g21** command calls for. This can give you an error message and the program may not run.

Let the BS end and the programming begin

From this point on, you have to learn programming at your CNC system. You can't learn how to control these systems in your favorite easy chair; therefore, you should take a printed copy of "Part Two" to your CNC system and enter,

test and understand each sample program before going on to the next. Write a few similar programs, with each being different enough to make it challenging and constantly test yourself as you go. No shortcuts!

Are you ready?

About now I'd be "chomping at the bit" to see something happen, so lets do something the best manual machinist couldn't do to start. Hell, anyone can cut a straight a straight line so let's make a circle. With the stepper motor controller power switched off, manually crank the handles and approximately center the X-, Y- and Z-axes. Leave the power off to the stepper motors until we get the program all checked out with Ray Henry's backplot command. Ray Henry has been a spark plug for the EMC group for some time now and we all owe him a debt of gratitude.

g-code programming

These are the basic g-codes that you'll be working with. There is a complete listing of them at the end of this section copied from the www.linuxcnc.org site. There isn't any reason for you to remember them all at this time, because I want you to thoroughly learn them as you use them, however, you can get a general idea of how the system works by reading them now. They are quite simple. Remember that modal g-codes remain in effect until the control receives a new g-code that over-rides the present code. Modal g-codes control function, direction or speed and, in general, cannot be used in the same line of code with other modal g-codes for obvious reasons.

g00 rapid positioning

g01 linear interpolation

g02 circular/helical interpolation (clockwise)

g03 circular/helical interpolation (c-clockwise)

g04 dwell

g10 coordinate system origin setting

g17 xy plane selection

g18 xz plane selection

g19 yz plane selection

g20 inch system selection

g21 millimeter system selection

g40 cancel cutter diameter compensation

g41 start cutter diameter compensation left

g42 start cutter diameter compensation right

g43 start tool length compensation

g49 cancel tool length compensation

g90 absolute distance mode

g91 incremental distance mode

Notes on feed rates and stepper motors

We added this section to the original instructions because it covers an area of programming for our size machine that was lacking. Because our machine is a scaled down version of a full size machine, the feed rates and depth of cuts need to be scaled down too. The following section will explain feed rates, stepper motors, stalling and missed steps, and some insight for programming feeds. The maximum feed rate that you can use on a Sherline machine is 32 in/min. Looking at the program shown below (See Figure 1), it has feed rates showing 60 in/min and 120 in/min. When you put in feed rates that exceed the maximum feed rate of the machine, the machine will default to the highest feed rate possible. This means that regardless of what your programmed feed is, the machine is feeding at 32 in/min throughout your entire program.

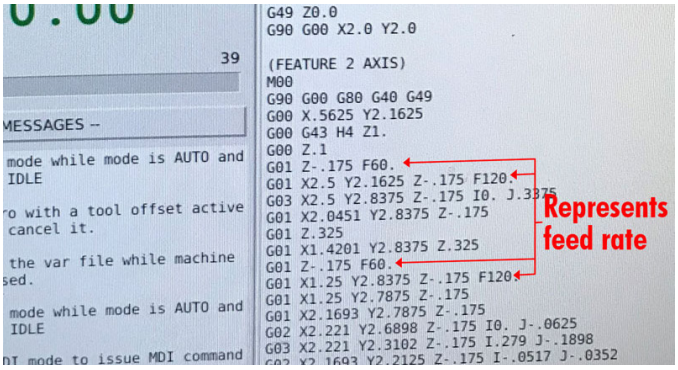


Figure 1

Next, the way stepper motors work is that they have the most torque at the slowest speeds (feed rates). As you increase the feed rate, the torque drops down. The maximum feed rate (32 in/min) is for rapid moves only (these are positioning moves), and the lower feed rates are for cutting.

What will happen if you try to cut your parts using higher feed rates (or the rapid feed of 32 in/min), is your stepper motor will stall or miss steps while it is cutting. The reason why this is happening is because you are trying to cut material when your stepper motor has the least amount of torque available. Because these are stepper motors, they do not send a position signal back to the computer to verify that they actually moved to the position that was sent by the computer. Your computer tells an axis to move a specific distance, and then assumes that your stepper motor made it to that location when it stops. If your stepper motor stalls for a split second due to excessive feeds and it misses a step or two, your computer doesn't know that. If you have DRO on your machine, the DRO will show exactly how far the axis moved, so when your stepper motor stalls during a cut, the DRO is going to show a lower amount than the computer. (Notice the difference in the XYZ positions between Figures 2 and 3 below.)

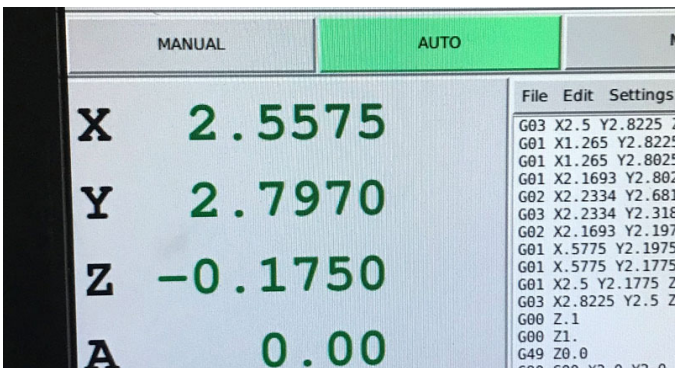


Figure 2

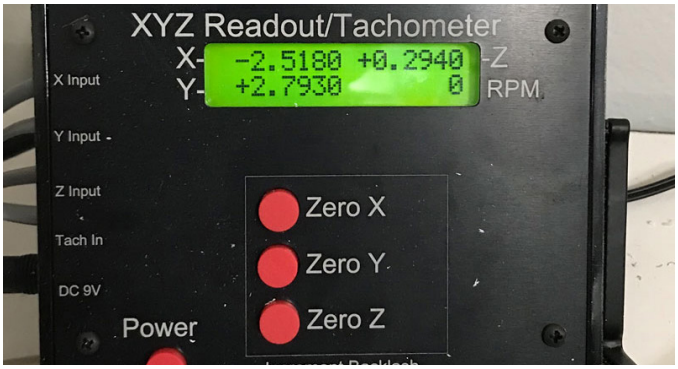


Figure 3

If your machine is missing steps, you need to slow down your feed rates (and/or take lighter cuts). Depending on your cutter size and your depth of cut, you may need to be as low as 4 or 5 in/min, or as high as 10 in/min.

As a standard practice you should always write down the numbers on your handwheel when the machine is at its home position (X__,Y__,Z__). When the program ends, your machine should be back at the home position, and the numbers on your handwheels should be correct. If the numbers are different, then your motors are stalling and missing steps. To check this, run another part and watch the stepper motors while it runs. You should notice a slight jerking motion of the handwheels on the stepper motors when they are stalling due to excessive feed rates, and your machine will not return to its original home position because of those missed steps.

This is a small machine and it can't take the same cuts that you can take on a full size mill. The rule-of-thumb for slot cuts, as an example, is that your depth of cut when cutting a full width slot should be 1/2 of the cutter diameter. On our machines it should be 1/4 to 1/3 of the cutter diameter. If you are using a 1/4" end mill, your depth of cut would be between .060 and .080. The same holds true for your step over amount. If you are milling off an entire top surface, each pass should be 1/3 to 1/2 of the cutter diameter (1/3 is better). It is better to use multiple smaller cuts.

If you see the stepper motors struggling, or if they are missing steps, lower the feed rates (and or depth of cut). You will also need to watch your cutters. As your cutters get dull, there is more force on your stepper motors to make the same cut as they were making without issue when the cutters were sharp.

Don't get discouraged. You are learning an entire trade from scratch. The more you work with the machine the better your understanding and knowledge will be.

There are no erasers in the machine trades

One other quick thing to note, it should be obvious that I could have provided a disk with all the examples typed in or to be downloaded, but accurately typing in a program is as much a part of learning CNC as making chips. When you misspell a word in a letter, the reader of the letter usually knows what you were trying to say and will not even notice it. In the world of robots, our robots can self-destruct trying to carry out your foolish typing error. This kind of typing requires them same precision as machining. There are no erasers in the machine trades. Only people who express themselves with their mouths and writings have the use of this marvelous tool. Machinists express themselves with their work, and that's the way I like it.

In this exercise the home position is going to be where the tables were located when the LinuxCNC was turned on, which is why I had you center the machine slides and is called x0,y0 and z0. The X-axis is the table that will move to your left when it moves in a positive direction. The x0, y0 and z0 is also designated in the backplot box with arrows pointing towards the positive direction away from this position. No more excuses, we're going to program that circle I promised you. **Note:** Although the LinuxCNC doesn't need the "0" after the "g" in g00 or g01 I'll use them because they are the designations all popular CNC systems use. Also, I prefer lower case letters because you are less likely to make errors and mix them with numbers. The capital letter O and the number 0 can be difficult with some fonts and are located close together on any keyboard. You are not writing a letter when you program, and it has to be accurate.

Circle program

Note: In these instructions, the code you should retype into your program will always shown in **bold face red** type.

The entire program:

```
%
(g02 circle program)
g01 g20 g40 g49 g90 x0 y0 z0 f2
g02 x0 y0 i-.5 j0
g01 g90 x0 y0 z0 f2
m2
%
```

1) **%** –Always start a new program with a % sign. This informs our robot that a program is coming in. Each time you press [Enter], the program will start a new line, so [ENTER] it is, and we're off and running.

2) **(g02 circle program)** –Words in parentheses are not part of program and are used to help the operators quickly understand the program. You can enter notes into a program at any point to remind yourself or another operator in the future why the upcoming code was entered.

3) **Note:** The g20, g40, g49 commands are not totally necessary for this program to run. They are added to protect your program from being polluted by commands that may have been active in previous programs and never canceled out. About the only one of this group that could eliminate a potential problem is the g20 which could keep your inch program from running in metric if a g21 was left active. I really didn't want to get into this subject this early in the instructions, however, if you were using a computer that other people use it could be a problem so I believed I should mention it and I'll only do so in this first programming example.

4) **g01 g20 g40 g49 g90 x0 y0 z0 f2** –What we just ordered our robot to do is go to the home position if it's not already there.

A) **g01**–Enters a mode where the feed (speed) is controlled by "f" which can be entered. Also note that the slow feed of f2 was entered so you could fully appreciate just what the machine slides do to generate a circle. The feed could be set for a faster rate for the backplot test and slowed down for the machine test if desired.)

B) **g20**–Orders the machine to use the inch measurement system. Metric machine owners will use a **g21** in place of the g20.

C) **g40**–-Cancels any unwanted tool diameter compensation that may have been left active from previous run programs.

D) **g49**–Cancels any unwanted tool length compensation that may have been left active from previous run programs

E) **g90**–Orders the machine to use the absolute coordinate system where the position will always be referenced from a zero point. Again, you can zero the axis by going to [Pop In] [Offset].

F) **x0, y0, z0** is self evident. If the slides weren't in the 0 position they would have moved to it.

This is the basic information that all programs should start with. If I had entered a g00 command to start with (orders the machine to move to the programmed position at fastest speed) it would be a quicker but one that can be more risky to start with. Modern CNC machines can perform g00 moves at speeds of 1400 inches a minute (35 mpm), and you're making a mistake by not taking advantage of this fact, but we're not pros at this time and I want you to learn about feeds. Hit the [ENTER] key.

5) **g02 x0 y0 z0 i-.5 j0** – This line is the code that is going to actually produce the x y movements to create that circle we have been talking about. In this case, all the previous commands are still in effect except the g02 is replacing the g01.

A) **g02** is ordering our robot to cut a circle or start an arc with a clockwise (CW) move. For our faithful robot to carry out this order it needs more information or it's going to start complaining and sending you nasty little notes and refuse to work. Remember that the g02 will remain in effect until a g01 is called for.

B) The g02 the program needs the **x, y** position entered as to where this arc will end, and since we are creating a circle it obviously will end where it started so the present position is entered.

C) Here are a couple of new commands; **i and j**, and they are used to describe the center location of our circle. These dimensions are given as the actual distance the center is located from the present position. It is very important to understand that our present position is not the center of this circle and only a starting point for this circle to be created. The **i** represents where the center of the circle falls on the x-axis and the **j** represents the y-axis.

I would also like to add at this time that the distances calculated and entered for computer generated shapes have to be extremely accurate, and a 0.0002″ error could stop your program from running. This doesn't have anything to do with the accuracy you need; it pertains to the accuracy the computer needs to calculate the many points it creates to generate the x y movements for this one circle. This shouldn't be any problem because of the low cost calculators we all have today that figure things to many decimal places. Hit [ENTER]. (This is the last time for the [ENTER] command to be instructed. From now on you will hit [Enter] at the end of each line of code.)

6) **g01 g90 x0 y0 z0 f2**–I always add a line of code with a g01 or g00 and a x,y,z to return to the home position. The "g" commands stay in effect until they are overwritten by another g command. In this case it would have left the control in a circle command if we didn't add this line of code which wouldn't be a good practice.

7) **m2**–Informs the computer the computer the program has come to an end.

8) **%** –The standard format of EMC or LinuxCNC is to start and the entire program with a percentage sign.

9) Now we have to load this program for your machine to run. [File] [Save and Load] In this case we are not ready for prime time, and we are going to check our program with [Pop In] [Backplot].

10) One more check –the program should look like this:

```
%
(circle g02)
g01 g90 x0 y0 z0 f2
g02 x0 y0 i-.500 j0
g01 g90 x0 y0 z0 f2
m2
%
```

11) **Go for it**, click on [Run]. The backplot program should start tracing the path the spindle will travel in relation to the work. If you got one of those little messages and it didn't run, check for typing errors and correct them. Right-click [Editor] [file] then [Save and Load] [Run] until it works.

12) The time has finally come to watch this little beauty in action, so turn on the machine control and **"let her rip."** Notice how the slides are constantly changing speeds as each axis reaches its apex and changes direction. The circle is also being generated in the CW direction as the g02 requires. Run this little program several times and take the time to really understand what's going on. From the control panel you can single step the program through and run things until you're bored.

You had your fun and it's time to go to school again

Now guys and girls, it's time to go back as many times as it is necessary until you are absolutely positive you know exactly what every letter of your program does. I'm not

looking over your shoulder, and the only way you can act foolish is by being on page 10 when you should be on page 3.

What a difference a minus sign can make

The first thing I want you to learn is how important that minus sign can be. (Also, a misplaced decimal point can easily cause a catastrophic machine event.) First, turn off the machine because there isn't any sense in having it run while we're doing this exercise. It'd also be distracting. Run your existing program through once again and generate the circle you started with. Leave the circle there and don't [Reset] until we are done with this entire exercise. Right-click to Edit your program and the only thing I want you to do is remove the minus sign from the i value in the g02 command. [Save and Load] and [Run]. I'll bet you didn't think it was going to do that! Just think what could have happened to that expensive fixture if you were controlling a 30 HP mill. A quick way to the unemployment line. I'm now going to give you the golden rule #1 of any CNC shop.

Don't ever press the green [Start] button on a CNC machine unless you know exactly what the machine will do BEFORE you

PRESS IT!

Now let's take a closer look at that minus change we made and we'll see what happened is really obvious. By changing the "i" to a positive number the center of the radius is now moved to the positive side of our zero starting point, which in this case is a total distance of 1". The g02 command still went in a CW direction and the only difference is that the circle was generated on the positive side of the X plane. Are you up for your first test? You better be because I didn't write all this stuff for my own enjoyment.

From what I explained to this point, you should be able generate a 4-leaf clover on the backplot screen. Don't forget to use save and reload as you make these changes. You're on your own.

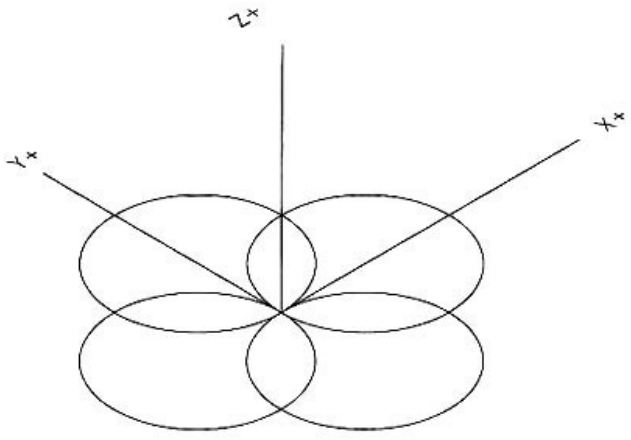


Figure 4

If you're reading this, I'll assume you passed the test, so now take your second test by erasing your program with the backspace key and doing the entire exercise over from memory.

Meet Millie

Think about what you accomplished at this point. You're beginning to understand how we humans can order our robot around. I think we should give our little robot a name. Seeing it's a mill we're working with and I'm a male, I think I'll call mine Millie. It will be the only thing with a female name in my lifetime that I stand a chance of controlling. I really want you to know the stuff I taught you blindfolded because we are heading for the hard, oops, I mean interesting part.

I'm buying

I'm a little bored with this exercise myself so let's drop over to route g41 and see what's going on. The programmers that reside on g41 are a little on the snobby side because they think they have some special talent that us regular guys can never learn. That was probably true when they learned it, but they didn't have Joe Martin teaching them. In Joe's class no one is left behind, and let me promise you this, guys, we are going move into their neighborhood so fast they will think we always lived there. But for the hell of it, let's stop by their fancy cocktail lounge and have a beer so they can get used to us. I'm buying.

Homing your machine

Now would be a good time to slip something in about "home position," which is the actual x, y and z position the machine is in when you start to run your program. I will usually have the zero points in the same location as they are on the drawing, which makes the program easier to write without errors, however, it is seldom the actual position you'd want your machine to be in to load or unload parts or check dimensions. The best way I've found to determine the home position is to first locate the zero position. Jog the slides to the approximate zero position and then move them manually with the handwheels (power to the drivers turned off) to the exact position where your cut will start. Zero the control's readout by using the "zero all axes" button. With driver power switched back on, the slides can then be moved to a position that you believe would be a good home position using the incremental jog feature or the MDI program. It's a good idea to make these whole numbers such as x2 (or 50 mm), y2 and z2 because you'll have to type these in when writing your program. You'll have to be careful of having tools loaded in the spindle and manually calling a zero position before the program has a tool length offset loaded. This last warning will make much more sense later on in the instructions.

Tool height offsets

Before we get too involved in the more complex cutter diameter programming, I want you to learn how to deal with the difference in tool lengths. It is obviously simpler to have a single zero point for all tools, and that is what the LinuxCNC allows you to do. The length and diameter of each individual tool is stored in a file called "Tools," located along the top row of the control panel. The tool length is usually called up in the first z-axis move using each individual tool by entering a g43 h1, 2, 3,--. The g43 tells the computer to use the tool length (h) stored in the line (1, 2, 3,---) of the tool data file. A g49 is entered when the z-axis is brought back to its home position to cancel out the tool length compensation that was called up when you first started using the tool.

From a practical standpoint, I like to make the shoulder located behind the ¾-16 spindle nose thread the zero point

on the Sherline mill and use screw-on Sherline tool holders as an inexpensive, quick and accurate way to change tools. If just one tool holder is used, the cutter height must be adjusted each time it is changed. If multiple holders are used, each with its own tool pre-adjusted to the correct height, tools can be interchanged quickly with excellent repeatability on their height setting. There is also a holder with a thread to accept a Jacobs 1/4" or 3/8" drill chuck, so they can also be interchanged like end mill holders. It is P/N 3074.

This allows you to measure the tool length directly with calipers by measuring the overall length of the holder with the tool mounted. Also remember that the home stopping position for the z-axis shouldn't be zero. It should be a positive number that allows you to change the tool if necessary. The x0, y0 and z0 position should be chosen carefully and be related to the part that is being programmed, not to a convenient place to change tools.

Write a couple of lines of code with and without the tool length compensation to test what you have just learned. Run it using the backplot to fully understand it before going on, and, above all, always remember to cancel the tool length compensation after using it.

Time to get serious

Let's think about just what g40, g41 and g42 does. To begin with g41 (g42 is the opposite of g41) will locate the spindle to the left of the programmed edge that you want to machine according to the d (tool dia.) amount entered into the tool data table. G40 simply cancels the g41 or g42 commands.

Computers can keep you from ruining your part

These commands allow you to put the actual dimensions of the part you want directly into the program and then it will take the diameter that you entered into a [Tools] file and automatically correct the path of the cutter so that the edge of the cutter is always located on the edge on the part. Your computer will be making thousands of calculations a second to do this, and old Millie will just bust her hump for you to make this happen if you follow their rules. If you don't, they'll start sending you those nasty little messages again, but look at it as a blessing. They'll keep you from wrecking your part.

Another thing to keep in mind is that one of the great benefits of the tool offset commands is being able to use end mills with slightly different diameters without having to calculate all new tangent points. Before CNC, when NC was the only system available, there was a surplus of used end mills on the market that had not been re-sharpened because it was cheaper to scrap out dull end mills without re-sharpening them than to write long complex programs to account for their smaller diameter.

Don't ask for impossible

The first thing you have to learn about using the g41 or g42 cutter comp commands is that you can't ask your machine to do the impossible. What is impossible is cutting an inside "V" with an end-mill.

If you firmly imbed this into your mind, you'll save the many hours of wasted time that I spent learning. Of course I knew that this is impossible, but I kept asking my computer to do it when I was entering the cutter offset mode. As soon as a g41 or g42 is entered into the program you can't ask your machine to cut "V" shapes without an arc being programmed that is at least the radius of the cutter you are using. What confused me was the cutter diameter that was retrieved with the d-command had an effect on whether you would or would not have an error. I believe I truly understand it now and had to rewrite several pages of garbage I wrote with confusing rules.

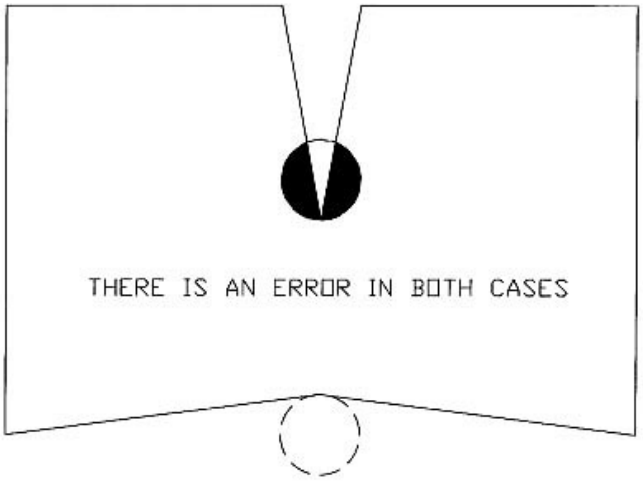


Figure 5

Safety first

I believe you shouldn't approach the work at this time in a rapid g00 mode for safety reasons; therefore, all my examples will include a short section of controlled feed rates before the point of contact is reached. My program examples will always be complete and will run on the backplot program that comes with your system. I have yet to find a program that checks out with this program that wouldn't run on the Sherline mill if the slides were located in the proper position before the program was run. The more I work with the LinuxCNC, the more I like it.

Flying Sherline Airlines requires you to learn how to land before flying

The whole trick in using g41 or g42 is getting to the starting point in a very specific manner. Also, remember that our computer is very smart and looks ahead at what you plan to do when you enter cutter comp. If you ask it to do the impossible, it will send you a message. What we want to accomplish is approach the starting point in two separate moves so that we are coming into the part like a plane on final approach to an airport. The plane flies to the first point, where it lowers it landing gear (g41—a move where the cutter comp is entered during the move) and feeds (g01) to the final landing point with its wheels (cutter) in position to land. Come in too steep and you'll create a deep V cut that is impossible to machine. (See Figure 5.)

You can overcome this error by programming a 90° radius cut

to the part. Of course, the programmed radius must be larger than the radius of your cutter. To think of a programmed move in aircraft terms you are "flaring out" for touchdown. This has a second advantage from a machining standpoint of not leaving a slight undercut that's created when a cutter is brought up to edge and immediately makes a sharp turn. This undercut is caused by cutter deflection.

The coming programming examples don't use this method of bringing the cutter up to the part because it wasn't necessary in my examples. In the real world, this will be a problem that you'll regularly encounter, and you'll have to use every trick in the book to produce the parts that are needed.

For the time being, remember that the closer you align your cutter (aircraft) with the directional edge that you want to machine (runway) the better the landing. Remember, in the world of CNC there is no gravity, and you can touch down any way you please.

Let's go back and add g41 to the circle program. Modify your circle program to read like this. Refer to the drawings with examples in this section.

The entire program:

```
%
(circle using g40, g41, g42)
g00 g90 g40 x1 y1 z1
g42 d1 x.5 y1
x0
z0
g01 z-.3 f10
g03 x0 y1 i0 j-.5
g00 z1
g40 x1 y1 z1
m2
%
```

Now I'll go through the program one line at a time

Note: Before this program can be run the tool offset has to be entered. Go to [TOOLS] and enter 0.250" (6.4 mm) for the tool length, and 0.375" (8.4 mm) for the tool diameter. You must use the [Enter] key after entering each change in order for the change to be entered into the table. Close page.

1) **%** –Last time—Start program.
2) **(circle using g40, g41, g42)** –Last time—Program Name.
3) **g00 g90 g40 x1 y1 z1** –Home.
4) **g42 d1 x.5 y1** –End Mill (aircraft) approaches part as tool compensation is entered (gear down). **d1** will define the offset as it moves to this new g42 position. We're still traveling too fast for landing.
5) **x0** – We're directly above the touchdown area and come to a halt.
6) **z0** – Fortunately we're flying the new Osprey and they can come straight down.
7) **g01 z-.3 f10** –Landing speed and touchdown from a vertical plane.
8) **g03 x0 y1 i0 j-.5** –We cut a quick circle just like last time and we're out of here.
9) **g00 z1** –Straight up.
10) **g40 x1 y1 z1** –And home.
11) **m2** – End program.
12) **%** –Park it and have a coffee. Good flight!

The diameter was generated as before, but this time the final size of this diameter will be controlled by the diameter entered in to the d1 location. By entering a diameter larger than the actual tool, which is 0.375", we are assured that the diameter will not come out undersized, because it keeps the spindle farther away from the finished edge. This can give you the opportunity to take a second pass and allow you to bring it accurately to size by changing the d1 value. You have to admit, I'm a clever old bastard.

Rectangle with cutter compensation

I think it's a good time to take a break from circles and cut something straight before you get motion sickness. Create a new file for this flight. How about a nice simple rectangle with cutter comp of course? The material we'll be serving today is 1" x 1" x 2" (25 mm x 25 mm) 6061 T6 aluminum, and our cutting speed will be 6 IPM. In the event of a crash, prepare to pick up your paycheck on the way out the door. Have a nice flight.

We are going to do a profile cut around a 1" square with a 3/8" end mill using cutter compensation.

The part shown below is the 1" square and the "Datum Point" (X0,Y0) is at the bottom left corner.

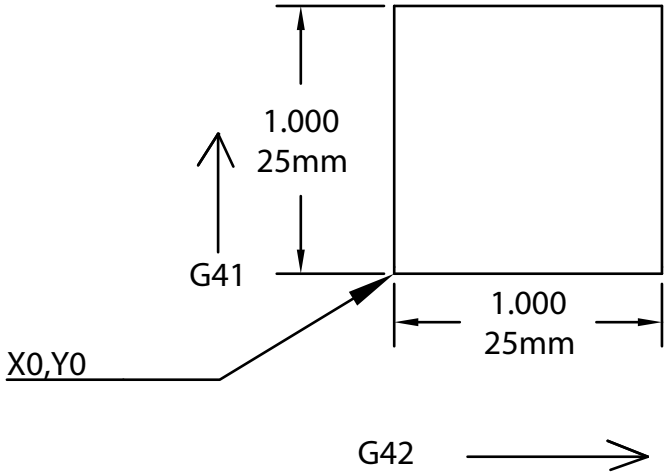


Figure 6

We are going to use a G41 which is cutter comp left and we are going to cut in a Clockwise direction. G41 CC Left means that whatever line we program, the computer is going to shift the cutter to the left of that line by an amount that is equal to the radius of the tool.

The tool length and diameter are stored on the tooling page. These are the values that the computer will use to calculate where to move the cutter.

NOTE: When initiating the cutter compensation move into the part, the cutter needs to start away from the part, and move into it. On this first move the computer will do the offset for the cutter compensation away from the line

that you are going to cut. Your initial point for this move needs to be away from the part by a distance that is greater than the radius of the tool. This move can't be a move that is perpendicular to the line to be cut. The easiest way to make this move is to start at a point that is 45 degrees away from the starting point of the line. In the program below the starting point on our first line is X0.0 Y-.050. We picked a point that is .500 away and at 45 degrees from the starting point (X-.5 Y-.550).

Program sample G41, CW cutter path, climb milling:

NOTE: Anything that is inside (parentheses) is a message and it is not read by the computer.

```
%
( tool 1 ENDMILL ROUGH D.375  C0. L2.5)
G17 G20 G40 G49 G80 G90 G00 G94 X0 Y0 Z0
(MSG, " Change to .375 ENDMILL ROUGH")
G43 G90 H1 (this line of code takes the tool length
information from the tooling page)
(Begin FEATURE 2 AXIS)
G0 X-.5 Y-.550 (you need to start your cut away
from the part by an amount that is larger than the
radius of the tool)
G0 Z.1
G1 Z-.15 F5.0
G41 D1 X0.0 Y-.050
Y1.
X1.
Y0.
X-.050
G0 G40 X-.5 Y-.5
G0 Z.1
G49 Z0.0
G90 G00 G40 G80 X0.0 Y0.0 Z0.0
M2
```

If we wanted to do conventional milling instead of climb milling (which the program above is doing), we would cut the square Counter-Clockwise and use G42 Cutter Compensation Right.

Program sample G42, CCW cutter path, Conventional milling:

```
%
( tool 1 ENDMILL ROUGH D.375  C0. L2.5)
G17 G20 G40 G49 G80 G90 G00 G94 X0 Y0 Z0
(MSG, " Change to .375 ENDMILL ROUGH")
G43 G90 H1 (this line of code takes the tool length
information from the tooling page)
(Begin FEATURE 2 AXIS)
G0 X-.55 Y-.5 (you need to start you cut away from
the part by an amount that is larger than the radius
of the tool)
G0 Z.1
G1 Z-.15 F5.0
G41 D1 X-.050 Y0.0
X1.
Y1.
X0
Y-.050
G0 G40 X-.5 Y-.5
G0 Z.1
G49 Z0.0
G90 G00 G40 G80 X0.0 Y0.0 Z0.0
M2
```

Notice that I started the program .050 before the part corner (X0) and finished the cut .050 below the part corner (Y0) to ensure that the cutter went beyond the corner to get a clean square corner. What you can do is set the value of d1 at a larger diameter and run the program through. Then set the d1 at the true diameter of the cutter used and you can see the shape of the part and then the cutter path followed using g41. Notice how the machine goes around sharp corners. Neat! This isn't a bad idea to use throughout these problems to help you better understand the process.

Planes that don't fly

In the CNC world there are three different planes. They come into play when we want our computer to do circular interpolation in a vertical plane. With this being the case, then either the x or y coordinate will have to be predominate with the Z-axis. We can control these with the g17 (x, y) g18 (x, z) and g19 (y, z) inputs. When we are working in these vertical planes generating arcs and circles, the z offset for the arc center point will be controlled by the k input, again designating the offset in the z direction given as an incremental value. Got that?

Here is an interesting little program I wrote:

The atomic circle program

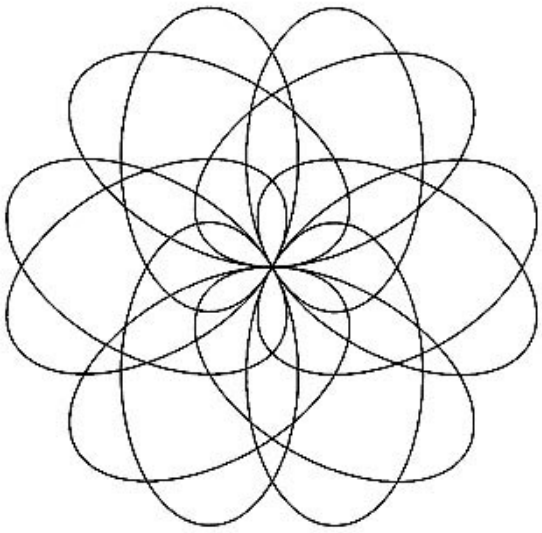


Figure 7

The entire program:

```
%
(atomic circle)
g90 g17 g40 g00 x0 y0 z0
```

```
g03 x0 y0 i0 j1 f100
g02 x0 y0 i0 j-1
g03 x0 y0 i1 j0
g03 x0 y0 i-1 j0
g18 g03 x0 z0 i0 k1
g02 x0 z0 i0 k-1
g03 x0 z0 i1 k0
g02 x0 z0 i-1 k0
g19 g03 y0 z0 j0 k1
g03 y0 z0 j0 k-1
g02 y0 z0 j1 k0
g03 y0 z0 j-1 k0
g90 g17 g40 g00 x0 y0 z0
m2
%
```

Run it on the backplot program. Millie could also run it, but you'll have to change the f100 feed program to f12 so Millie can handle it. Be sure the slides are located in a position where the slides can be moved two inches (50 mm) in every direction. Single step this program through and again be sure you are positive that you can understand every move made so well that you can explain to someone what the move is and why it is going to make it this or that way before going on. Our class is very understanding and we are all going to wait for you while you go back and catch up, but don't let it happen again.

The Great Race

Instead of R/C, let's go PC aircraft racing. We'll watch the race using the backplot program. First, set the tool diameter to 0.600" in d1. Next, we'll write a short program that will lay out a triangular course. Our racing aircraft will be a micro delta aircraft that has strange flying characteristics allowing it to fly forward or backward in a nose down attitude (sometimes it just takes a good imagination). Racing rules state that it can only go around the pylons by turning left during the race. Cutter comp will make this easy because it will automatically calculate the tangent points needed to run this program. By juggling the turn-in points we can shorten the course, thereby lowering the time. There'll be more rules but first lay out a triangular pylon race course. The vertical lines are the pylon marks. The start finish line is included, and the program will be designed to lay the course out at the beginning of each new race.

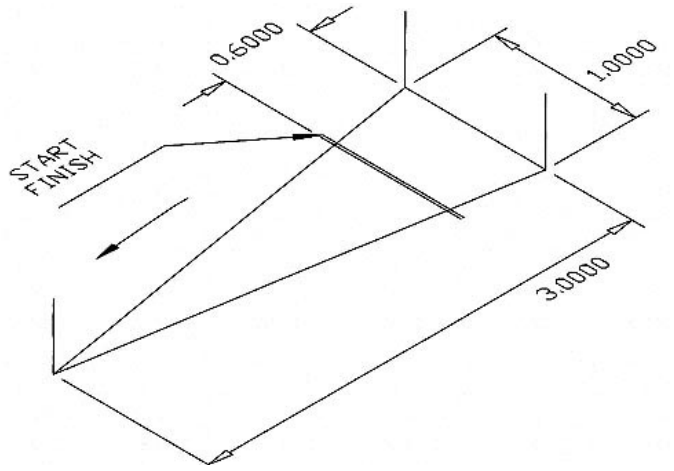


Figure 8

The entire program:

```
%
(EMC race course)
(Set tool diameter D1 to .600)
g00 g17 g40 g90 x-1 y1 z0
g01 y.5 f100
z.5
z0
x-4 y1
z.5
z0
x-1 y1.5
z.5
z0
y.5
g00 x-1.6 y.8
g01 y1.8 f100
(course complete)
(Taxi to starting line)
x-1.6 y1.3 f15
m00
```

This will pause the program while you get ready to set your stop watch. To start the race you'll have to use [Resume] button because the [Run] button will not restart the program when it is in a "paused" mode caused by the m00. Now for the race…

```
(Use resume button to start race)
x-2 f6
x-4 y1.4 z.5 f10
g03 x-4 y.6 i0 j-.4
g01 g42 d1 x-3 y.5 f20
x-2
x-1 y.5
y1.5
(lap 2)
x-4 y1 f25
```

x-1 y.5
y1.5
(lap 3)
x-4 y1 f30
(top speed)
x-1.123 y.578
y1.373
(lap 4)
x-3.862 y.961
x-1.123 y.578
y1.373
(lap 5)
x-3.862 y.961
x-1.123 y.578
y1.373
x-1.6
(race over)
g40 x-3.5 z1
x-4 y1.5
g03 x-4 y.5 i0 j-.5
g01 g90 x-2
g18 g02 x-2 z1 i0 k1.5
g01 g90 g17 x2
g02 x2 y-.5 i0 j-.5
g01 x-2 z0 f20
x-2.5 z.15 f15
x-3 z0 f10
x-3.2 z.1 f5
x-3.4 z0
x-3.6 z.05
x-3.7 z0
g03 x-3.7 y-.9 i0 j-.2
g01 g90 x-1 f25
g03 x-.5 y-.4 i0 j.5 f15
g01 y.5
g03 x-1 y1 i-.5 j0
g40 g90 g00 x-1 y1 z0
m2
%

The same program with comments from the pilot:

x-2 f6 –Gaining airspeed and lift-off

x-4 y1.4 z.5 f10 –Climbing out and headed for turn 1

g03 x-4 y.6 i0 j-.4 –Large radius turn to prevent stall

g01 g42 d1 x-3 y.5 f20 –Turn on navigation aid to help in turns

x-2 –2nd point for entering g42

x-1 y.5 –Turn 2 point

y1.5 –Turn 3 point

(lap 2)

x-4 y1 f25 –More speed and back to turn 1

x-1 y.5 –Turn 2 point

y1.5 –Turn 3 point

(lap 3)

x-4 y1 f30 –We're at top speed headed for 1

(top speed)

x-1.123 y.578 –We tighten up the course for the rest a race

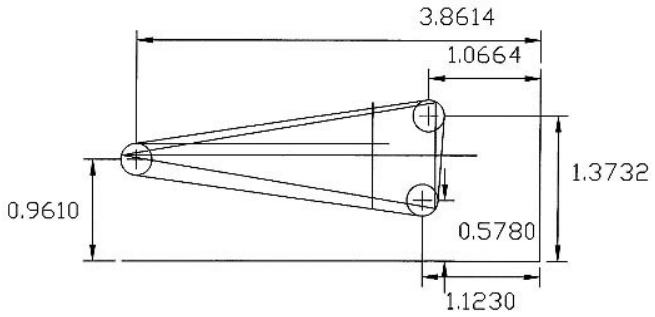


Figure 9

I cheated and used AutoCad®…

y1.373

(lap 4)

x-3.862 y.961

x-1.123 y.578

y1.373

(lap 5)

x-3.862 y.961

x-1.123 y.578

y1.373

x-1.6

(race over) –We won

g40 x-3.5 z1 –Turn off the navigational aid and climb out

x-4 y1.5

g03 x-4 y.5 i0 j-.5 – A nice lazy fast turn

g01 g90 x-2

g18 g02 x-2 z1 i0 k1.5 –We change planes and do a victory loop

g01 g90 g17 x2 –Enter the landing pattern back in x, y plane

g02 x2 y-.5 i0 j-.5 –Turn onto final landing approach

g01 x-2 z0 f20 –Throttle back, oops, dam it (bounce)

x-2.5 z.15 f15 –Flare it out, dummy

x-3 z0 f10 –Damn, right in front of everyone

x-3.2 z.1 f5 –I give up. I never did like tail-draggers

x-3.4 z0

x-3.6 z.05 –At least I'm still alive

x-3.7 z-.01 –I think I dented their runway

z0

g03 x-3.7 y-.9 i0 j-.2 –Let's get out of here before they remember me for that lousy landing

g01 g90 x-1 f25 –A nice fast taxi home

g03 x-.5 y-.4 i0 j.5 f15

g01 y.5

g03 x-1 y1 i-.5 j0

g40 g90 g00 x-1 y1 z0 –Home at last. I wonder if they'll remember my lousy landing more than our win

m2

%

NOTE: For those who have purchased the complete CNC system with computer, this program has been added to the "g-code" folder as "TheGreatRace.ngc" so that you can open it and test run it without having to retype everything.

Your homework assignment is to figure out this program that I just spent more time on than I care to admit. There isn't anything new in it, and you have to understand all the code used. It may be a little "far out" but it is easy to follow if you know about aircraft.

Your next assignment is to see if you can lower my time from leaving the starting line until the race is over time is displayed. After lap 2 is completed, the turn-in points are the only thing you have to work with, and you can't change the feed-speed or the feed override on the panel.

My time was an astounding 1 min 42.7 sec. Your job—lower it.

Test two

Up until now all you have done is copy my examples into your computer, and you're probably thinking this CNC stuff really isn't that hard after all. It's time to bust your bubble before you get a big head and become a pain.

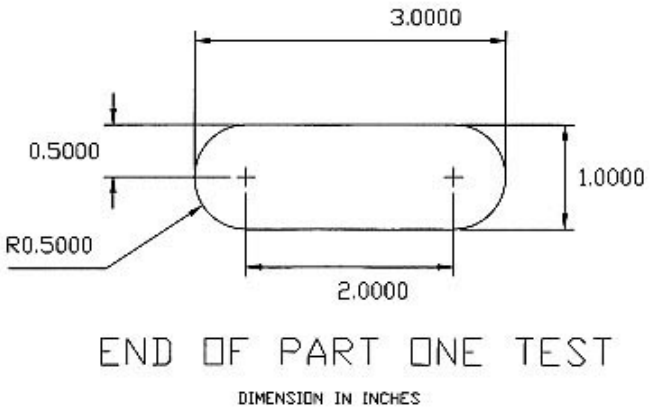


Figure 10

I want you to write a complete program on your own to profile this part. To pass the test, you must use cutter comp. I have taught you all the fundamentals to write this program, and the only suggestion I will make is to test the program a segment at a time as you write it. That's the great advantage of having Ray Henry's backplot program to work with.

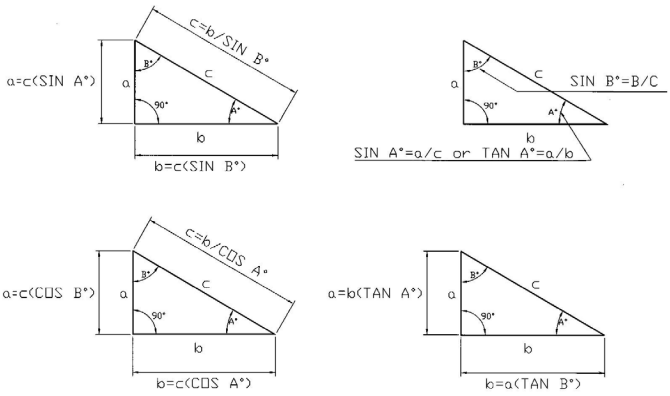
Change a couple of dimensions and make one end square and take another test.

There isn't any sense in entering the next section unless you can easily write these programs. If not, go through that section again and again until it is second nature.

Back to school

We're on the move again and leaving route g41 for a while, but I'm sure we'll return. We are headed back to school for a refresher class on trigonometry, more commonly called trig. Today all you need is a $10 calculator that has trig functions or a hook-up to the Internet where they have a site that will solve your problems instantly by entering any two known sides or a side and an angle other than 90°. See the handy trig calculator at http://www.pagetutor.com/trigcalc/trig.html

It just doesn't get any easier than this, or you can use the formulas below and a calculator. All you have to enter are any two known things about the right triangle you want to solve other than the 90-degree angle which is always present.



Basic Trigonometric Functions

Also: $c^2=a^2+b^2$ $c=\sqrt{(a^2+b^2)}$ $a=\sqrt{(c^2-b^2)}$ $b=\sqrt{(c^2-a^2)}$

Figure 11

When programmers solve for tangent points, they usually start off knowing the hypotenuse of the triangle they are working with, because it is the radius of the arc of which they are solving for the start and stop points.

The hardest part of this part of the course isn't solving a right triangle. It's finding the correct right triangle to solve to give you the answer to determine your tangent coordinates that will determine the start and stop points of an arc. You'll remember that I told you that these points must be given to an accuracy of four decimal places in order to work in unison with the computer. It has nothing to do with the accuracy you require, but only with what our loyal computer requires.

I dug out my old calculator to solve these problems and then checked my answers against what I had already calculated in AutoCad®. I'm going to assume you will be making the same mistakes I did, so I'll pass my thoughts on about the problems I had. When you're dealing with angles less than 10° changes are subtle and errors will not stand out. Be absolutely sure that you are using the correct equation to work with the information that you have available. All you need are two sides or a side and an angle other than 90°.

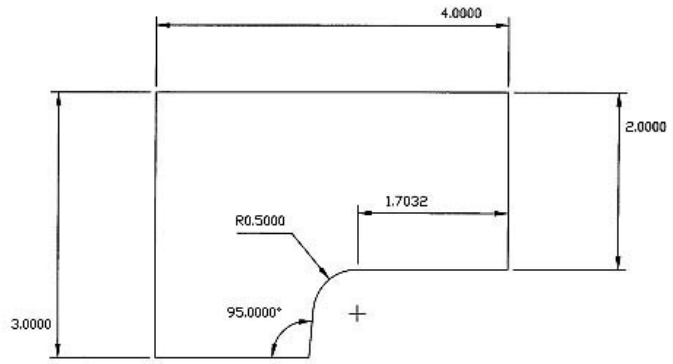


Figure 12

Rectangle, arc and tangent program

The first example I'm going to explain looks deceivingly simple and is typical of what you have to start with, and believe me it isn't easy, so pay attention. I started off this drawing by putting a circle in a rectangle in no particular position on the X-axis and then putting a 5° line against the tangent point of the circle. My next task was to figure out the missing points with a calculator. The only practical way to accurately figure these points out is with trig.

First we have to find and solve the missing values for the right triangles shown in Fig. 10 below. The key triangles are shown in solid black.

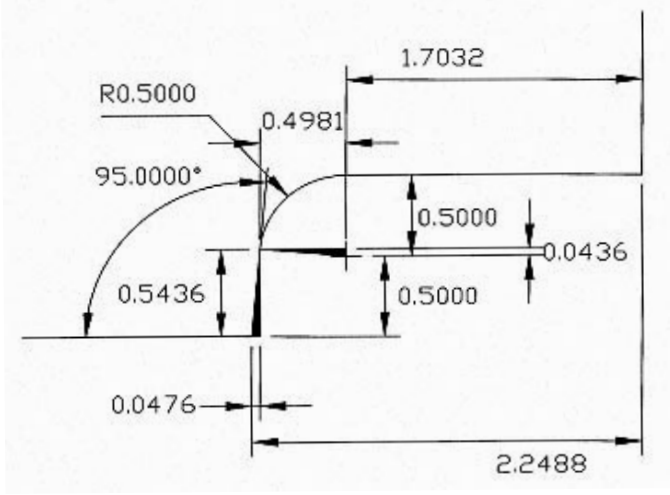


Figure 13

First, what two things do we have to solve this triangle? We have the hypotenuse because it is the radius of the arc and the 5° angle. From our tables we see that the 0.0436 was derived by multiplying the hypotenuse (0.5) times the sin 5°; and the 0.4981 was the product of 0.5 x cos 5°. This will give us our end-of-arc position.

The 0.0476 has to be calculated to give us the final X-axis point of the angle. We can easily calculate the 0.5436 because it is the sum of 0.500 and 0.0436. This gives us one side of the triangle that we know is 5°; therefore the dimension needed is the sin 5° x 0.5436. Adding up these 3 dimensions we end up with the 2.2488 X-axis dimension. We now have all the numbers we need to generate the code to produce this part.

Take one more look at the complete drawing before starting to write the program. The main thing I want you to learn is how important it is to solve for the correct triangle to get the answers you need, and the many times you have to solve more than a single triangle to solve what may seem as a straightforward problem.

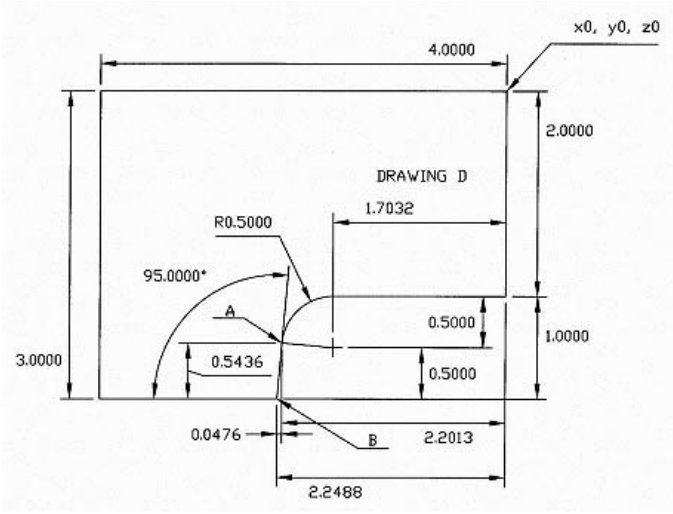


Figure 14

This program is just an exercise, and I'm not going to write any Z-axis code. The 0 starting point is located in the upper right-hand corner. Of course, we'll also use cutter comp to generate this shape. Now, read the code I wrote and examine Fig. 11 again and visualize where the starting and stopping points are in your mind before you run the program in backplot. This is what we are striving for, because a real professional programmer can read code like a letter.

The entire program:

```
%
(rectangle, arc and tangent)
g00 g40 g90 x0 y1 z0
g41 d1 x0 y.5
g01 y0 f16
y-2.000
x-1.7032
g03 x-2.2013 y-2.4564 i0 j-.5
g01 x-2.2488 y-3.000
x-4.000
y0
x0
x.5
g00 g40 g90 x0 y1 z0
m2
%
```

With comments

```
%
g00 g40 g90 x0 y1 z0
g41 d1 x0 y.5
g01 y0 f16
```

Last time, I started at y1 position so I could enter into the cutter comp mode in two moves and coming in from the correct direction (remember—gear down-touch

down).

y-2.000

x-1.7032

g03 x-2.2013 y-2.4564 i0 j-.5 –Last time, The x and y positions where the arc ends.

g01 x-2.2488 y-3.000

x-4.000

y0

x0

x.5 –Exiting g41 mode in correct direction.

g00 g40 g90 x0 y1 z0

m2

%

Work out a couple of similar problems on your own while I get ahead of my students again. Boy, you guys are smarter than I thought you were.

The plot is about to thicken with your next problem that I called the pulley program.

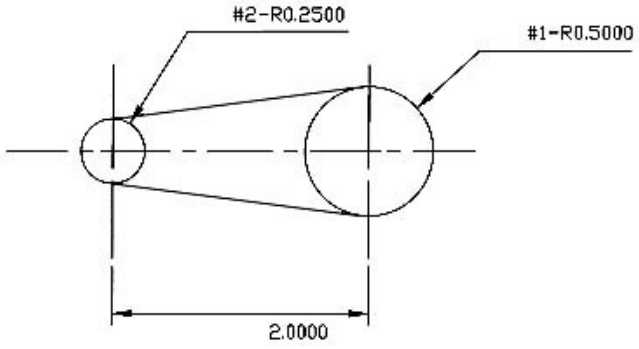


Figure 15

In this problem all we know is the radii of both circles and the distance between them, yet we have to arrive at the tangent points.

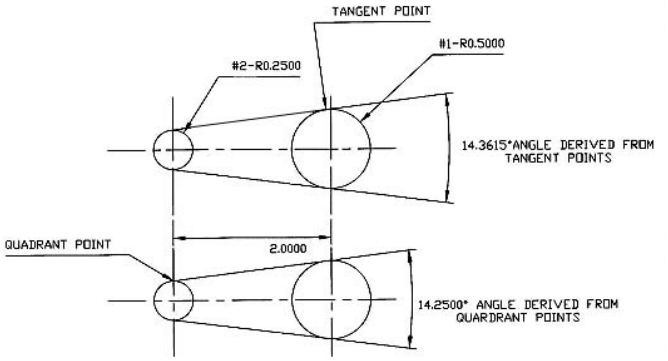


Figure 16

Here is the real problem. The seemingly obvious way of solving this problem would be to solve the angle that can be generated with the differences between the radii and the distances between centers and then use this information to come up the needed points. Wrong. Compare the two angles in Fig. 14 below. Although the error isn't that great for laying out sheet metal work, it's a gigantic error for a computer to swallow, and you have to be accurate to four places.

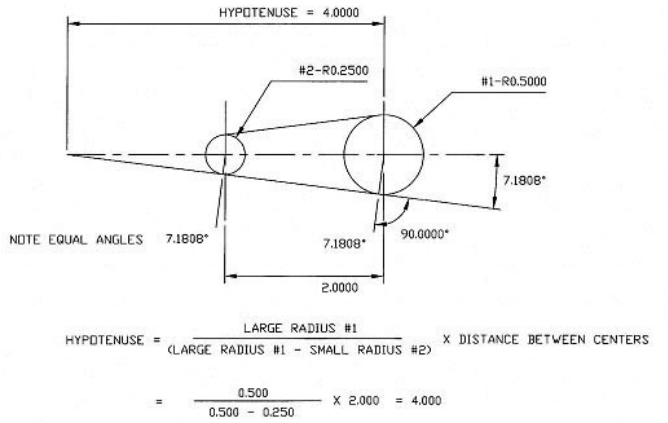


Figure 17

Believe me, I studied this problem for a long time before I was able to solve it. I also came up with unusually simple way to arrive at the second side of the right triangle so the problem was solvable. I don't want to blow my own horn, but it is extremely rare that an individual like myself, who hasn't studied high math, will ever come up with a complex mathematical rule on his own that works, whether mathematicians know about it or not.

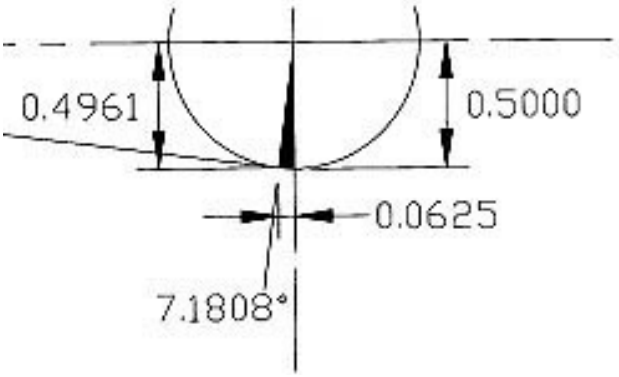


Figure 18

Now we can calculate the angle because the sin B = hypotenuse ÷ side b. With this angle we now know the remaining missing tangent points can be solved with your calculator. Note that once we know that angle, it should be noted the same angle falls in two other places as shown in Figure 18 above. This allows us to solve the remainder of the problem in Figure 19 in the same manner as we did in Figure 18.

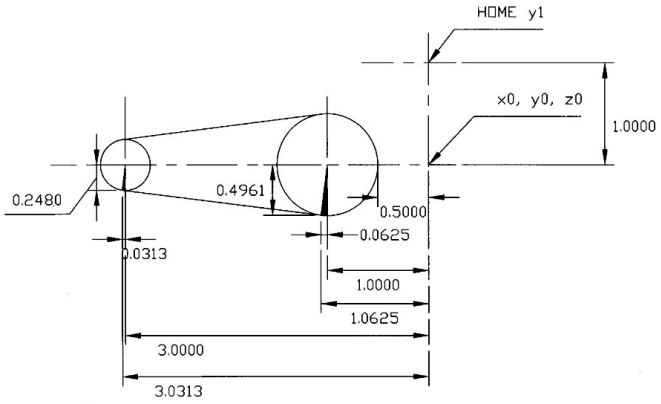


Figure 19

Let's write the code for this simple yet not-so-simple shape. One thing I must state again is that I'm having a terrible time with careless errors. I think the problem is that I'm more worried about how I'm going to explain each problem than I am in entering the correct numbers. Don't fall into my trap. You don't have to explain to me how you did it, so learn from my mistakes. Become a hermit while you are studying so you don't make careless errors.

The entire program:

```
%
(pulley program)
g00 g40 g90 x0 y1 z0
x-.500
g41 d1 y.500
g01 y0 f12
g02 x-1.0625 y-.4961 i-.500 j0
g01 x-3.0313 y-.2480
g02 x-3.0313 y.2480 i.0313 j.2480
g01 x-1.0625 y.4961
g02 x-.500  y0 i.0625 j-.4961
g00 g40 x0
x0 y1 z0
m2
%
```

I believe that you should be able to follow the pulley program without comments now that you have calculated the tangent points.

Our next problem is linking to arcs together, and in this case the arcs are going in opposite directions. What I want you to notice in this particular program is the fact that you enter cutting the second 0.300 radius from the opposite direction as you did in the same radius on the opposite side. This changes the i and j values going in and coming out.

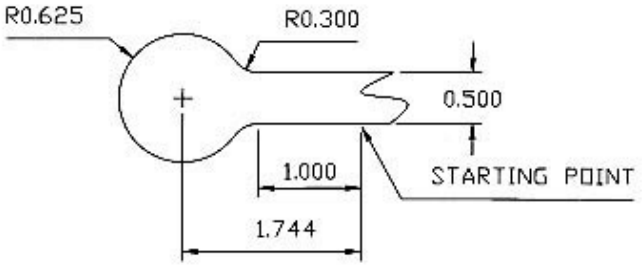


Figure 20

When you have to solve a problem like this, the first thing you have to do is determine what you have to work with. In this case, we know the two radii of the arc and the width of the straight section leading to it. Studying the drawing you will find that with the information known there is only one position where we can begin to solve for the missing dimensions to write our program.

The radii will give us the hypotenuse of the triangle we need to form, and we can determine the second side of the triangle by adding together the radius of the first arc and the distance the second arc's center is from the edge. Now we can calculate not only the starting point of the first arc ($b=\sqrt{c^2 - a^2} = 0.7437$), but also the angle of that triangle ($\sin A° = a \div c = 36.4837°$). With the angle now known we can calculate the x, y ending points of the arc, and we have already calculated the i and j values for that radius.

The ending point of our first arc is also the starting point of our second arc; therefore, because the part is symmetrical, the ending point of the second arc is located the same distance from the center point of that arc. To start the third arc new i and j values must be determined, because we are approaching this arc from the opposite direction even though this arc is identical to arc one.

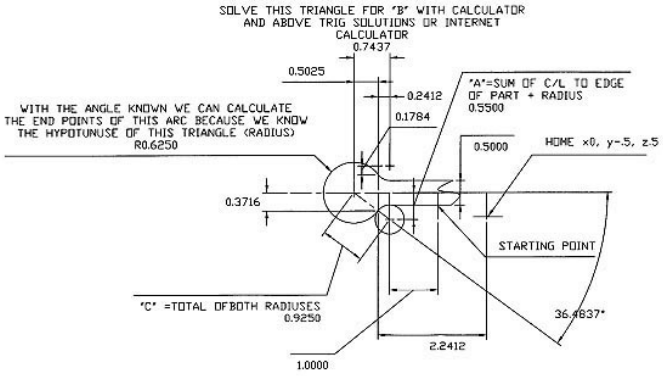


Figure 21

Don't just look at my answers and assume they are right. In the real world you don't solve problems you know the answers for. I'm writing this stuff for you to teach you how to do it. I already know. Be original and change a few basic dimensions and see if you can be confident that your answers are correct. That's what it is all about. Don't find out that you don't know how to do it by making bad parts. We are machinists who don't have erases to work with. Don't become one of those "Oh shit" machinists who can only have their mind work at full potential after they screwed up! Another free lecture by J. Martin. Now let's write the code for this beauty. Also, I added a couple of Z-axis movements that might be used in the real world.

```
%
(rod-end)
g00 g40 g90 x0 y0 z0
g41 d1 x-.75 y-.25
g01 x-1 y-.25 f25
z-.5
x-2
g03 x-2.2412 y-.3716 i0 j-.3
g02 x-2.2412 y.3716 i-.5025 j.3716
g03 x-2 y.25 i.2412 j.1784
g01 x-1
x-.75
z0
g00 g40 x0 y0 z0
m2
%
```

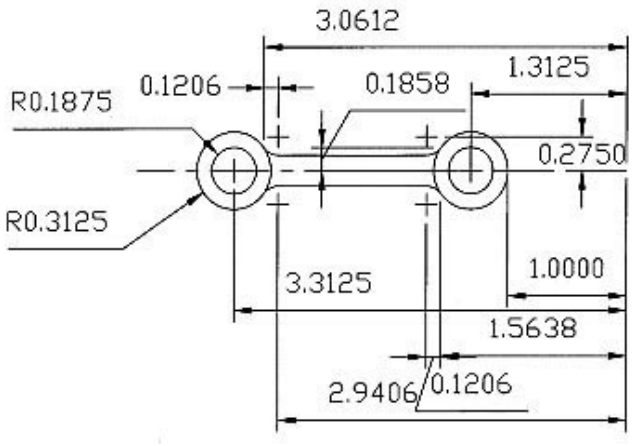


Figure 22

The rod program adds a useful new tool to our collection. The "r" letter designates "radius" when it's used in the same block as the circle command g02 or g03. This simplifies code writing because it eliminates the need to calculate the "I" and "j" points that are always changing as seen in our last example, however, the start and end points of an arc must be calculated to the same degree of accuracy. I felt obligated to teach you the hard way first because many older CNC systems don't have the advantage of the r command, and that's the type of systems that many beginners will end up running. You will see in the rod program just how much tedious code writing it eliminates.

(For this program you will need to add a new Tool #1 listed as .250" long and .125" diameter.)

```
%
(Connecting rod)
g00 g90 g40 x0 y0 z0
g01 x-1 y.5
g41 d1 y.1
g01 y0 f15
g02 x-1.5638 y-.1858 r.3125
g03 x-1.6844 y-.125 r.15
g01 x-2.9406
g03 x-3.0612 y-.1858 r.15
g02 x-3.0612 y.1858 r-.3125
g03 x-2.9406 y.125 r.15
g01 x-1.6844
g03 x-1.5638 y.1858 r.15
g02 x-1 y0 r.3125
g01 y-.1
g00 g40 x-.5
x0 y0 z0
m2
%
```

EMC Tip—Pausing a cut during a long program

If for some reason you have to stop your machine while it's running a long program and you are planning to complete the part the next day, the safest way to accomplish this is to stop the program by using [FEEDHOLD]. Turn the spindle and servo driver off, but leave the computer on. The next day you can restart your program by simply turning the drivers and spindle on and clicking on the [CONTINUE] button. In this case the [FEEDHOLD] button becomes the "Continue" button when it is used to stop the slide movement. If possible, try to stop the operation at some point where the cutter is either not cutting or is cutting in an area where finish doesn't matter. It's also a good idea to write down the zero positions of the handwheels and slides just in case you have to use the emergency stop.

The "o" Command

This command allows you to create Sub Programs.

A sub program is a program within a program that is defined by the letter "o". LinuxCNC will not allow you to call up a sub program from another file. These Sub programs (or Subroutines) are entered at the beginning of the Main Program and are labeled by the letter "o" followed by a number followed by the word sub. You can't use a letter or letter and number to define a sub program. I find it wise to further define the sub program inside parentheses.

Example: **o100 call (mill 2″ hole)** A sub program may call up another sub program and they call this procedure "nesting." LinuxCNC will allow you to nest as many as ten sub programs together.

When the sub program is complete, you enter Example: **o100 endsub** The words I highlighted in red must be entered in order for the "o" command to function.

Sub Programs offer a programmer a way to shorten a program by thousands of lines. They are similar to "canned cycles." Prior to canned cycles if you wanted to peck drill deep holes, you had to program all of the Z axis moves one line at a time. This could take several lines of program for each hole. Canned cycles allowed you to put all of the drilling information on one line, and then program position moves for all of the holes.

Some general areas were sub programs are used are. 1. On parts that have the same shape cut in several different places. 2. Machining the same parts on several different fixtures. 3. Machining the same shape in different locations. Sub programs can be short and simple or very long and intricate.

LinuxCNC uses "o" codes (this is the letter o, it is not Zero) for programming sub programs.

Sub programs should be written in a g91 incremental format. If g90 was used the sub program would only run in a single location, however, you could use g90 if you reset your home position with a g92 before calling up your sub program. This would be a better choice for long complex sub programs that would be difficult to read if written in incremental movements. Your choice.

This would be a typical sub program:

o200 sub (the starting code for the sub program within the main program. The word sub is required)

```
g91 g01 z-.3 f15.0
g00 z.300
x.25
```

g01 z-.3
g00 z.3
y.25
g01 z-.3
g00 z.3
x-.25
g01 z-.3
g00 z.3
o200 endsub (the ending code for the sub program within the sub program will locate the cutter in the exact same spot it was before the sub program was run)

This would be a typical main program that calls a sub program:

In the main program body you will move the cutter to the starting position for the shape or operation that you want to machine. Then you will call up the sub program using an o-code. Your program will look something like this:

%
(Program Name)
o200 sub (It's best to have a standard format and my suggestion is to always enter all your sub programs at the beginning of your program.)
g91 g01 z-.3 f15.0
g00 z.300
x.25
g01 z-.3
g00 z.3
y.25
g01 z-.3
g00 z.3
x-.25
g01 z-.3
g00 z.3
o200 endsub
g90 g00 g40 g80 x1.0 y1.0 z0 (move into position for the starting point of the sub program)
o200 call (call up sub program o200)
g90 g00 x1.0 y-1.0 z0 (A g90 must be entered in the line of code calling for its next position move) (Be sure to have your spindle high enough to clear all obstacles such as clamping devices.)

o200 call (call up sub program o200)

As many times as needed.

Now every time the machine sees "**o200 call**" in the program, it will search for "**o200 sub**". Then it will run that section of the program until it reads "**o200 endsub**". At this point, the spindle will go back to the position it was before the "o200 call" was initiated and continue to run the main program.

%
(Main Program with a Sub Program Nesting Sample)
o200 sub (4 hole pattern)
g91 g01 z-.3 f15.0
g00 z.300
x.25
g01 z-.3
g00 z.3
y.25
g01 z-.3
g00 z.3
x-.25
g01 z-.3
g00 z.3
o300 call (x move)
o200 endsub
o300 sub (mill shape)
(mill shape g-code)
o300 endsub
(Main Program with a Sub Program Nesting Sample)
g90 g00 g40 g80 g54 g17 x0 y0 z0
g90 g00 x1.0 y1.0 (move into position for the starting point of the sub program)
o200 call (4 hole pattern)
g90 g00 x4.0 y1.0
o200 call (4 hole pattern)(Which also calls o300 sub (mill shape))
g90 g00 g40 g80 x0 y0 z0
m2
%

Again:

1. All o-codes must be defined by numbers (EX. o200, o154, etc).
2. No letters or words are allowed after the o-code number except the words (sub, call and endsub).
3. No comments are allowed unless they are inside parentheses (xxxxx).

More handy g-codes

g-codes that you can learn as you need them

Canned Cycles

There are quite a few canned cycles designed into the LinuxCNC. They can save a lot of repetitious programming. There are some others that LinuxCNC does not support or Sherline machines cannot perform, so we'll only consider the canned cycles that are useable with Sherline CNC machines. These commands are primarily used for drilling and boring cycles to save programming time when you have a lot of

repetitious operations to perform, like a series of holes. However, at this point you have learned enough g-codes to be able to make most parts, and I believe you should take a break from learning and take the time to become more proficient with the skills you have learned up to now. Although these remaining commands are easier to learn than the ones you have been using, I'd suggest you take a look at my last two programs at the end of this section and read my closing comments before going on. Though you now know enough to make parts without them, the following g-commands, when used appropriately, can eliminate many lines of code from your programs and allow you to enjoy the full power of EMC.

The G80 Command

The g80 command is used to cancel a canned cycle. It's used in a similar fashion as the g40 and g49 commands.

The g81 Command

The g81 command is intended for drilling, and because x, y movements usually take place with the drill located well above the part to avoid accidental collisions with fixtures, the most efficient method is to lower the drill in rapid (g00) to the starting point. However, the computer needs to know the drill starting point to accomplish this. This second z-axis position is given by entering an r-command defining an absolute position in the same line of code that contains the g-command that requires it. (Note that the r-command is also used as a method of entering the radius when programming movements that produce circular shapes, however, the EMC can tell the difference based on when it is used.) If the depth of the hole exceeds 1.5 times the diameter of the drill you should use the g83 command. Also note that we are using the absolute mode.

Example:

```
%
g90 g40 g20 g00 g80 z2
x0 y0
g81 x2 y1 z-0.50 r0.010 f3
g00 g80 x0 y0
%
(Consider the top of your part to be the 0 position.)
%
```

An explanation, line by line:

g90 g40 g20 g00 g80 z2 – It's a good idea to get the z-axis up out of the way before moving the x- or the y-axis. For example, consider what happens if you start the program from a different position than you intended and the drill hits the fixture on the move to its home position.

x0 y0

g81 x2 y1 z-0.50 r0.010 f3 – This line of code contains all the information to put the g81 canned cycle into effect, and you'll be able to repeat it each time a new x, y position is given.

g00 g80 x0 y0 – When you have completed drilling all your holes, a g80 is entered to cancel the canned g-cycle command.

For drilling more than 1 hole, add the needed x, y positions for each hole; for example:

```
%
g40 g20 g90 g00 g80 z2
x0 y0
g81 x2 y1 z-0.50 r0.010 f3
x3
x4
g00 g80 x0 y0
%
```

The g82 Command

G82 is intended for drilling when you want a dwell at the bottom of the hole. (I find this a bad idea when drilling, because some of the more exotic metals like stainless steel may work harden and cause the drill to become dull sooner than it should.)

1. Move the z-axis only at the current feed rate to the z position.
2. Dwell for the given number of seconds. The length of the dwell is specified by a p# designation in the g82 block and time entered would seldom be more than a second.
3. Retract the z-axis at rapid rate to clear z. The motion of a g82 canned cycle looks just like g81 with the addition of a dwell at the bottom of the z-axis move.

Example:

```
%
g90 g40 g20 g80 g00 z2
x0 y0
g82 x4 y5 z-0.50 r0.010 p0.2 f3
g00 g80 x0 y0
%
```

The new command to learn is command is **p**.

The g83 Command

G83 is intended for deep drilling because you'll break the drill if you don't retract it periodically. When you retract the drill it allows the chips to fly off and lets cutting fluid wet the bit and the work before they gall or weld themselves to the drill bit. The general rule when using a CNC machine is to have the drill cut no deeper than its diameter without retracting it for the above reasons. This is called peck drilling.

The peck drilling cycle should also be used when center drills are used to create starting points for your drills. Center drills twist off very easily because their drilling section is not relieved as much as a drill is. This is done to force the drill to seek center, however, they become very vulnerable to breakage. The z-axis should be retracted when center drill's depth has reached 60% of the needed depth. Note

that drills will wander by a far greater amount than you can imagine (especially if they are old and dull) if a center or spotting drill isn't used to develop a starting point. I prefer spotting drills (short drills with a 90 degree point that cuts to center) over center drills because there is no small point to twist off. They also have a proper chamfer angle and a faster machine operation.

This cycle takes a q value given as an increment value. This represents the increment of movement along the z-axis before the drill is retracted to clear the chips. Again, machinists refer to this as peck drilling.

1. Move the z-axis only at the current feed rate downward by delta or to the z-position, whichever is less deep.
2. Retract at traverse rate to clear z
3. Repeat steps above until the z-position is reached.
4. Retract the z-axis at traverse rate to clear z.

Example using a ¼" drill and putting 3-holes in 1.00" deep:

```
%
g00 g40 g20 g90 g80 z2
x0 y0
g83 x1 y1 z-1 r0.010 q.25 f2
x2
x3
g00 g80 x0 y0
%
```

The new command to learn is q.

The g84 Command

G84 is intended for right-hand tapping. The Sherline CNC system doesn't have any provision to accomplish this.

The g85 Command

G85 is intended for boring, which is a motion very similar to G81 except it adds feed out. Usually you use a feed rate around 0.001" per revolution.

Therefore, 1000 RPM = 1-inch feed rate and so on. The reason you want to feed a boring tool out is to eliminate a spiral tool mark that would result from a rapid retract; therefore, the tool will feed out at the same rate as it was fed in. The tool will usually take a second cut as it is withdrawn because a different cutting edge is cutting on the way out. For an explanation of why you bore holes rather than drill them see the boring head instructions at www.sherline.com/wp-content/uploads/2015/11/3054inst.pdf.

Example:

```
%
g90 g40 g80 g00 g20 z2
x0 y0
g85 x2 y1 z-0.50 r0.010 f3
g00 g80 x0 y0
%
```

The g86, g87 and g88 Commands

G86, g87 and g88 are intended for boring with starts or stops of the spindle, so these are not used for the Sherline CNC system because Sherline machines do not have spindle motor control.

The g89 Command

G89 is intended for boring and uses a p value, where p specifies the number of seconds to dwell at the end of the hole. The problem with dwelling at the bottom of a bored hole, especially with a machine as light as a Sherline, is that tools have a tendency to chatter. I usually don't bore a hole to the exact bottom unless it is absolutely necessary. Remember that in the Sherline general instructions, which I hope you have read, chatter is usually cause by taking too heavy a cut, but it can also be caused by having a cut so light that the tool skips around on the surface. Also consider the fact that chatter can damage the hole, because as the chatter occurs at the bottom of the hole, the tool will usually remove material from the sides and cause it to be oversize.

1. Move the z-axis only at the current feed rate to the z position
2. Dwell for given number of seconds p1 (1 second)
3. Retract the z-axis at the current feed rate to clear z. This cycle is like G82 except that the tool is drawn back at feed rate rather than rapid.

Example:

```
%
g90 g40 g80 g00 z2
x0 y0
g89 x1 y1 z-0.50 r0.010 p0.5 f3
g00 g80 x0 y0
%
```

For boring more than one hole, add the needed x, y positions for each hole.

The L and the g91 Command

NOTE: Although I recommend using lower case letters in your programs because I believe it makes the code easier to read, the letter l is a special case because it is easily mistaken for the number one; therefore, I use the upper case letter L. (You might also want to consider this for the letter Q.)

It's time to learn about drilling a series of holes without writing any unnecessary lines of code. The L command will allow you to drill or bore many holes on either the x- or y-axis with a single line of code, but it can be tricky to use at times. Unless this is something your need to be doing I'd skip it for the time being.

The reason confusion can quickly develop is you have to be in g91 (incremental) in order to implement this function and many things change. When you're in g90 (absolute) the r value is given as a point, however, in g91 you use the actual distance the r point is from where you are located, and this will always be a negative number. The z programmed distance in the canned cycle g-code is now given as the

distance in incremental from the r-point. This will also be a negative number and many times the number will be less of a negative number than the r number as in the example below. Note that we are using the g98 command in this series of examples.

(**Note:** The letter "L" is capitalized in the following example so it is not confused with a number "1".

Example using a ¼" drill and putting 3-holes in a row ¼" deep:

```
%
g00 g20 g40 g80 g90 z0.50
x0.50 y0
g91 g81 g98 x1 y0 z-0.30 r-0.45 L3 f3
g00 g90 g80 x0.50 y0
%
```

Example using a ¼" drill and putting 12-holes ¼ " deep:

```
%
g00 g20 g40 g80 g90 z0.50
x0.50 y0
g91 g81 g98 x1 y0 z-0.30 r-0.45 L4 f3
x0 y1
x-1 y0 L3
x0 y1
x1 y0 L3
g00 g90 g80 x0.50 y0
m2
%
```

An explanation of each line of code:

%

g00 g20 g40 g80 g90 z0.50

x0.50 y0 – Home position

g91 g81 g98 x1 y0 z-0.30 r-0.45 L4 f3 – First row with 4 holes, in increments of 1 along the x-axis from (x1.5, y0, z0.50). Same motion as described in previous example, except for number of holes (number of repetitions L is set to L4). The g98 command is required to have the z-axis at its original starting position, which is 0.500" above the part because we are in incremental (g91).

x0 y1 – 1st hole in the second row, increments are x = 0, y = 1; therefore, the tool travels from the last hole in the first row to the 1st hole in the second row for 1 along y-axis and makes one hole (motion along z-axis is the same as described in previous example).

x-1 y0 L3 – Next 3 holes in the second row, increment for x = -1, y = 0. Since L3 is number of repetitions, 3 holes in increments of x-1 along the x-axis are going to be drilled (motion along z-axis is the same as described in previous example).

x0 y1 – 1st hole in the third row, increments for x = 0, y = 1, tool travels from the last hole in second row to the 1st hole in the third row for 1 along y-axis and makes one hole (motion along z-axis is the same as described in the previous example).

x1 y0 L3 – Next 3 holes (L3) in the third row in increments of 1 along x-axis, increments for x = 1, y = 0, (motion along z-axis is the same as described in the previous example).

g00 g90 g80 x0.50 y0 – Back to home position, g80, g90, cancel canned g-cycle and incremental distance mode.

More on the "L" Command…

Sherline's shop foreman Karl Rohlin offers this sample "L" program, which is based on one we used in our own factory. It uses a standard g81 canned cycle with the "L" added to the canned cycle. (**Note:** Again, the letter "L" is capitalized here because the lower case letter "l" is often confused with the numeral "1".)

This is a g81 canned cycle that uses an "L" function to drill multiple, evenly spaced holes. (Drill 5 holes .300 deep ¼" apart with a canned cycle g81)

```
%
n1 g90 g00 g40 g49 g80 x0 y0 z0
n5 g01 f50 x0
n10 g91 g81 x.25 y0 z-.3 r-.1 L5
n15 x0 y.25 r0
n20 x-.25 y0 L4
n25 x0 y.25
n30 x.25 y0 L4
n35 g80g90 g00 x0 y0 z0
n40 m2
%
```

Here is the above code with a description line by line.

%

n1 g90 g00 g40 g49 g80 x0 y0 z0 (Cancels all canned cycles and offsets and moves to Absolute Zero in the X, Y and Z)

n5 g01 f50 x0 (Moves to x0 at a feed of 50)

n10 g91 g81 x.25 y0 z-.3 r-.1 L5 (Moves in Incremental to x.250, rapids down to z-.1, Drills to z-.4, rapids back to z-.1, then moves to x.5, x.75, x1.0, and x1.25 and drill to z-.4 at each place.)

n15 x0 y.25 r0 (Moves in incremental to x1.25, y.25, drills one hole to z-.4, and Z returns to z-.1)

n20 x-.25 y0 L4 (Moves in incremental in the negative direction to x1.0, x.75, x.5, & x.25 and drills to z-.4 at each place.)

n25 x0 y.25 (Moves in incremental to x.25, y.50, drills one hole to z-.4, and Z returns to z-.1)

n30 x.25 y0 L4 (Moves in incremental in the positive direction to x.5, x.75, x1.0, & x1.25 and drills to z-.4 at each place.)

n35 g80g90 g00 x0 y0 z0 (Cancels canned cycle and moves in absolute to x0,y0,and z0,)

n40 m2 (End of program)

%

If you want the program to start drilling at x0 and then move over by ¼" and drill (4) more holes ending at x1.0, then you will need to change the program as follows:

Change **n10 to (g91 g81 x0 y0 z-.3 r-.1)** (No "L" on this

line. Now it will drill one hole at x0,y0).

Now insert **n12 (x.25 y0 r0 L4)** (Now it will move to x.25, x.5, x.75, & x1.0 and drill to z-.40 at each place.)

Note: Because the next move on line **n15** is in incremental, the tool will move to Absolute position x1.0, y.250, then line n20 will drill (4) more holes and end at x0, y.25.

The g92 and g92.2 Commands

Can you imagine how hard cnc programming would be if you had to write programs in relation to the work's position on the machine? It's always easier to write code from a convenient point in relation to the part rather than its actual position on the machine. This is usually more of a problem on full-size machining centers that have built in homing cycles than on a Sherline with its convenient "Zero All Axes" command. Many times you may want to produce duplications of the same part in a single run. Without the g92 command each part would require separate programming.

The g92 is a handy command to eliminate these problems. It simply resets the axis included in the same line of code to the number included with the axis designation which is usually zero.

For example:

g92 x0 y0 resets your present position to x0 y0.

The next problem is canceling this command and getting back to the standard g90 coordinate system after that particular section of code has been run. The rules to do this are a little more specific: You must be in exactly the same position when you cancel this command with a g92.2 as you were when you entered it; therefore, this is what a full block of code would look like that might be inserted into some program starting at the g92 command:

```
g92 x0 y0
g00 x0.500
z-0.500
g01 z-0.750 f3
x1.500
y1.000
x0.500
y0
z0
x0 y0
g92.2
```

This leaves you exactly back where you started and back under the g90 command. To eliminate confusion it's also important to note that the EMC position display also resets to whatever is called for when the g92 command is given.

The g98 and g99 Commands

These commands are needed when you are **under the g91 command (incremental) to control the retracted z-axis position when drilling a series of holes using the L command.** It's also another method of bringing confusion into your life.

The g99 Command

G99 avoids retracting a drill all the way up to the original z-position between moves, thus saving valuable time on a project that involves hundreds of holes.

Example using a ¼" dia. drill and putting 8 holes ¼ " deep:

```
%
g00 g20 g40 g90 g80 z0.50
g91 g81 g99 x0 y0 z-0.30 r-0.45 f4
x1.5 y0 r0
x1 y0 L2
x0 y1
x-1 y0 L3
g98
g00 g90 g80 z0.50
x0 y0
m2
%
```

This program looks pretty much the same as the program that uses g98, however, there's a significant difference. Note the added r0 in line 4. The reason it's needed is to cancel out the r-0.45 in line 3. Remember, when we implemented the g99 command the z-axis was located 0.45" above the position where the z-axis is now located. If you didn't enter the r0 the z-axis would rapid down one more time for 0.45" and break the drill. The x-1 in line 7 is to over-ride the x1 in line 5 that will control the direction on the x-axis the holes are drilled in. Enter this program into EMC and run it without the drivers turned on to watch the results, and you'll realize why a complete understanding of these new commands is necessary to avoid a lot of tool breakage and ruined parts.

Last, but not least, I'll write the program that we have just completed in the code that we knew before learning about canned cycles.

```
%
g00 g20 g40 g90 z0.50
x0 y0
z0.050
g01 z-0.250 f3
g00 z0.050
x1.500
g01 z-0.250
g00 z0.050
x2.500
g01 z-0.250
g00 z0.050
x3.500
g01 z-0.250
g00 z0.050
y1
g01 z-0.250
g00 z0.050
```

```
x2.500
g01 z-0.250
g00 z0.050
x1.500
g01 z-0.250
g00 z0.050
x.500
g01 z-0.250
g00 z0.050
z0.500
x0 y0
m2
%
```

Now consider how much programming labor these canned cycles can save once you learn them. Twenty-eight lines of code was reduced to nine lines, however, I spent far more time just learning why you have to put the r0 in programs using the g99 command than I did writing the above program, especially when you can just use the [Copy] and [Paste] commands.

This concludes your basic course on CNC g-code programming. You'll know if you passed this course as soon as you attempt to write your own code to make a particular part that you need. You now have the knowledge to produce a very complex part. The next stage of codes to learn will only simplify the writing of the code you now know how to write. I believe that working before you move on you should build up your confidence by producing some parts that you can use before traveling on.

Conclusion

You're ready to take off on your own

The next stop on the EMC highway is to visit Linux-CNC. The road to this town requires a hookup to the internet to travel on. If you're not connected to the internet you're missing one of the great advancements of civilization. It's like having all the great libraries of the world on your personal bookshelf. I have taught you the basic language of g-code programming, and much more is available on the http://www.linuxCNC.org/ site. A couple of members from this group have written some programs for simplifying code-writing that I'm sure you'll find interesting. They have a users group that can be of great help in solving code writing problems.

If you're like me you will find a lot of satisfaction with this new ability to write a program and then watch your faithful robot make parts for you at the push of a button. Though the parts you make initially on this machine may be small, what you have learned is not. When you master this, you will have gained the ability to make parts in shapes and in numbers that cannot be matched by the world's best machinists working by hand. If you want, you can apply what you have learned to larger machines and larger parts later on. Whether you took on this challenge just for the fun of it or as a way to solve an immediate practical need for small precision parts, you are on your way to joining the leading edge of today's modern machinists.

Again, you should remember that Sherline hasn't charged you a penny for the custom LinuxCNC program even though we have spent many thousands of dollars designing a system and writing instructions that an average hobbyist can understand. In turn we expect you the customer to make an honest effort to find the answers to your questions in these instructions before calling Sherline on their 800 number. The only way that Sherline makes a profit is by selling products it makes in the USA throughout a very competitive world. We don't make a profit helping you find careless typing errors in your programs.

If you found my instructions useful and you are not reading them to operate a Sherline machine, may I suggest a way to show your appreciation and help a good cause would be to donate a few dollars to my foundation that is dedicated to getting great craftsmen the respect they deserve in this world. Check out www.craftsmanshipmuseum.com/jmfound.htm, and be sure to visit the Joe Martin Foundation's Internet Craftsmanship Museum at www.craftsmanshipmuseum.com/ where the marvelous work of these great craftsmen can be viewed and enjoyed 24 hours a day.

Using the Optional 4th Axis

The Sherline driver board was designed to accommodate the fourth (A) axis; therefore, all that is needed is P/N 8730 rotary table that includes both the proper stepper motor and the stepper motor mount. The A-axis is programmed in degrees. A minus entry will rotate the table in a counter-clockwise direction. The A-axis can move in unison with the X-, Y- and Z-axes. Use slow feed rates to avoid a situation where the A-axis cannot keep up because of the 72 to 1 worm ratio of the rotary table. There isn't any EMC software available at this time to write the complex code associated with engraving or machining on a round surface, however, I believe the type of parts that will be made by Sherline machinist are straightforward and can be easily programmed. Engrave parts such as jewelry will need special programs.

FIGURE 23—The Sherline 8730 CNC Rotary Table with stepper motor can be plugged directly into the A-Axis cord pre-wired into your computer to operate as a 4th axis.

Definitions and Codes

The word definitions listed below were copied from the Linux CNC website. It's where I gathered the information

to write part two of these instructions. It's worth reading and using to go to the next step in programming, but for the time being stick with just my instructions. I never intended to write a complete programming manual and was only interested in writing enough instructions to get you up and running in the marvelous world that we call CNC.

http://linuxcnc.org/docs/html/gcode/g-code.html

I have highlighted in **red bold face** the main words we'll be working with when using a Sherline machine, and, although many of these words we will not directly use, I wanted you to be aware of the codes used in industry. A CNC program "word" is defined as an acceptable letter followed by a real value.

Table 1 —Words acceptable to the EMC interpreter

D	Tool radius compensation number
F	Feed Rate
G	General function (see below)
H	Tool length offset
I	X-axis offset for arcs X offset in G87 canned cycle
J	Y-axis offset for arcs and Y offset in G87 canned cycle
K	Z-axis offset for arcs and Z offset in G87 canned cycle
L	Number of repetitions in canned cycles and key used with G10
M	Miscellaneous function (see below)
N	Line Number
O	Subprograms
P	Dwell time with G4 and canned cycles, key used with G10
Q	Feed increment in G83 canned cycle
R	Arc Radius, canned cycle plane
S	Spindle speed
T	Tool selection
X	X-axis of machine
Y	Y-axis of machine
Z	Z-axis of machine

Miscellaneous words

M words are used to control many of the I/O functions of a machine. M words can start the spindle and turn on mist or flood coolant. M words also signal the end of a program or a stop within a program.

Table 2—M Word List

M0	**program stop**
M1	optional program stop
M2	**program end**
M3	turn spindle clockwise
M4	turn spindle counterclockwise
M5	stop spindle turning
M6	tool change
M7	mist coolant on
M8	flood coolant on
M9	mist and flood coolant off
M26	enable automatic b-axis clamping
M27	disable automatic b-axis clamping
M30	program end, pallet shuttle, and reset
M48	enable speed and feed overrides
M49	disable speed and feed overrides
M60	pallet shuttle and program stop

Preparatory Words

Some g words alter the state of the machine so that it changes from cutting straight lines to cutting arcs. Other g words cause the interpretation of numbers as millimeters rather than inches. While still others set or remove tool length or diameter offsets. Most of the g words tend to be related to motion or sets of motions. Table 3 lists the currently available g words.

Table 3—g-code List

g0	**rapid positioning**
g1	**linear interpolation**
g2	**circular/helical interpolation (clockwise)**
g3	**circular/helical interpolation (counterclockwise)**
g4	dwell
g10	coordinate system origin setting
g17	**xy plane selection**

g18	**xz plane selection**
g19	**yz plane selection**
g20	**inch system selection**
g21	**millimeter system selection**
g40	**cancel cutter diameter compensation**
g41	**start cutter diameter comp. left**
g42	**start cutter diameter comp. right**
g43	**tool length offset (plus)**
g49	**cancel tool length offset**
g53	motion in machine coordinate system
g54	use preset work coordinate system 1
g55	use preset work coordinate system 2
g56	use preset work coordinate system 3
g57	use preset work coordinate system 4
g58	use preset work coordinate system 5
g59	use preset work coordinate system 6
g59	1 use preset work coordinate system 7
g59	2 use preset work coordinate system 8
g59	3 use preset work coordinate system 9
g80	cancel motion mode (includes canned)
g81	drilling canned cycle
g82	drilling with dwell canned cycle
g83	chip-breaking drilling canned cycle
g84	right hand tapping canned cycle
g85	boring, no dwell, feed out canned cycle
g86	boring, spindle stop, rapid out canned
g87	back boring canned cycle
g88	boring, spindle stop, manual out canned
g89	boring, dwell, feed out canned cycle
g90	**absolute distance mode**
g91	**incremental distance mode**
g92	**offset coordinate systems**
g92	**2 cancel offset coordinate systems**
g93	inverse time feed mode
g94	feed per minute mode
g98	initial level return in canned cycles

Note: a complete list of g-codes and their explanations can be found on the Sherline Ubuntu CNC computer. In the main menu go to ***Applications>CNC>g-code Quick Reference.***

Table 4—G- and M-Code Modal Groups

Group 1 = {g0, g1, g2, g3, g80, g81, g82, g83, g84, g85, g86, g87, g88, g89} —motion
Group 2 = {g17, g18, g19} — plane selection
Group 3 = {g90, g91} - distance mode
Group 5 = {g93, g94} - spindle speed mode
Group 6 = {g20, g21} - units
Group 7 = {g40, g41, g42} - cutter diameter compensation
Group 8 = {g43, g49} - tool length offset
Group 10 = {g98, g99} - return mode in canned cycles
Group12 = {g54, g55, g56, g57, g58, g59, g59.1, g59.2, g59.3} coordinate system selection
Group 2 = {m26, m27} - axis clamping
Group 4 = {m0, m1, m2, m30, m60} - stopping
Group 6 = {m6} - tool change
Group 7 = {m3, m4, m5} - spindle turning
Group 8 = {m7, m8, m9} - coolant
Group 9 = {m48, m49} - feed and speed override bypass

There is some question about the reasons why some codes are included in the modal group that surrounds them. But most of the modal groupings make sense in that only one state can be active at a time. *Note: It isn't necessary to try to remember the words I highlighted. You'll learn these as we go.*

Handy Tip: Milling threads without a cnc rotary table. You can mill threads using a rotary table to turn the part; however if you don't have a 4th axis, this program would do the job. It uses helical interpolation and a single-pointed threading tool to cut six threads at 10 TPI. The reason I switched to incremental was I could use the copy and paste edit program for each revolution of the thread.

```
%
g90 g40 g49 g00 x1 y0 z0
x0
g91g02 x0 y0 z-0.100 i0 j0.5 f6
g91g02 x0 y0 z-0.100 i0 j0.5 f6
g91g02 x0 y0 z-0.100 i0 j0.5 f6
g91g02 x0 y0 z-0.100 i0 j0.5 f6
g91g02 x0 y0 z-0.100 i0 j0.5 f6
g91g02 x0 y0 z-0.100 i0 j0.5 f6
g90 g00 x0.5
z0
x1 y0 z0
m2
%
```

NOTE: *I have purposely started using lower case letters to designate cnc because I would like to encourage its use as simply a word rather than an abbreviation for Computer Numeric Control. This is the future of machining—simply the way things will be done—and the designation cnc will become so common eventually as to be just the word people use to describe having a machine driven by a computer make a part for you. –Joe Martin*

Troubleshooting

In order to keep this section as up to date as possible, rather than include it in the instructions we now keep one centralized copy of the troubleshooting guide posted on our website at www.sherline.com/troubleshooting-cnc/. This section explains what to do if you experience problems with the operation of your stepper motors, driver board or computer. The most common problem people have with a new system is forgetting to plug the parallel cable that comes out of the bottom of the computer into the parallel port higher up on the back of the computer. This is how the computer communicates with the driver box, and if it is not connected, the stepper motors will not run. All other problems we are familiar with are discussed at the link above.

Frequently Asked Questions

The "Instructions and Utilities" CD that came with your computer or driver box contains a "Frequently Asked Questions" file called *CNCfaq.pdf*. An up-to-date version is also available on the Sherline website at www.sherline.com/cnc-machining-faqs/. This page will answer many of the questions a new user might have about Linux, LinuxCNC, g-code, and other aspects of CNC. General machining questions are addressed at www.sherline.com/standard-faq/.

What if I Have a Question that Wasn't Answered Here?

If you need more information about prices, accessories or availability, customers in the USA or Canada can call our toll free number: **1-800-541-0735** M-F, 7:30-5:00 PM (Pacific). If you have other Sherline-related questions and we can't answer them, we'll do our best to point you in the right direction. Outside the USA or Canada, call 1-760-727-5857 or fax 1-760-727-7857. CNC-specific questions from system owners should be directed to the Linux group at www.linuxcnc.org. The best source for information on Sherline tools, 24 hours a day, is always our website at www.sherline.com. Sherline tools and accessories can be purchased factory direct, 24 hours a day, at www.sherline.com, or you can see our Sherline authorized dealer list for a dealer nearest you.

Sherline Products Inc.
3235 Executive Ridge • Vista • CA 92081-8527
(760) 727-5857
Email: sherline@sherline.com
website: www.sherline.com