DAK's TurboCNC v3.1a Documentation July 28, 2002

This file is best viewed in a fixed width font.

Upgrading? See changes.txt for the latest in this version.

---
0. Legal notice:

In summary form:

1. It's your own damn fault.
2. I don't have any money anyway.
3. Piracy is stealing.
4. And of course, you agree.

WARNINGS appear throughout this manual. Do not take these to be applicable in all situations, nor to completely describe the hazards involved. CNC machinery supplies the power to do your work more effectively. You supply the judgment. Although I do my best to stand behind this product, I won't be liable for damages incurred.

---
1. Overview:

This program allows a regular PC to act as a CNC controller using the parallel ports and EIA standard Gcode (RS-274). I originally wrote it to control my converted 7x10 mini-lathe since CNC software was prohibitively expensive at the time (1997). Since then I have added features to make it more flexible and easy to use at the request of the growing number of users. It is my hope that other CNC hobbyists and professionals find it useful in their endeavors.

TurboCNC is open-source shareware. With a $20 registration fee, the source code will be emailed to you and you can do anything you want with it except redistribution, building an entire CNC empire if you like. This also gets you on the "preferred priority" list for support issues, rather than the more easy-going "when I get around to it" stack. This program has no cripples, as I consider that to be unethical.

Send payment to admin@dakeng.com through PayPal (www.paypal.com), or put a check in the mail to the address in section 13. Paypal allows you to pay conveniently with a credit card through a secure escrow service.

If you need more assistance with anything feel free to email me. Contact info is in the last section. It is assumed that one is familiar with the basic principles of CNC operation.

The program runs in DOS. This means a full-blown DOS box, or at least a restart of Windows in DOS mode (Usually F8 on startup will get you there, or pick Shutdown --> Restart in MS-DOS mode). Depending on your system you might be able to get away with multi-tasking it, but I doubt it'll be satisfactory.

---
2. Specifications:

Runs in DOS and uses the parallel ports on a PC
Supports up to 8 axes of motion simultaneously
28 kHz step rate on a 486-66 PC (single axis move)
Supports linear/circular/helical interpolation, and single-axis threading
Provides step/dir or phase drive outputs (compatible with Drigotti and MaxNC boards)
External trip, limit, and home switch inputs
Support for digitizer probes and encoder input for jogging
Outputs to control coolant pumps, spindles and PLC's
Feed overrides on the fly
Turning, slab milling, and peck drilling cycles
20 tool offsets, incremental/absolute inch or metric modes, and subroutines
Uses canonical EIA-274D G code, works with CAM programs such as DeskEngrave, VectorCAM and others
Reads one line at a time from disk for unlimited file sizes
Old school programming, not "bloat ware". Executable is a mere 200k.

Limitations:

Versions 3.0g and above require an 80386 or better with math coprocessor
M60 and M62 subroutines can be 20 layers deep max.
179 character max line length.
16 phase patterns max per direct step axis.
Accuracy suffers if the distance per step is smaller than about 10^-10 units.
No single move can require more than 2 billion steps on any axis.

Rapid moves can be no faster than 2000 ipm or 30 deg/sec.

Known bugs:

- If the part takes longer than 24 hours to machine, the displayed execution time will likely be incorrect. Workaround: Keep track of what day it is by leaving the shop once in a while for coffee.

- Some computers don't like to run this program. Look at the troubleshooting section first, but if all else fails, try another box. When I can get my hands on such a computer, I'll be able to figure out what the problem is. Typical symptoms seem to be hiccup-y or no motion from your machine, or skipped program lines.

---

3. Setting up your machine

TurboCNC outputs step/direction or phase codes to drive stepper motors from the parallel ports. When you run the program for the first time, go to option (4) Machine Setup. The onscreen prompts will allow you to set up your machine configuration. You can press F1 at most screens for a mini-help function.

Set the number of axes first. Common configurations are 2 axes for a lathe, 3 axes for a mill. If you have extra gadgets, such as a rotary table, power quill or tailstock, robot, or what-have you, then by all means include these in the axis count. Max number of axes in this version is 8, in any mix of angular and linear.

Then under the axis setup option, set the parameters for the parallel port configuration. Any letter may be used as an axis designator except for the following (M,F,G,I,J,K,#,N,T,R,S,D,H,L,Q). Standard designations for CNC equipment are X,Y,Z,U,V,W for linear axes; A,B,C for angular.

On a mill, you should set your axes up in the order X Y Z. On a lathe, set them in the order Z X. This so that the cursor keys under the jogging menu work intuitively.

Step increment is the distance in INCHES or DEGREES that the axis will move with each step. If Metric mode is set, the step distance will be in mm.

If you need help figuring the step increment for a linear axis, input COMP at the prompt for step increment. You will be asked for steps/rev of the motor, screw TPI, micro stepping ratio, and reduction ratio (if present, use 1 for direct drive). TPI stands for threads per inch of course. This is English units only at present, and only for a linear axis.

Select a port to control each axis. All the step/direction or winding outputs for one axis must be on the same physical port. LPT1 is usually $0378, although

configurations vary. TurboCNC reports the installed ports and their addresses at startup. It's a pretty safe bet that on a machine with only one printer port, the address will be $0378.

When entering the winding code information (option 6 under axis setup if option 5 is set to direct windings), enter it so that the axis is traveling in the positive direction as you go down the list. Up to 12 patterns can be input from the setup screen, 16 if you edit the ini file directly (see section 12 for details). A sample winding sequence:

1. 1000XXXXXXXX
2. 0100XXXXXXXX
3. 0010XXXXXXXX
4. 0001XXXXXXXX

That would be for a 4 phase unipolar motor running full step on pins 2-5 of the selected port. At each pattern, a 1 will set a pin HIGH on the port, 0 will pull it LOW, and an X will not affect the state. With care, H-bridge and other custom home-brew concoctions may be driven by this scheme directly as well. The pin order corresponds as follows (corresponding to the first pattern in the example above):

Pin state: 1 0 0 0 X X X X X X X X
Pin ID: 2 3 4 5 6 7 8 9 1 14 16 17

Pins 1,14,16,17 are driven separately from the other 8 since the computer can only change one byte at a time on the port. Therefore, keep all the phases for one motor either on pins 2-9 or on pins 1,14,16 and 17. Step and direction drives can use the pins in any combination.

Motion parameters set the performance envelope for each motor. Start speed is the fastest step rate that can be applied to your motors instantly from a dead stop without locking the rotor. Max speed is the maximum step rate, and acceleration is in Hertz gained or lost per second. For a typical NEMA 34 double-stack full step stepper good initial values are:

Start: 500 Hz
Max: 4000 Hz
Accel: 2500 Hz/s

Keep them all as high as you can for good performance. That's important!! The Max speed will never be exceeded, ever. The Start speed also decides the speed the motor will decelerate to at the end of a move.

Backlash comp uses a non-accumulating algorithm and is applied at the beginning of a move, not the end. Here's an example commanding moves from position 1 to 2 and back and what's happening exactly on the machine.

;Axis X starts at 0

```
G00 X1  ;Preload
G00 X2  ;Start at 1 and move to 2
G00 X1  ;Back to 1

Preload:    ---| * |---
by move to 1: ---|*  |---
Move to 2:     ---|*  |---
Backlash comp:       ---|  *|---
Back to 1:      ---|  *|---
```

Backlash is taken up at inflection points of arcs and circles as well, and on angular axes, if selected. Units of backlash are inches or degrees as appropriate, and the rate of compensation is the start speed rate for the axis being compensated.

An option is available on the Manual Machining menu to turn on an indicator for the backlash preload. An up arrow means the last move was toward the positive direction, down arrow means the last move was toward the negative direction. A small infinity symbol will be displayed for an axis that does not have backlash set, or has not moved since power-up (indeterminate preload).

The pulse width parameter changes the duration of the step pulse on step/dir controlled axes, as some drives need a few microseconds to recognize that the step line has changed state. The parameter is set directly in 1 uS increments*. Setting this too high costs you speed, and setting it too low can make the drive response erratic. Some equipment combinations are fairly tolerant in this respect, whereas others are not. On my 486-66's, using an IB-104 or G201 commercial stepper driver, I can get by with leaving the pulse width at 0. Using the same computer with the U-2 unipolar driver (see www.dakeng.com) requires a pulse width of 30 uS. I'd go above 20 uS with trepidation here since a high pulse width can cause delayed pulses on other axes that need to move while that line is active.

* (Actually, this is a multiple of the motherboard timer, which outputs a 1.11931799704 MHz clock tick that is the basis for the step timing in TurboCNC. It's close, but not exactly equal to one microsecond.)

Optoelectronic buffers can take a lot of time to register a step pulse. For these, I would input number in the rang of 50 to 150 uS if you have trouble with erratic stepping. Using Gecko 201/210's, you can usually get away with 0 for pulse width, but something like 5 uS is a bit safer.

The usual setting for a step/dir configuration on the step pin is ACTIVE LOW (pulled low for a few uS each step). Very rarely will the configuration be the opposite, although I have it on good authority that drives based on the UNC5804B IC's require an active high pulse, as well as the popular Gecko 320 and 340 servo drives.

The Misc I/O functions are optional. If you enable one of them, you have to specify the port and pin that services that function. This is mainly for configuring relays and such for coolant and spindle, plasma cutting torches, filament heaters for foam cutters, digitizer probes, PLC inputs, etc.... Motor enable lines (required by the Stepper World SP3 drives and others) are also available here. The panic input stops motion instantly, as you would expect.

WARNING In some localities, you are legally required to have a hardware based panic stop switch. This switch should stop the machining action cold, by cutting power or through other means, but not through software. In such an arrangement, you may wire the panic stop input to your panic switchgear so that the computer is told to stop when the machine does.

This is all positive logic on the input side, so ground a pin to make it LOW; HIGH is 5V. I usually just ground the inputs since they tend to float high by themselves on most cards (but not all - confirm this if you use this method!). If you're not sure what runs where, run FKEYBIT.EXE, which is included. Be sure to use this version only, as it has been re-compiled to read the port the same way that the CNC program does. Note that Fkeybit is limited to driving 8 outputs (pins 2-9) and reading 8 inputs (pins 10-17) for now.

Homing switches are now available, thanks to the programming efforts of Nigel Christianson. The setup is self-explanatory for the most part, following the same basic procedure as the misc I/O functions. An axis that is at its home position (e.g., switch is closed) displays an asterisk (*) next to the axis designator in the STATUS window.

A quadrature encoder wheel may now be used for jogging input on the manual machining screen. Set up the channel A and B inputs here. The wheel is only read in the jogging mode, and only when the jog increment is .010" or less. For safety, the input is not buffered, so a good spin of the wheel will only move the machine as long as the wheel is turning.

A digitizer probe input is now available as well. Set the active state to correspond to the "struck" condition. When the machine is in motion and tracking the probe input, the transition from inactive to active is taken as an indication of the part boundary.

For now, threading only accepts a spindle index pulse once per rev. Experimentation shows that this is quite accurate enough in practical applications. The spindle index input should be "clean" (no ringing or false transitions) and active about 10 degrees or so per revolution for best results. A placeholder input has been set aside for a finer resolution encoder on the spindle in addition to the index, but in this revision the program does not use it.

WARNING There is no "babysitter" code that checks that you haven't assigned the same pin to different functions, or named two axes with the same letter, etc... This allows some fairly creative things to be done, but do be careful.

If the changes are ones that you want to keep, save the ini file with option 8. Reloading the ini file (option 7) will wipe out any changes you've made in favor of the information on disk. Most changes won't take effect until you save the ini file and reset the ports.

If you have multiple machines or configurations, you can start TurboCNC with a command line parameter that specifies the name of the ini file to be used. For example, typing:

turbocnc myfile.ini

from the prompt will cause the program to read myfile instead of turbocnc.ini at startup and use it throughout the session for machine configuration editing. The file can have any name, as long as it has an ini extension and respects DOS's limitation of eight characters in a filename. The path is optional if it's in the same directory as turbocnc.exe. By default, the ini filename is the same as the executable - so if you rename turbocnc.exe to lathe.exe, the program will search for lathe.ini at startup instead of the usual filename.

All the pertinent machine configuration info is in the .ini file. If you really get fouled up, you can delete it entirely and start over.

When you get the machine set up the way you want it (and have saved the ini file), get back to the main menu screen by hitting <ESC> and then press 3, then 1. This gets you to the jogging panel, where you can try moving things around to see how the performances is. Typically it takes a few tries to get the motion
parameters right. In the future there will likely be a "Wizard" to help with set up, but for now you'll have to hop around a bit here.

Metric users: If you set the units to millimeters and save the ini file, the next time you load the program you'll default to metric.

Some "stock" .ini files are included with the download. As of this writing, there is an ini for the MaxNC 10 open loop mill and a basic EMC configuration. Please share your ini files so that other common drive configurations can be included.

Note that with version 3.0g and up the ini file is "man-readable", meaning that you can conveniently inspect and edit the file with any text editor. Information pertaining to the ini file format is in section 11 of this manual.

<REMEMBER, YOU CAN PRESS F1 ON MOST SCREENS FOR HELP>

---

4. File Format, G code flavor, and quirks.

Two formats are available for the Gcode files in TurboCNC. Originally this program used a special header that looked like this:

```
=========================
{dakcnc3.0}

General setup notes can go here.

{program start}
G90   ;or first line of the program
...
M02 ;must be used as the last program line
=========================
```

See the included file SAMPLE.CNC for a demonstration of the old file format. The program now accepts standard ASCII files for input however, so the special headers are no longer necessary. Take a look at the included .CNC files for some basic examples.

Typically I use Notepad or MS-DOS EDIT to write my programs using the shell command from TurboCNC. With some minor tweaking, output from ACE converter, VectorCAM, MasterCAM and DeskEngrave is known to work just fine.

WARNING Please familiarize yourself with the G code style used by TurboCNC. It is not unusual, but has a few quirks you will probably want to be aware of. This will help prevent surprises and scrapped work.

NOTES:

- Use only ONE G or M word per line. This is important. For example:

```
G17 G00 X0Y0Z0 ;No good, G17 is not read
```

- N codes are not necessary except when using subroutines. When using subs, put a unique line number (e.g., N010 and N012) on the calling line AND the target line.

- Jumping to or from a sub loses the modal Gcode, but not the feeds or anything else.

- G codes and feeds (F words) are modal, with the exception of G92.

- Circular and helical interpolations are called using either I and J, or R notation, for any combination of axes (see primer in section 8). The plane selector (G16-19) should be called beforehand. I and J are always incremental.

- Set off comments with a semicolon or parenthesis, e.g.:

```
N010 G00 X0.000   ;comment here
```

G92 X1 Y0 (or comment here)

Putting a semicolon at the beginning of a line negates the line. The comment must be the last thing on a program line.

- All moves involving more than one axis are interpolated, even in rapid. On some mill controls, the Z axis will always retract first before moving X and Y. Not so here!

- Sometimes a little "rounding error" will be present at the end of a helical or circular interpolation. If you are off in calculating I and J for the move, the problem is exacerbated. (Note, in rev 3.00d this was pretty bad, it's better now! If you have a program that develops a motion error of this type, please let me know.)

- Angular axes are always driven in degrees modulo 360. For example, if a table is at 10deg and you command it to 350deg, it will go around "the long way". If you commanded it to -10deg instead, it will zip over to -10deg "the short way" and then report the current position as 350deg. Commanding a +720deg incremental move will index the table around two full revs, but the coordinate will remain unchanged since it is always set to somewhere between 0 and 360 only.

- If you use IPR feeds through a G95 call, use the S word to set the spindle speed in RPM from which the feedrate will be calculated first. The code to control the spindle from the computer is "empty" - this is so the users can add it themselves.

- Metric distances, feeds, etc, are in mm and mm/min or mm/rev as appropriate.

- Dwells are programmed in integer seconds, from one second (G04 #1) on up to a max of 9 hours (32767 seconds). You can get out of a dwell by hitting <ESC> or <ABORT>.

Exception: The dwells for G82, G78, and G83 are in milliseconds.

- The feedrate in an interpolated move is based on the actual distance covered by each of the linear axes involved. If there are none, then the first angular axis on the list is assigned the feedrate (degrees/sec).

- If you hit ANY limit switch, the machine won't move again until you either disable it, or use Jog mode to back away from the switch. These are for preventing "going off the rails" only - use the home switches for calibrating the machine.

- When block delete is active, a line with a "/" as the first character is skipped. Any word in a line that is preceded by a "/" is not executed. For example:

G01 X1.234 F5 /F0.5

In block delete mode, the F5 is read as the feedrate and the second F word is ignored. In normal mode however, the F0.5 will supercede F5 on this line, and so the feedrate will be much slower.

---

5. Running parts

Under the manual and auto machining menus there are a variety of options to actually cut metal. After loading a CNC file (1, 1 from the main menu), use Manual Mode (3,1 from the main menu) to set the tool position(s) and get everything set up. Here's how you might use the tool offsets to make a part using multiple tools on a mill:

Put your first tool in the machine, e.g., the one you calibrate the stock position
with. Some machinists I work with use a short starter drill, others a "calibrated" R8 shank with a tapered point on it or an edge finder. Whatever is comfortable for you.

Be sure that "TOOL 0" is showing on the status window (second line from the bottom). Use the R & T keys to change this until it is.

Now move it down to a reference point on your work, say, the corner of the workpiece or vise using the jogging screen. Zero the tool at this position by pressing "0" for each axis, or use the "Z" function to zero them all. Now the master tool is set. You can make a part using only that one tool, in principle. Many people do.

For the second tool, hit "T" to switch to the next tool offset. It should now show "TOOL 1" in the status window. You are now in a new coordinate system for tool #1. Jog around, slap your next tool in the spindle and zero it to the same corner of the vise as the first. The coordinate offsets of this tool are automatically stored in relation to the master tool - ergo, the "zero" for the master is not lost. Play with the R and T keys in jog mode and move around to convince yourself of this.

Now, say you had a starter drill for the master and a regular 3/8" drill for the #1 tool in this example. Here's how you'd use that in pseudo-code:

```
...
M06 T0
... move to position and drill starter hole
...
... retract
M06 T1 ; Pause to allow human to change tool
... move to same position and drill thru.
```

... etc

This is powerful, because you don't have to worry that one tool is 3" longer than the other when you write the program.

Once you get all the tools set up, I recommend that you save the offsets using the option under the file menu since it takes a little time at first to get them all in there. Typically, on a mill you'll only be adjusting the Z coordinate from tool to tool, but on a lathe you need to worry about both Z and X. Hence, the tool offsets work in every axis for flexibility. These are similar to fixture offsets on some commercial CNC controls.

NOTE: Never put the tool word on a line by itself, as the modal Gcode is still active from the previous line when the offset changes. Use M00 T1, or M06 T1, or something similar for safety.

You can also manually run G/M codes from the MDI screen. The input is shadowed to a file MDI.CNC (or other of your choosing) with a timestamp for your convenience. All the codes are supported in MDI except subroutines. You can use CRSRUP and CRSRDN or PGUP/PGDN to use any of the last 20 lines input as a basis for the new line about to be entered. This is static between MDI sessions.

Now move over to the Automatic Machining option from the main menu.

One-step is for when you're testing out a new program for "sticking points". You can exit at any time using <ESC>, and keep hitting <ENTER> or <SPACE> to execute the lines one at a time.

Dry lock verify runs through the file without moving your machine or turning anything on. This is to see if there are errors in the file syntax, and also to obtain an estimated running time for the program. Expect the estimated time to be just a shade on the low side.

Operator moderated production is the usual setting for chucking operations where the operating sequence is one of "load, run, stop, unload, load" by a human operator. A running count of cycles executed is kept, and can be set at any number to keep track of how many parts you've made, up to 2 billion. Why so high? Well for starters, you can set the number to some arbitrary value to begin with, and use it as the serial number for each part as you make it.

Automatic production runs the CNC file continuously for some number of cycles that you specify. This is great if you've got pallet changing, bar pulling, PLCs, or other robotic hardware to do the loading and unloading for you. The G codes to run the robots as other axes must be included in the same program with the code to machine the part. This is the reason why I've included so many axes of motion.

Panic stop and limit switches are wired to the parallel ports as logical inputs.

This is to get around the keyboard buffer in case of an emergency, and also allows for fast polling of the input states. You have the option, after a panic event, of continuing where you left off or aborting completely. The former option is good for fixing simple things, like a incompletely tightened tool or something that you noticed just before the "rubber met the road", so to speak.

You can also use <ESC> as a panic stop key, and activate feedrate overrides on the fly using the greater than/less than (<,>) symbols. Use the shift key and < > to get a finer degree of override control.

Note that a minijog function is available while in a stop mode (M00/M01/M06/PANIC). Use the arrow keys for axis one and two, + and - keys for axis three. The keyboard layout assumes a three axis mill set up as XYZ. For lathes, set up Z as the first axis, and X as the second. The XYZ keys zero axes as appropriate, N & D enable and disable the drives (if configured), A & B toggle the coolants, and the # key zeroes all of the axes.

The status window at the left of the screen shows the axis positions, status of coolant and spindle, the current tool and motion modes, feed override, feedrate, modal Gcode, and so on. Generally, anything important is displayed in white, whereas unimportant or default stuff goes to brown to keep the information manageable. The position of each axis is updated at the end of a move, and "In motion..." will be displayed while things are still moving. In previous versions you were allowed to have the position updated each step, but I found this costs too much CPU time to keep up with.

---
6. How it works.

At the software level, motion control can be fairly simple yet tantalizingly complex if one is to do it well. I invite you to look at the source code if you're interested in nuts and bolts (registered folks only). It is written in Turbo Pascal 7 and 80386/FPU assembly, and encompasses some 18,000 lines of code these days.

What you'll see is essentially 80% menus, help screens, and handshaking code, and about 20% actual hard-core motion control.

Moving a single axis is easy. Once it can be established how many steps you're going and how quickly you want to take each one, then take your first step. Take the next step a little faster, and the third faster yet according to an acceleration algorithm that allows a linearly increasing velocity with time (the trapezoidal move). While you're at it, keep track of how many steps it'll take to get back down to the initial speed, and start slowing down when there are exactly that many steps left. Simple right?

The big boondoggle with performing even that simple trick was getting a precision timer set up to resolve down to microseconds without a lot of

overhead. As does all the competing software, TurboCNC uses the system motherboard timer's 1.1193180 MHz timer tick to pace out the step pulses. I won't name any names, but I've found TurboCNC to output smoother pulse trains than some of the competing programs out there - primarily because this motion algorithm has been heavily optimized in assembly.

Linear interpolation is just several of the single axis moves stacked together, with the accelerations calculated such that each axis stops and starts in sync with the others. This implementation is quite general, and can support asynchronous moves on any number of axes at the same time. It would be possible, if fed the right data, to simultaneously cut arcs in two different planes at different feedrates with this motion control unit (MCU), and have each end at a different time - say on a four axis lathe for example. Each axis has a target velocity, and threading works by updating this target velocity for each affected axis when the spindle index pulse is read. This is quite general too, and if the Gcode language could support it, multiple axis threading and non-threading moves could be intermixed.

For now, my circular and helical interpolation algorithm is basically a cop-out from doing any "real" programming. A imaginary point is swept through space by a constantly updated vector. Whenever the tool position is more than one increment away from that point in any axis, it is moved closer by one increment (or step). The resolution of the sweep is based on the resolution of the axes involved so that little computational overhead is invoked. Soon, the arcs will be a series of short line segments fed into a constant velocity engine with a "sag" parameter for defining the accuracy instead.

The interpreter is simple also. It parses a line into "chunks" based on the spaces it finds in the line, and then checks the first byte of each chunk. If it's a recognized letter, then the rest of the chunk should be a number and is assigned to a variable accordingly. A procedure is called based on the value of the G or M variable. In rev 3.00f and up, I've added some preprocessing steps to accept output from CAM programs that scrunch all the letters together or introduce odd spacing by adjusting the line format before it hits the parser.

---
7. Future improvements.

No project of mine is ever done! Here is a list of some improvements I'd like to make in rev 4 of this program. Many of them are user suggestions.

- Mouse pull-down menus (especially for files)
- Integrated code editor
- Conversational machining / G code builder
- Toolpath visualization
- Radius compensation
- Improved hyperlink based help system
- Parabolic and function based f(x) curve generator

- Parameters and looping codes
- Multiple input file flavors (TurboCNC, Fanuc, CNCpro, HPGL, etc..)
- Letter engraving macros or font reading capability
- Anti-crash watchdog
- PET "PLC Emulator Thingy"
- Overtemperature trip
- "Top Five" most recent files/macro commands
- Look-ahead ramping
- More than three lines visible?
- Limit switch +/- inputs rather than "bang, you're dead" inputs

One thing I'd like to do is step away from building a Gcode interpreter based on the usual logic (if x then...) and build an AI inference engine to handle the file input. This would be "trained" against a wide collection of file formats and have some scoring function to develop a reasonable interpretation of the input, even if the specific file format has never been seen before. An interesting idea....

---
8. G code primer

A list of the supported G, M, and additional codes and their meanings:

****
G00 Rapid move
****
Moves to the coordinates specified as fast as possible.
Usage:

G00 X1.2 Y0.3 ;Moves to coordinate (1.2,0.3) at full speed.

****
G01 Interpolated feedrate move
****
Moves to the coordinates specified using the programmed feedrate for interpolation (as modified by the override)
Usage:

G01 X1.2 Y0.3 F3.0 ;Moves to coordinate (1.2,0.3) at 3 units/minute

Here are the rules for feedrates (assuming english units):
- A single linear axis moves in ipm or ipr.
- A single angular axis moves in deg/sec.
- Multiple linear axes move in ipm/ipr according to the square root of the sum of their squares distance.
- Multiple angular axes move in deg/sec for the first axis on the line, all others follow.
- Mixed angular and linear axes follow the rules for linear axes, the

angular axes will follow.
- No axis will ever go faster than the max speed established in setup.

The feed override affects all moves except for rapids (G00).

****
G02 CW circular interpolation
****
  Moves in a CW circular arc with its center set by signed I and J increments, to the destination given.
Usage:

G02 X1.0 Y1.0 I0 J1 F2.0 ;Arc to (1,1) with center at origin

  That will move from the current point to (1,1) in an arc with its center at the current point + 1 unit in the Y direction at 2 units/minute.

  In G16 mode (implicit planes) I is the center point offset for the first axis on the line (X in this case) and J is the center point offset for the second (Y in this case) regardless of the actual axis names.

  If the destination is the same as the origin, a full circle will be described by the tool motion. I and J are always incremental. (PLEASE NOTE: In rev 3.00g and prior, the I and J word were absolute in absolute mode, incremental in incremental mode. This is no longer the case.)

  In G17/18/19 modes, the order of parameters on the program line is not important. Use the I word for offset in the X direction, J for offset in Y, and K for offset in Z to describe the center position from the starting point.

  An alternate syntax is to define the radius of the arc and the endpoints only, as follows:

G02 X1.0 Y1.0 R0.25 ;Radius defined arc to (1,1)

  This heads to (1,1) from the current point with an arc of .25 radius. Using a negative radius chooses the arc greater than 180 degrees that crosses both points, a positive radius takes the arc that is 180 degrees or less. The program will stop with a warning if you have a radius that is impossible.

  Although you can cut a full circle by specifying the endpoint to be the same as the start point, I don't recommend this since rounding of the values can cause the control to cut the infinitesimally short arc instead. Program both halves of a full circle. Eventually I'll add a "full circle" code to get around this.

****
G03 CCW circular interpolation
****

Similar to G02, but counterclockwise.

****
G04 Dwell
****
Pauses execution for an integer number of seconds.
Usage:

G04 #6 ;pauses for 6 seconds

WARNING Don't use G04 and your hands as a tool-changer. It takes longer than you think... Use M00 or M06 instead.

****
G16 Set implicit planes (TurboCNC former standard)
****
The original TurboCNC thru rev 3.00g+ used an implicit plane specification in which the first two axes called out on a G02/03 line became the plane of action for the interpolation. This is the default mode at startup for compatibility with files written for earlier versions. I suggest you re-write them to use the conventional plane selectors if possible. Note that I and J are always incremental now.

****
G17 Set XY plane
****
Sets plane for circular and helical interpolation to X-Y.

****
G18 Set XZ plane
****
Sets plane for circular and helical interpolation to Z-X.

****
G19 Set YZ plane
****
Sets plane for circular and helical interpolation to Z-Y.

The rule for planes and CW/CCW is this (from the RS-274 standard):

Arcs shall be CW or CCW as viewed in the negative direction of the out-of-plane axis in a standard right hand axis system.

Translation: For mill users, if +X is rightward and +Y is toward the back of the machine (That's tool motion relative to the work!) and +Z is up, then as you look down (-Z) at the table in G17 mode, arcs will look CW or CCW as you'd expect.

For lathe users, if +Z is away from the headstock, and +X is increasing

diameter, then in G18 mode CW/CCW will look reversed if you look down at the slide. This is because in this right hand system, -Y is looking up.

****
G20 Inch units
****
Sets distance unit to inches.

****
G21 Metric units
****
Sets distance unit to mm.

NOTE: G20 and G21 are identical to G70 and G71.

****
G28 Home all axes
****
Each axis with a home switch will be driven to the home position and its position reset. This is for recalibrating your machine during long cycle times where temperature creep or position loss is a factor.

****
G31 Probe move
****
Move with continuous check for digitizer probe strike. If the probe hits something, the machines stops and write the position to the file POINTS.DAT in the same directory as the executable.

Use G31 just like G01, e.g.:

G31 Z-4 F10

This will move to Z-4 at 10 ipm, unless the probe input goes active. If so, it'll stop and write the tripped position to points.dat in the current directory. You can use G31 to get off of the part again if you need to, the program only accepts the rising edge as valid input. Note that if you move twice in the same direction with G31, you can smash your probe!

Manual probing: When the probe is "on" in jogging mode, every move is like a G31, except that the points are not written when the input is triggered. Use "F" to write a point anytime. This is mainly for tinkering and testing the probe geometry, although clever folks might figure out how to make a teachable CNC using this feature.

****
G32 Probe cycle
****
Repetitive probing to digitizer scan a 2 or 3D surface

Usage:

G32 Xbound Ybound Zbound Idiscretization Ffeed #reserved

This is 2 or 3-axis general, so the last axis cycles the most and the first axis only once through a number of points determined by the discretization distance. In the future, separate algorithms for this will be supported, so until then don't use the # code. The digitizing scheme is of the "bed of nails" variety, e.g.: straight up and down in the last axis on the line.

WARNING: Rounding error may prevent the last row or column from being scanned. Add a small amount to the bounding distance to avoid this, e.g.:

G32 X1 Y1 Z-1 I0.250 F10 ;X1,Y1 may or may not be reached

G32 X1.001 Y1.001 Z-1 I0.250 F10 ;This is better

The output is written to the a scan file in the same directory as the executable, with a filename of surfscan.dat.

****

G33 Single pass, single axis threading

****

Moves an axis to a destination point, in synchronization with the tool spindle at some lead per revolution.

Usage:

G33 Z-1.25 K0.050 ;20 TPI to Z=-1.25

A spindle index pulse must be enabled and present for this to work. The program will time-out the spindle pulse after 5 seconds of inactivity. K is the lead parameter for any axis that this code is used in conjunction with.

Allow a few revs of "lead in" for the axis to synchronize. You need a lot of acceleration to keep up with the changes in velocity - I use 14000 for the accel parameter and it's still a tad sluggish. If the axis falls behind and can't keep up, it'll stop and warn you.

The synchronization is always from the start point, so for multi-start threads, offset the start position by some fraction of the thread lead. For 30 degree infeed, change the start position on each pass along a 30 deg vector.

****

G53 Change to master coordinates

****

Puts the machine in the master coordinate system. This has the same effect as calling T0.

TurboCNC defaults to the master coordinate system at startup. Once a system of offsets is created in jog/setup mode, it may be saved using the file menu for later retrieval.

****

G50 Probe hole ID

****

Finds the center of a hole using a touch probe.

Put the probe inside the hole approximately in the center, and call G50 from within a program. This is an orthogonal six hit probing algorithm - clock positions struck are 12,6,9,3,12,6 in the XY plane as you normally view the part on a mill. In jog mode, press C to do this. You must have a touch probe installed and the input configured for this to work. This is limited to probing in the XY plane for now.

In jog mode, the two-place roundness TIR and diameter will be reported after you input the probe diameter.

Watch out for keyways and similar things that can throw you off. The reason for making this available within a program is for auto-setting boring operations or fixture locating where some part-to-part variation is expected.

****

G54-G59 Change to coordinate offset 1-6

****

This is an alias for T1, T2, T3, etc... (G54 is T1). The coordinates of each axis are reset to align with the origin point in the new offset. For example, in jog mode:

- Get into T0 (use R and T keys)
- Move to some special point on your machine, and zero all the axes. This is the master origin.
- Press T to go to tool 1
- Jog to some other point, and zero the axes again. This is the 1st offset origin. (G54 or T1)
- If you press R to get to master/machine coordinates (T0 or G53), the coordinates in the STATUS window will change to reflect your position relative to the first point again.

The most common uses for this are tool length adjustment and multiple fixturing on the same table.

****

G70 Inch mode

****

Sets distance unit to inches.

****
G71 Millimeter mode
****
  Sets distance unit to mm.

****
G72 CW helical interpolation
****
  Like G02, but allows a third axis to travel linearly as well.
Usage:

G72 X1.0 Y1.0 Z.125 I0.0 J1 F2.0 ; Helical move

  This line moves from the current point to (1,1) in an arc with its center at +1 unit from the starting point in Y; at 2 units/minute. Meanwhile, Z will move linearly to 0.125.

  The radius form of definition using the R word may be used here as well, just like G02.

****
G73 CCW helical interpolation
****
  Similar to G72, but counterclockwise.

****
G77 Turning/Boring/Milling Cycle
****
  Cuts with multiple passes, infeeding by an amount set by the I word.
Usage:

G77 Z-1.250 X0.250 I0.050 F5.0 ; Diameter turning cycle

  That cuts to an X coordinate of 0.250, back and forth in Z from current position to -1.250, feeding in 0.050 each pass and cutting at 5 units/min. The sign of I is important! If you're turning, I should be +, use a - I for boring (same as change on infeed axis). Infeed is applied to the second axis on the line - so this code can be used for flycutting/slabmilling or cutting deep blind slots on mills or trimming the edges of stock. Here's the actual motion sequence for on the above, assuming the tool started at 0,0:

X0.050 slowly
Z-1.250 slowly
X-0.025 slowly
Z0 rapidly
X0.100 slowly
Z-1.250 slowly
... etc

NOTE: After each pass, the control "backs off" by 1 1/2 times the infeed. So if you're boring, make sure you have enough clearance for the bar.

Also, if the infeed doesn't divide into an even number of passes, don't worry. The program's smart enough take a small finishing pass.

The order of the operands defines the motion. The back-and-forth action will occur from the current position of the first axis on the line to the position specified. The second axis called out will eventually reach the position specified by feeding in increments of I each pass.

****
G78 Peck Drilling Cycle
****
  Feeds to a position on an axis, incrementally in "pecks" with a full rapid retract.
Usage:

G78 Z-2.000 I-0.100 F2.0

  That drills from the current position to Z-2.000 at 2 units/min 0.100 units at a time (retracts to original Z each time) . Sign of I must be the same as the direction of cutting. If you put in a dwell parameter with the # operator, a dwell in milliseconds will be taken at the "bottom of the hole". So:

G78 Z-2.000 I-0.100 F2.0 #100

  does the same as the above, with a tenth of a second dwell at the bottom of the hole. This helps the tool life considerably in some cases. After each peck and retract, the drill will rapid to 10% of the peck increment before the bottom of the hole before feeding in again.

  Note that this code can be used for "peck turning" plastic on a lathe to keep the chips short.

****
G80 Cancel drill cycle
****
  It's good practice, although not strictly necessary, to put this code after a series of the G81, G82, or G83 drill cycles. It clears the canned cycle variables from memory.

Usage:

G80 ;Clears canned cycles

  Some CAM programs automatically generate this code after every series of drilled holes.

****
G81 Drill cycle
****
  This is the canonical RS-274D drill cycle. Drills a hole at a specific XY position, to a depth Z at the current feedrate, and retracts to a release plane "R". If the release plane is further away from the work than the drill at the start, the control will rapid to the release plane first for maximum clearance.

Usage:

G81 X1 Y1 Z-0.75 F2.0 R0.25

  This line will cause the control to perform the following actions:

- Rapid to R plane if Z is less than 0.25 absolute
- Move the table to the XY position (1,1) specified, holding Z at the point it was before.
- Feed the Z axis to -0.75 at 10ipm.
- Rapid the Z axis to 0.25 (the release plane)

  To drill another hole just like the first, just input the XY position on the next line:

G81 X1.5 Y1.25

  This second hole will be done the exact same way was the first, but at the new position of X=1.5 and Y=1.25.

  If the R plane is between the current Z position and the bottom of the hole, the control will rapid to the R plane after moving XY and before drilling. If the R plane is "above" where the drill is in Z at the start of G81, the control will rapid to the R plane first before moving XY. This affords maximum safety without overly compromising speed.

  Note that R is absolute in absolute mode, incremental in incremental mode! All the other parameters behave similarly.

  All of the coordinates (XYZR) need to be called out on the first G81. These will "stay in effect" thereafter until G80 is called. So, if you have a series of holes that are all to the same depth, you can use this:

```
G81 X1 Y1 Z-0.75 F2.0 R0.25  ;First hole
G81 X2 Y2
G81 X3 Y2
X2.5
G80  ;Four holes were drilled at (1,1) (2,2) (3,2) and (2.5,2)
```

  NOTE: These cycles ignore the plane selector - they must be carried out conventionally in XYZ as shown.

****
G82 Drill + Dwell cycle
****
Similar to G81, but requires a # parameter for the dwell at the bottom of the hole in milliseconds.

Usage:

G82 X0 Y0.5 Z-1 F10 R0.25 #250

Drills at coord (0,0.5) to a depth of Z=-1 at 10 ipm. Dwells for a quarter second, and then retracts to Z=0.25.

****
G83 Peck drill cycle
****
Similar to G81/82, but requires an I parameter for the peck increment when drilling.

Usage:

G83 X0 Y0.5 Z-1 F10 R0.25 I-0.100 #250

Same as the G82 example above, but the drill will descend at the 10 ipm feedrate in 0.100 inch pecks with a rapid retract to the original Z. The drill rapids down to 10% of the peck distance above the bottom of the hole before feeding again to minimize "air time".

The dwell parameter (#) is optional with G83. The above example will dwell a quarter second at the end of each peck. The dwell is in integer milliseconds. Expect it to be accurate to about 15% or so, it doesn't use a high precision timer.

****
G90 Absolute coordinates
****
Set coordinates to absolute mode (default).

****
G91 Incremental coordinates
****
Set coordinates to incremental mode (offsets from current position)

****
G92 Preload of registers/Set machine coordinates
****
This code sets the position of any or all axes to a specific value. Use this to reset the position inside a program. No motion will occur.

Usage:

G92 X0 ;Zeroes X axis

G92 X0 Y0 Z0 ;Zeroes all principle axes on a mill

G92 Z1.234 ;Z is now set to 1.234

You must be in the master coordinate system to use this code. All of the other offsets (1-20_ follow the master. Ergo, if the origin in offset 1 is set to be exactly 3" away from the master origin (in G53 mode), then that relationship is maintained as the master origin moves.

Use jog mode to setup the coordinate offsets (tool offsets) and save them through the file menu. This command is not modal in versions 3.1 and up.

****
G94 IPM feedrate
****
Set feedrate unit to units/min

****
G95 IPR feedrate
****
Set feedrate unit to units/rev. Be sure to have set the "S" parameter when calling this function!

G95 S1000 F0.002 ;feed is .002/rev at 1000 rpm
.
.
G01 F0.001 ;Finish pass at .001/rev

****
G97 Program spindle RPM
****
Set spindle RPM using S word.

Usage:

G97 S1000   ;1000 rpm

This function is a "placeholder" in the source code. Registered users can use it as a starting point to program their own spindle control routines into TurboCNC. Note that if you use inches/rev feedrates by calling G95, you must call G97 or otherwise set the S parameter beforehand with the spindle speed.

****
M00 Automatic halt

****

Halts program until operator presses a key.

****

M01 Optional halt

****

Same as M00, but can be disabled by the operator. (Option available under machining menu)

****

M02 End of program

****

Stops execution and cycles. This must be the last line of the program.

****

M03 Spindle on CW (usual for a lathe or mill)

****

Turns on spindle.

****

M04 Spindle on CCW

****

Similar to M03.

****

M05 Spindle off

****

Turns spindle off to coast to a stop.

****

M06 Tool change

****

Essentially the same as M00, but with a prompt to tell the user which tool is being requested. Registered users may program their own, more sophisticated routines for automated tool changes here.

Usage:

M06 T1 ;Prompts for change to tool #1 and changes to tool 1 coord offset.

****

M07 Coolant A on (flood)

****

Switches on relay A.

****

M08 Coolant B on (mist)

****

Switches on relay B.

****
M09 Coolants off
****
Switches off both relay A & B.

****
M13 Spindle CW and coolant A on
****
Turns the spindle on in the CW direction and activate the "A" coolant.

****
M14 Spindle CCW and coolant A on
****
Similar to M14.

****
M17 Enable drives
****
The Stepper World SP3 and some versions of the MAXNC drives require an enable signal in order to operate. After the enable pins have been setup in TurboCNC, use this code to turn them on. The drives will be disabled during a panic abort, or when you exit TurboCNC normally. They will be turned on when the program starts.

****
M18 Disable drives
****
Similar to M18, this turns the enable lines off. By default, they are enabled when TurboCNC starts. There are hotkeys in the jogging mode to switch these on and off as well, mainly so that you can confirm that everything works.

****
M30 End of program & rewind
****
Functionally identical to M02. Hard disks don't need to be "rewound" of course, but paper tapes used to be! Some CAM programs generate this code instead of M02 at the end of a program, so it's here for compatibility.

****
M48 Restore feed override
****
After an M49, this brings the feed override back to whatever it was just before the M49 call.

****
M49 Cancel feed override
****
This resets the feed override to 100% "from the inside". Use it before

entering a critical section of your program that requires an exact feedrate.

****
M50 Read tachometer
****
  This reads the spindle speed into the program for use with IPR feedrates by reading the spindle index pulse. You must have an index pulse enabled to use this code. See the short section on threading in the setup section for details on how to set this up.

****
M60 Jump to subroutine
****
  Jump to line number given and save return address.
Usage:

N020 M60 #10   ;Jumps to line 10. Use a unique N code when calling subs.

****
M62 Return from subroutine
****
  Back to most recent M60 call.
Usage:

M62 ;Will jump back to line N020 and execute next line in the above example.

See sample.cnc for a simple subroutine example. Please note that when using subroutines, you'll need a unique N code on each calling line so that the program knows where to go back to.

****
M70 Set PLC handshake output to inactive
****
  Two PLC handshaking lines can be configured in TurboCNC. These are for telling external logic, such as an automatic tool changer or what-have-you to do some work. M70 sets the output handshake line to the inactive state. This can also be used to control extra solenoids or relays.

****
M71 Set PLC handshake output to active
****
  Similar to M70, M71 sets the output handshake line to the active state.

****
M72 Wait for PLC handshake input to go inactive
****
  When this code is called, the program stops running and waits for the PLC input line (separate from the output line) to go to the inactive state. This is intended to be used to synchronize the program with external logic that might be

loading more stock, or performing some other function.

****
M73 Wait for PLC handshake input to go active
****
  Similar to M72, this pauses until the PLC line goes to the active state. With either of these two codes, the user can press <ESC> to bypass the paused condition.

****
Code letters:
****

N Line number
F Feedrate
I Interpolation parameter, 1st axis; or infeed parameter
J Interpolation parameter, 2nd axis
K Thread lead parameter (not yet implemented)
T Tool offset
R Arc radius; or release plane
S Spindle speed parameter
D,L,H,Q Future Use
Timer or subroutine parameter

---
9. Command line parameters

- A handful of command line parameters and switches are available in the program. Perhaps the most useful is the .ini file switch. Normally TurboCNC will look for the .ini file with the same filename as the executable in the current directory at startup. If you want to change this, call out the filename on the command line:

turbocnc filename.ini
OR
turbocnc c:\mydir\filename.ini

  The only requirement is that the file end with the ".ini" extension and have no more than eight characters (DOS limitation, sorry!).

- Monochrome mode may be engaged by using a -m switch on the command line. The program will "remember" this if you save the ini file at all while in mono mode for next time.

turbocnc -m

  If you saved your ini file while in monochrome mode, you won't be able to get

back into color mode until you edit this line in your ini file:

Monochrome=TRUE

Change it to FALSE to load in color. It should be in the first ten lines or so.

- You can load up your favorite tool offset file at startup instead of going through the menus like this:

turbocnc -tools filename
OR
turbocnc -tools c:\mytools\filename

"filename" is again limited to eight characters plus a three letter extension.

- You can skip all the menus completely and run a file with just one button, like this:

turbocnc -run filename

where "filename" is a valid G-code file. You'll be brought directly to the automatic machining menu where you can hit ESC to cancel, or another key to run the part. After one run, you'll press a key again and land back in DOS. With a batch file calling the shots, you could probably get a lot of work done with this....

- To just skip the startup screens, use the quick option

turbocnc -quick

- If you exit the program while in machine coordinate mode (G53, or TOOL 0), the position and backlash sense is stored for startup next time. If for some reason you want to bypass this, use the -nopos command line parameter:

turbocnc -nopos

The machine coordinates will then default to 0.0 in every axis, and the backlash preload directions will be cleared.

All of the command line options are mixable, e.g.:

turbocnc -m -run part.cnc -tools standard.ofs millmast.ini -nopos

is a valid input.

In the near future, I expect to add foreign language support to the program as a command line option. If you have a language you want covered, and are willing

to help translate, let me know. This is a lot of work, so don't take it on lightly.

---
10. Suggested resources

Resources on CNC abound. A few of my favorites:

Machinery's Handbook - Industrial Press. A must-have handbook covering many aspects of machining, CNC, and metallurgy. A copy of the pertinent info from the RS-274D Gcode standard is in here.

http://groups.yahoo.com/group/CAD_CAM_EDM_DRO A hobby oriented list group devoted to CNC machines.

alt.machines.cnc Usenet newsgroup on CNC equipment. Commercial bent.

http://www.geckodrive.com Source for stepper and servo drives

http://www.rutex.com More stepper and servo drives

http://www.seanet.com/~dmauch Plug and play CNC drives with motors, very well priced. Some software as well.

http://microsystemsgeorgia.com/cnc.htm Glossary of CNC terms

http://www.cnckits.com Information on how to setup a hobby CNC system, parallel breakout boards

http://www.hobbycnc.com Nice inexpensive drivers and motors for CNC.

http://groups.yahoo.com/group/turbocnc User's group for this program

---
11. Troubleshooting

If the program does nothing when you try to move, check these things:

- Are the ports set up properly?
- Is the command syntax correct?
- Are you jumping to a sub that does not exist?

If your steppers are losing steps, back off on the max speed and acceleration parameters. If they vibrate badly or start in reverse sometimes, bump up the start speed until you get clear of the resonance band. Also check to see if the winding activation sequence is correct, and play with the pulse width parameter. Make it wider until the "sound" is consistent.

"Sticky Stepping" is normally caused by a pulse width that is too short on a step
and direction drive, or noise in the electrical system. Check the grounds, and shield the logic cables if they are at all lengthy. Be aware that not all printer connectors have pins 18-25 as ground - some will have pin 18 only or similar. If the pulse width is too long (above 50 is *really* pushing it) it can
cause dropouts on multi-axis moves.

Also, be sure that you aren't running under Windows, or that a hard-disk caching program isn't running in the background. EMM386.exe is known to cause some trouble with timing under DOS as well. At startup, you'll be warned if something is stealing time from the program or if the computer is way too slow. RAMDRIVE.sys and HIMEM.sys can also be bad actors in this regard, not to mention WINDOWS.

If programs refuse to run, check to be sure the syntax is ok. Look at the program MINIMAL.CNC for an example. Programs can be written in Notepad or MS-DOS EDIT, or any other text editor that saves in ASCII format. Don't use Microsoft Word - it throws extra characters in.

Be sure that you're using just one G/M code per line. This is a common pitfall for users accustomed to other controls.

Sometime switching computers helps, for reasons unknown. TurboCNC reprograms the motherboard timer and does some other things that certain machines don't seem to like. The code is, as of version 3.1, optimized for a 486DX or DX2 processor.

Weird stuff? RT-errors? Circle looks non-circular? If they can be duplicated (e.g.: every time I load this file and press this, then this, it screws up the same way), then send in your ini file and your tear-filled tale of woe to me at admin@dakeng.com. I'll do what I can... Some of you have been quite good at finding problems, and for that I thank you.

---
12. TurboCNC.INI file format

If you look at the TurboCNC.ini file, you'll see that it has a format not unlike that of a standard Windows ini file. There are several sections that define each component of the machine by category, and comments are set off with a semicolon.

Each parameter has a default if it is not read - so the ini file only contains the information necessary to uniquely define your machine. If you only have three axes, then you'll only see definitions for three axes in the file, not all eight.

The first section is titled [General] and contains the number of axes on the machine, and whether Metric mode will be the default as startup, among other things.

```
[GENERAL]
NumberOfAxes=3   ;Number of axes on machine
Metric=FALSE     ;Will not default to Metric mode on startup
Monchrome=FALSE  ;Color at startup
MDIfilename=MDI.CNC ;Default MDI filename
EndSound=STANDARD   ;Sound played at end of gcode file
ListDir=C:\X\       ;Default directory for files
ReverseDelay(ms)=0  ;Optional delay on axis reversal
TimerMode=STANDARD  ;Timer mode switch
```

Some of these are advanced functions that can only be changed in the ini file. Here are the settings and their uses:

Endsound may set to NONE, STANDARD, WEASEL, or STARWARS. This controls the sound effect that is played when you finish running a Gcode file.

ListDir is the default directory for loading and listing files.

ReverseDelay(ms) is a special delay that can be specified for when an axis reverses direction. If you lose steps on a reversal, due to ringing machine structure or low torque, set this delay to 100 milliseconds or so. Any number from 0 to 1000 (one second) is valid. 0 is no delay at all (default).

Timermode selects the timebase used by TurboCNC. This is for a special feature that didn't quite make it into this version. Leave it set to STANDARD.

Next each axis is defined in one of two ways, depending on whether it's a step/direction controlled axis or a phase drive axis:

```
[AXIS1]
Designator=X
StepIncrement= 1.0000000000E-03
IsLinear=TRUE
IsStep/Dir=TRUE
PortAddress=$0378
StepPin=2
IsActiveLow=TRUE
Pulsewidth=50
DirPin=3
LowIsPositive=TRUE
Acceleration= 2.5000000000E+03
StartSpeed= 2.0000000000E+02
MaxSpeed= 1.2000000000E+03
Backlash= 3.0000000000E-03
```

```
HomePosition= 0.0000000000E+00
HomeInPositiveDir=TRUE
```

Most of the above is self-explanatory. For a phase-drive axis, the step and direction pin information is not written, and IsStep/Dir is set to False. Here's an example from my own machine:

```
[AXIS2]
Designator=X
StepIncrement= 5.0000000000E-05
IsLinear=TRUE
IsStep/Dir=FALSE
Phases=8
Phase1=1001XXXXXXXX
Phase2=0001XXXXXXXX
Phase3=0011XXXXXXXX
Phase4=0010XXXXXXXX
Phase5=0110XXXXXXXX
Phase6=0100XXXXXXXX
Phase7=1100XXXXXXXX
Phase8=1000XXXXXXXX
Phase9=XXXXXXXXXXXX
Phase10=XXXXXXXXXXXX
Phase11=XXXXXXXXXXXX
Phase12=XXXXXXXXXXXX
Phase13=XXXXXXXXXXXX
Phase14=XXXXXXXXXXXX
Phase15=XXXXXXXXXXXX
Phase16=XXXXXXXXXXXX
PortAddress=$0378
Acceleration= 2.2500000000E+03
StartSpeed= 9.0000000000E+02
MaxSpeed= 4.0000000000E+03
HomeInPositiveDir=TRUE
```

Note that all 16 phase patterns are written, even if they are not all used. This is not strictly necessary, but it makes the computer happier as the file access takes longer if the phase is not found. Once again, the pin order is from left to right: 2-9, 1, 14, 16, 17. A 0 is low, 1 is high, and X is "no state change".

Each of the accessory I/O functions is written the same way. Following are two examples:

```
[SPINDLE_ON_(OUTPUT)]
Enabled=TRUE
Pin=17
PortAddress=$0378
ActiveHigh=TRUE
```

[HomeSwitch1]
Enabled=TRUE
Pin=10
PortAddress=$0378
ActiveHigh=TRUE

This should be pretty self-explanatory also. There are 18 I/O functions in TurboCNC, and each of the headers is spelled as follows:

[SPINDLE_ON_(OUTPUT)]
[SPINDLE_FWD_(OUTPUT)]
[COOLANT_A_ON_(OUTPUT)]
[COOLANT_B_ON_(OUTPUT)]
[PLC_HANDSHAKE_ACTIVE_(OUTPUT)]
[DRIVE_ENABLE_1_(OUTPUT)]
[DRIVE_ENABLE_2_(OUTPUT)]
[DRIVE_ENABLE_3_(OUTPUT)]
[PANIC_STOP_(INPUT)]
[LIMIT_SWITCH_1_(INPUT)]
[LIMIT_SWITCH_2_(INPUT)]
[LIMIT_SWITCH_3_(INPUT)]
[PLC_HANDSHAKE_ACTIVE_(INPUT)]
[SPINDLE_INDEX_(INPUT)]
[SPINDLE_ENCODER_A_(INPUT)] <--- this is not used at present
[TOUCH_PROBE_(INPUT)]
[JOG_ENCODER_CH_A_(INPUT)]
[JOG_ENCODER_CH_B_(INPUT)]

So to define one of the functions, write the header as above on one line, followed by Enabled=TRUE and the pin/port/state as appropriate. Remember that pins 2-9 are always output, pins 10,11,12,13,15 are always input, and pins 1,14,16,17 can be either input or output on the parallel port.

The Jog increments for Imperial and Metric units follow after that:

[JogIncInch]
0= 1.00000000000000E-0004
1= 5.00000000000000E-0004
2= 1.00000000000000E-0003
....

[JogIncMM]
0= 1.00000000000000E-0003
1= 5.00000000000000E-0003
2= 2.00000000000000E-0002
....

You can only edit the jog increments by manually editing the .ini file. For

jogging wheel users, the first five increments in either system are the only ones available through spinning the wheel. This helps prevent using a 1" jog, say, and accidentally crashing your machine.

Lastly, the tool position and backlash sense are written if you exited while in G53 or Tool 0:

```
[POSITION]
1=00000000E+0000
2 ....

[LASTBACKLASH]
1=1
2 ....
```

If the position is not valid, the value on the right side of the equal sign will be the word "empty":

```
[POSITION]
1=empty
2 ....
```

I'd have to recommend that the setup be done from within TurboCNC, as the error checking is more robust within the program. However, those that can't resist tinkering here can have at it!

This format ensures upward compatibility with new versions of the program as they are released.

---

13. Contacting the author

I try to respond to email pretty quickly on most days. My email address is:

dkowalcz@dakeng.com

Or if you prefer the snail mail route:

Dave Kowalczyk
4904 Glenwood Ave
Everett WA 98203 USA

Registration payments ($20) can be sent via Paypal to admin@dakeng.com, or by check to the address above.

Hope you enjoy using the software! If there are features you want to see in a future version of this program, don't hesitate to let me know about it. So far, I think I've put in just about everything people have asked for.