

Introducing CNC Construct™

Software Engineered for Manual CNC Programming

Abstract:

Part programming for computer controlled machining is very important to the manufacturing industry. It is so important in fact that many technologies have been developed and continue to evolve to accomplish the programming task. Specifically, language based programming, graphical CAD/CAM, conversational controllers, and most recently, STEP-NC were all developed for part programming. However, the one programming technique that has not advanced with computers is the oldest technique of all: manual programming, i.e., programming at the G-code level.

Presented here is **CNC Construct™**, software that was engineered to provide a solution for the math, time, and errors associated with G-code programming. Additionally, CNC Construct provides power and other benefits previously unavailable to manual programming.

Here, we will analyze three G-code programs (a 2D, a 2½D, and a turning operation) to seek trends and repeated commands and discuss how CNC Construct can quickly, accurately, and automatically write much of the code of these types of programs. Additionally, CNC Construct was developed to incorporate data from wireframe CAD to create the cutter coordinate and cutter motion information for manual programming. The procedure/requirements to use CAD to accomplish this task will be shown.

Finally, to show how it all comes together, a 2½D manual programming example will be presented using CNC Construct and wireframe CAD. This programming example can also be viewed as a video on the internet at <http://www.CNC-Construct.com>.

Background:

G-code is the standard programming language used to guide machine tools and was developed by the Electronics Industries Association (EIA) in the early 1960's. Programming at the G-code level is also referred to as manual programming. A simple 2½D G-code program for a drilling/grooving operation is shown in Appendix A-1.

Although G-code is the standard language used to guide machine tools, programming at the G-code level is time consuming, math intensive, and accordingly, error prone. Because of these and other problems much development has been accomplished to automate, avoid, and most recently, to even eliminate G-code.

The first technology to use computers to automate direct G-code programming was with language based processors such as APT, ADAPT, and AUTOSPOT.

The goal of language based processing was to create G-code programs automatically. This was done as an effort to describe both the many physical details of the part and the process

requirements of the machining operation such that the “processor” could do the mathematical calculations for the cutter coordinates. The cutter coordinate output file was then “post-processed” to the specific G-code requirements of the machine tool. An APT program is shown in Appendix A-2.

CAD/CAM is the graphical evolution of language-based programming. There are many players in this industry, including ProEngineer, CATIA, SurfCAM, MasterCAM, GibbsCAM, and EdgeCAM to name a few.

The benefit of CAD/CAM, as compared to language based programming, is that the geometry of the part is displayed graphically and the system can design the tool paths automatically in lieu of the written definitions used with language based programming. Textbook examples using CAD/CAM to write simple 2½D programs can be found in Valentino¹ and Nanfara².

Conversational programming (shop floor programming) has been developed as an added function of a machine tool control and avoids G-code programming altogether. With this technology, the machining operation is not “programmed” so much as the finished part and aspects of the machining operation are “described” through a series of graphical and menu driven prompts and functions. The machine control processes the information, performs the necessary tool motion calculations, and develops the program automatically. A textbook reference that includes a discussion of conversational programming can be found in Lynch³. Another reference for conversational programming is available on the Internet at:

<http://geindustrial.com/products/applications/ConvPrgonCNC.pdf>

The next technology with the intent to eliminate G-code altogether is a file transfer standard called STEP-NC. This engineering design file includes material, 3-D geometry, 3-D feature, 3-D tolerance, and 3-D process information. With this information, a CNC controller will have and/or be able to define tool paths, fixtures, stock, tools, feeds and speeds, and all other aspects of material removal without G-code. More information on this standard is available on the Internet at: <http://www.steptools.com/library/stepnc/>

Traditional G-code Part Programming:

With all the aforementioned technology working against G-code and the future thereof, it is still the standard programming language interpreted by machine tool controllers to cut parts. Put another way, G-code is the standard “man-machine” interface of computer controlled machining.

As a consequence of the G-code standard, competence is obviously important for those individuals involved with computer controlled machining. This importance explains why G-code is still taught at all levels of machining education, through and including technical schools, trade schools, and universities.

Manual G-code programming does have problems. Specifically, the time and the errors associated with manual tasks are significant. Depending on the requirements of the part, one major source of time and errors can be attributed to solving math problems to obtain cutter coordinates.

However, **at its core**, G-code commands are a direct correlation to material removal decisions, for example:

Conventional Machining Decision	G-code Command
Jog cutter to start position	G00 X## Z##
Adjust spindle speed to 2000RPM	S2000
Switch spindle to On	M03
Open valve for coolant	M08
Adjust feed rate to 0.02"/rev	F0.02
Move Cutter into Part	G01 Z##
Etc.	

The real power of G-code part programs and computer controlled machining compared to conventional machining is the accuracy of cutter coordinates and advanced programming features that can be interpreted quickly by the machine tool control. The most often used areas of advanced G-code programming include automatic tool changes, cutter length compensations, cutter radius compensations, canned cycles, and subroutine programming.

Regarding the effort required of manual programming, the first task is to plan the machining operation. Actually this step is required regardless of the programming technique. This step includes deciding on the machine, the tools, the holding fixture, calculating the feed rates, and the spindle speeds if necessary (wire EDM machines, water jets, plasma jets, etc. do not have spindles).

The next task is to obtain the cutter coordinates. Traditionally, the cutter coordinates are obtained either from the part drawing for simple part features (e.g., Appendix A-1) or from mathematical calculations for bolt hole patterns, contours, pockets, and every other part feature that is not dimensioned directly on the part drawing.

These cutter coordinates may be tabulated in a coordinate sheet to help organize the programming task. Most (if not all) CNC textbooks discuss coordinate sheets, including Lynch⁴, Smid⁵, and Nanfara⁶. The example coordinate sheet shown in Appendix B⁷ dates back to 1960.

Finally, the G-code program is written, command after command, line after line, top to bottom. In essence, each step of the machining operation is sequentially described with the appropriate G-code command. Following this task, the program is checked, transferred to the machine tool, and the part is cut.

With the exception of using calculators to help with the math, and text editors used to write the program, the tools and the requirements of manual programming have not changed for the 50+ years of computer controlled machining.

Advanced G-code Part Programming:

As mentioned, G-code programs are the sequential, step-by-step translation of machining decisions to a language that both the programmer and machine tool control understand. And accordingly, G-code programs written manually have always been written in this sequential, step-by-step, top to bottom manner.

But to automate the repetitive/tedious tasks of manual programming, these same G-code programs must be analyzed from a different perspective: “the forest, not the trees.” Specifically, all 2D, 2½D, and turning programs i.e., the programs that **can** be written manually, can be broken up into three categories:

- 1) Cutter coordinate and cutter motion commands,
- 2) Cutter on and cutter off commands, and
- 3) Preparatory commands that include program beginning, tool change, and program end commands.

Appendices C-1, C-2, and C-3 show examples of G-code programs for a 2½D milling, a 2D torch, and a turning operation, respectively. The three different categories of each program are formatted such that:

- 1) the cutter coordinate/cutter motion commands are normal type,
- 2) the cutter on/cutter off commands are underlined, and
- 3) the program beginning, tool change, and program end commands are shown as **bold**.

The significance of this analysis and this three-category perspective is that many G-code formatting requirements and many G-code commands repeat in each of these categories. CNC Construct was developed to write the most repetitive/tedious portions of these categories in accordance to the programmer’s requirements and do so quickly and accurately.

Format examples:

Examine the cutter coordinate/cutter motion sections of the three examples in Appendix C. Notice that **1)** each cutter coordinate in the milling and torch operation has a leading “X” or “Y”, whereas the turning operation has a leading “X” or “Z.” Notice that the coordinates are **2)** rounded to 2, 3, or 4 places. Notice that the cutter motions most often used are **3)** G00, G01, G02, or G03 corresponding to rapid, linear, clockwise, and counterclockwise motion respectively.

If a cutter motion or cutter coordinate command is not shown, the motion or coordinate **4)** has not changed, and need not be shown. Notice that circular motion requires **5)** the center coordinate be either absolute coordinates from program zero (torch) or incremental distances from the arc start (mill).

CNC Construct was developed to quickly and accurately write the G-code that includes these five, and many other format requirements of the cutter coordinates and cutter motion commands.

Cutter on/cutter off example:

Now examine the cutter on/cutter off commands of the torch operation in Appendix C-2. Two G-code commands are used to turn the cutting (oxygen) off just **prior** to a rapid motion (“G00”) command and two G-code commands are used to turn the cutting on just **after** that same rapid motion (“G00”) command. These four commands are repeated without change throughout the entire program.

Manually writing these commands into the program is a very repetitious/tedious activity prone to mistakes. CNC Construct was developed to prompt for and quickly and accurately write these cutter on/cutter off commands.

Preparatory command examples:

And finally, view the preparatory commands of the three examples in Appendix C. The preparatory sections for any milling program, for any torch program, and for any turning program, are almost identical from program to program **on the same machine**. Typically, only speeds, feeds, tool numbers, compensation registers, etc., specific to each machining operation change from one program to another on the same machine.

Two textbook references that discuss this commonality of preparatory commands can be found in Lynch⁸ and Smid⁹.

CNC Construct can quickly paste these commands from user-defined, machine-specific templates, instantaneously adding this category to the G-code program.

Coordinate Work Sheet computer file:

As CAD systems have developed with useful features and provide much more power than the drafting board, and as word processors have developed with useful features and provide much more power than typewriters, CNC Construct has been developed with useful programming features and much more power than can be done by hand (or with text editors).

It was previously discussed that CNC Construct was developed to 1) quickly and accurately write the formatted G-code commands of the cutter coordinates and cutter motion, 2) prompt for and write the cutter on/cutter off commands into the G-code program, and 3) to paste the preparatory commands from a user-defined machine-template into the G-code program.

Additionally, two more very powerful features incorporated into CNC Construct include the ability to graphically verify cutter coordinates to show the tool path and the ability to modify (scale, copy, rotate, etc.) the coordinates automatically.

Neither graphical verification of the tool path, nor the automatic modification of cutter coordinates has been possible with traditional manual programming. To appreciate how CNC Construct advances manual programming and incorporates these powerful features, the standard table of cutter coordinates needs to be analyzed.

At the heart of all G-code programs are the cutter coordinates. The cutter coordinates are important because they define the location of the cutter and the location of the cutter defines where material will be removed.

The cutter coordinates were shown in a tabulated hardcopy format in Appendix B. The purpose of the table is to be an organizational tool that the programmer can quickly reference. Although a benefit, the hardcopy table does have serious limitations.

The most obvious limitation and therefore the first advancement of the coordinate table is to convert it from hardcopy paper to a computer file.

Next: when the programmer references the coordinate table to write the G-code program, he/she must have committed to memory what tool motion corresponds to each set of coordinates because it is not included with the coordinate table. The coordinate sheet computer file must include cutter motion.

Next, although there are columns for X, Y, and Z coordinates, the table does not include columns for the center coordinates of circular motion. Since circular motion is one of the four basic motions (rapid, linear, clockwise, counterclockwise) of computer controlled machine tools, the coordinate sheet computer file must include center coordinates for circular motion.

And finally, if there is more than one tool used, the programmer must also have committed to memory which cutter coordinates correspond to which tool. Again, the coordinate sheet computer file must include tool information.

Obviously, center coordinates, cutter motion and tool changes are major components in G-code programs. These three missing items must be included in a cutter coordinate computer file if CNC Construct is to write and format the cutter motion/coordinate commands and insert the tool change commands **automatically**.

By including these missing items into a cutter coordinate computer file, CNC Construct can write the G-code, but can also graphically verify the tool path defined by the cutter coordinates, and automatically modify (scale, copy, rotate, etc.) these coordinates. These three powerful features significantly advance programming at the G-code level.

CNC Construct has incorporated these items in a coordinate work sheet file (*.cws) format as follows:

```
RPD,TL#,XX.XXXXXX,YY.YYYYYY  
LNR,TL#,XX.XXXXXX,YY.YYYYYY  
CW,TL#,XX.XXXXXX,YY.YYYYYY,II.IIIII,JJ.JJJJJ  
CCW,TL#,XX.XXXXXX,YY.YYYYYY,II.IIIII,JJ.JJJJJ
```

The file is of a simple comma delimited ASCII file where "RPD," "LNR," "CW," and "CCW" are the acronyms for rapid, linear, clockwise, and counterclockwise cutter motion respectively.

"TL#" is an organizational programming prompt to designate, in this example, the tool number that corresponds to the cutter coordinates that follow. This programming prompt was incorporated into CNC Construct for tool change purposes, but as a prompt it could also designate **changes** in cutter depths, feed rates, cutter compensation, part features, or anything else as needed by the programmer.

"XX.XXXXXX," and "YY.YYYYYY" are the coordinates of the cutter, relative to program zero. They also correspond to the radial "X" and "Z" coordinates for turning.

"II.IIIII," and "JJ.JJJJJ" are the center point coordinates, also relative to program zero, of the circular moves of the cutter and, accordingly, these also correspond to the "I" and "K" coordinates for turning.

Relative to the hardcopy coordinate table, this advanced coordinate work sheet file format contains enough information for CNC Construct to:

- 1) Convert the cutter coordinate/cutter motion to G-code,
- 2) Graphically verify the tool path defined by the cutter coordinates/cutter motion, and
- 3) Automatically modify (scale, copy, rotate, etc.) the cutter coordinates/cutter motion.

Use CAD to do the math:

Undoubtedly, the biggest problem with manual programming is the math required to obtain cutter coordinates. This is a problem on three fronts: solving math problems can be quite difficult, can take significant time, and is a great source for errors. And an error with the cutter coordinate in the G-code program will very likely scrap the part. A simple solution to this math problem is to use wireframe CAD.

For some parts, everything on a part drawing may be dimensioned with CAD to obtain all of the cutter coordinates necessary to write a complete part program, as shown by the programming example in Appendix A-1. Conversely, if the drawing does not dimension every necessary cutter coordinate, math is the fundamental technique used to obtain cutter coordinates as per the text book approach to manual programming. Obviously, based on the number of CAM systems, and conversational controllers in use, math is not practical.

Wireframe CAD has the power to create offset lines and arcs that can be used to design tool paths. Most (if not all) wireframe CAD systems also have a "Measure" feature that can be used to measure the distances from program zero to the intersections of the lines and arcs of the tool path. Using CAD to measure cutter coordinates can be much more efficient than doing the math but transferring the coordinates manually to the hardcopy coordinate sheet can still be tedious and another source for errors.

There is a much more powerful, accurate and faster feature of CAD than the "Measure" feature to obtain cutter coordinates, and it was somewhat presented in the coordinate table shown in Appendix B-1. Specifically, this coordinate table labels each row of coordinates as points and almost every wire frame CAD can create points.

By analyzing the standard ASCII data file (*.DXF) exported by most (if not all) CAD systems and by setting up a simple procedure, potentially any wireframe CAD system can be used to create objects that correspond to the cutter coordinates and cutter motion commands.

CNC Construct has incorporated this CAD technology with the procedure that follows:

- 1) The CAD must export the objects in the sequence they were created
- 2) The coordinates must be on their own layers as organized by programming prompt
- 3) The first object must be a point at Program Zero
- 4) Two sequential identical points are extracted to RPD,LAYER,XX.XXXXXX,YY.YYYYYY
- 5) One point is extracted to LNR,LAYER,XX.XXXXXX,YY.YYYYYY
- 6) For arcs less than 180 degrees, a point at arc end, followed by a point at arc center becomes either
CCW,LAYER,XX.XXXXXX,YY.YYYYYY,II.IIIII,JJ.JJJJJ or:
CW,LAYER,XX.XXXXXX,YY.YYYYYY,II.IIIII,JJ.JJJJJ
the direction chosen being dependent on which arc is less than 180 degrees.
- 7) One line from arc endpoint to arc center point is extracted to
CW,LAYER,XX.XXXXXX,YY.YYYYYY,II.IIIII,JJ.JJJJJ
- 8) Two identical sequential lines from arc endpoint to arc center point are extracted to
CCW,LAYER,XX.XXXXXX,YY.YYYYYY,II.IIIII,JJ.JJJJJ
- 9) The CAD file is saved/exported to DXF format
- 10) CNC Construct will search the DXF file for the points and lines in the selected layers and will generate the CWS file automatically.

The primary G-code programming benefit here is that math is eliminated, time is saved, and accuracy is improved. A secondary benefit is that the same CAD system can be used to design the part, generate the part drawing, design tool paths and create the objects that correspond to cutter coordinates/cutter motion without transfer to a CAM system.

Programming Example:

G-code programs, coordinate sheets, and data files exported by CAD systems have undergone significant analysis for the purpose of developing CNC Construct. Perhaps the best way to show the results of this work can be done by stepping through the often referenced programming example shown in Appendix A-1 using CNC Construct and CAD.

As with any programming job, the first thing to do is to select the machine – a mill, the holding fixture – a vice, and the tools, spindle speeds, and feeds rates. The first tool will be a ¼” drill to drill the four holes, set spindle speed at 3000rpm and feed rate at 15”/min. The second tool will be a ¼” flat end mill to machine the counterbore and the grooves, set spindle speed at 2000rpm, and feed rate at 12”/min. Set program zero at the top lower left corner of the part.

Because the part drawing dimensions every part feature needed for the machining operation, the coordinate work sheet could be created manually. For the purpose of this demonstration though:

Step 1) use the CAD system to create points on the part according to the aforementioned CAD discussion:

- 1) On new layer “T1” create a point at lower left corner of part (Program Zero).
- 2) Create two points each at each drilled hole: (0.5, 0.5), (1.5, 1.5), (0.5, 2.5), (0.5, 4.5).
- 3) On new layer “T2”, create two points at the upper counterbore: (0.5, 2.5)
- 4) Follow the centerline of the groove with a single point at (1.5, 2.5), (1.5, 3.5), (3.5, 3.5), (3.5,1.5), (1.5, 1.5)
- 5) Create two points at the lower counterbore: (0.5,0.5)
- 6) Follow the centerline of the groove with single points: (4, 0.5), (4, 1)
- 7) Create a point at the center of the circular move: (4, 1)
- 8) Follow the groove again with single points: (4.5, 4), (4, 4.5), (0.5, 4.5)
- 9) Save/export the file as “groove.dxf”.

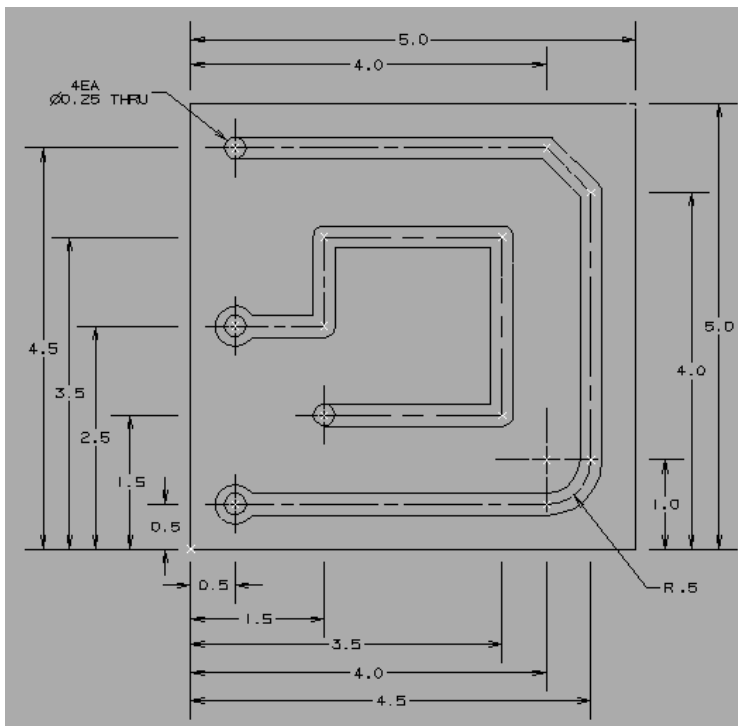


Figure 1. CAD drawing with points corresponding to motion/coordinates.

Step 2) With CNC Construct, extract layers T1 and T2 from groove.dxf. A coordinate work sheet (GROOVE.CWS) will be created automatically:

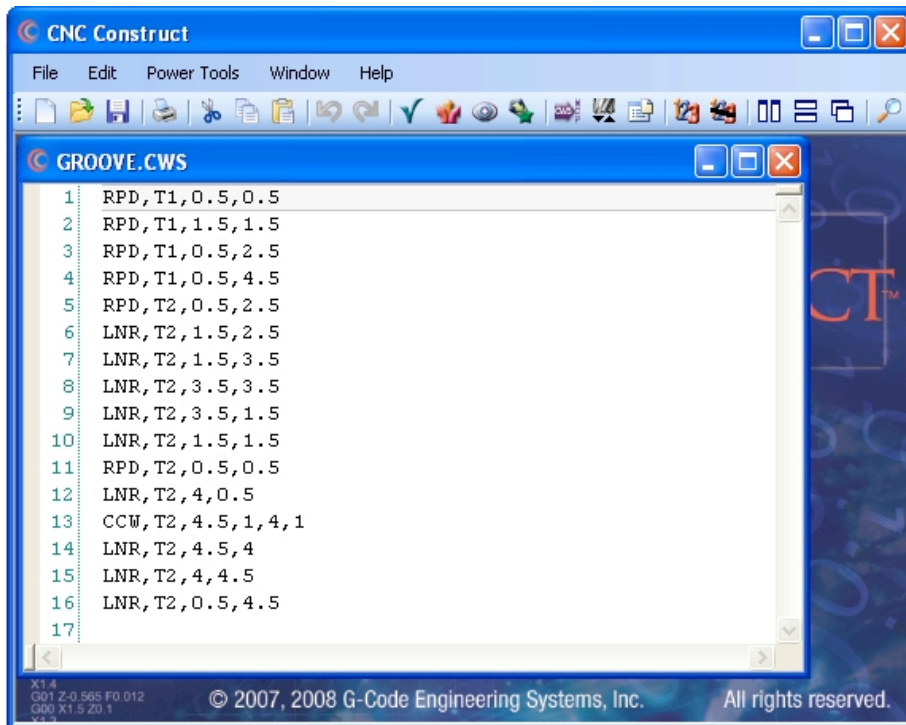
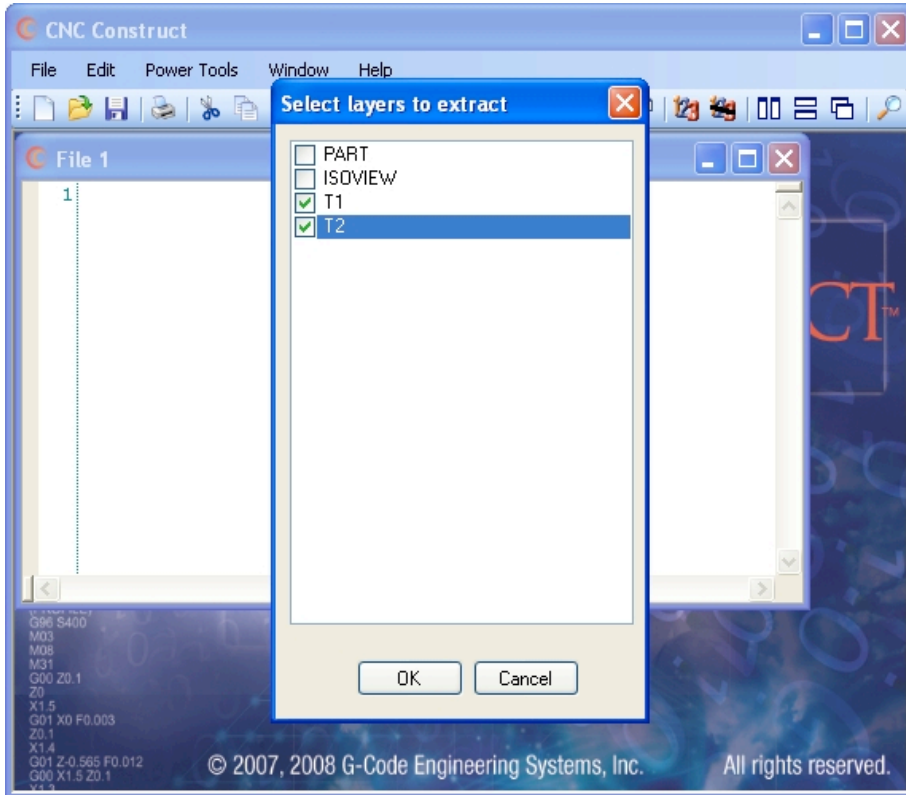


Figure 2. Dialog box to extract layers from DXF file and the coordinate work sheet file type.

Step 4) Convert and format these coordinates and motion data to G-code format according to the requirements of the operation. Note the many formatting options available to the programmer:

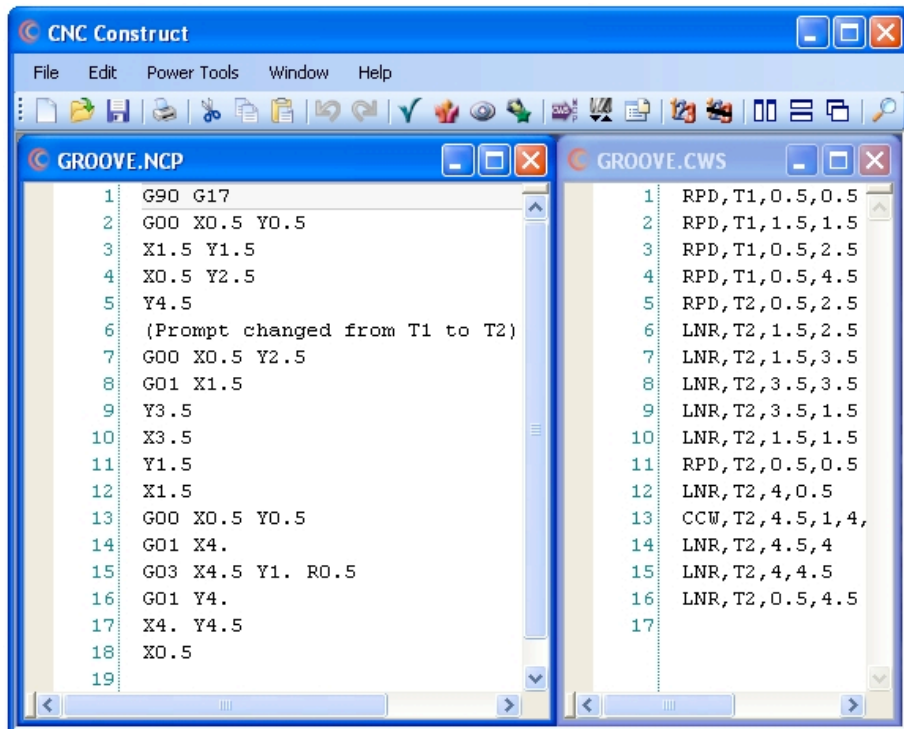
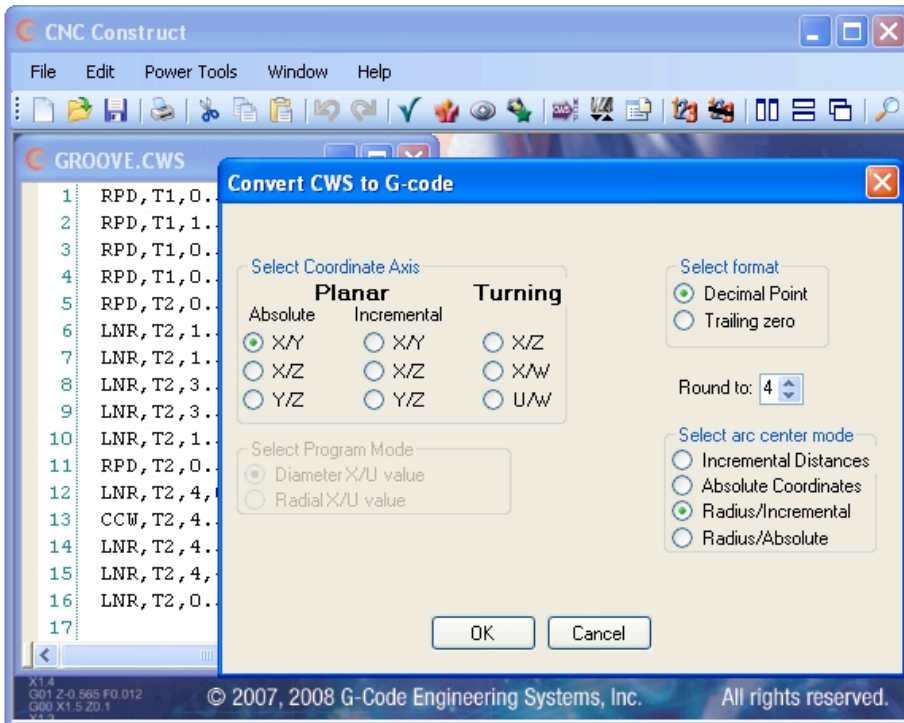


Figure 4. Dialog box of choices and results of conversion of CWS to G-code

Step 5) Add cutter on/cutter off commands for tool 2. (A canned cycle will be used for tool 1.) The cutter on commands include a linear move (G01) to the top surface (Z0), incrementally (G91) move X0.1, clockwise (G02) circular ramp into the part three times to get to the 0.125" depth, cut a final full circle to clean the ramp, go back to absolute motion (G90).

These cutter on commands are used to make the required 0.45" diameter x 0.125" deep counterbore using the 0.25" diameter flat end mill. The cutter off command is a rapid motion command (G00) above the part Z0.1.

Regardless of whether you have two, ten, or twenty cutter on/cutter off commands, and regardless of whether these commands need to be placed in the G-code program two, twenty, or two thousand times, you only need to enter them once as prompted and CNC Construct will accurately place them in the G-code program as quickly as the computer can process them.

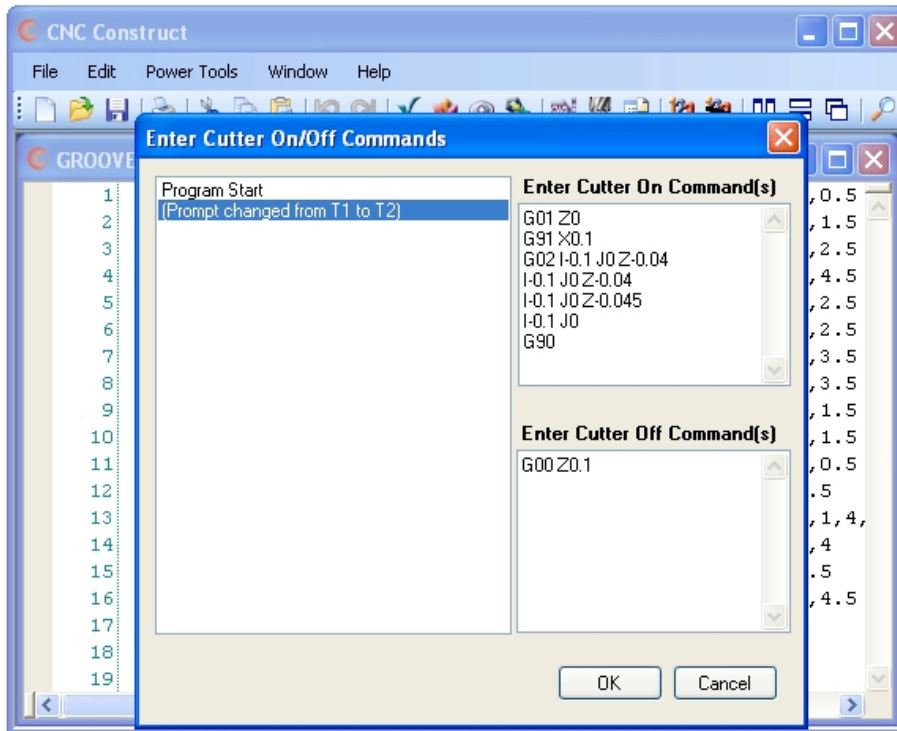


Figure 5. Dialog box to prompt for cutter on/cutter off commands.

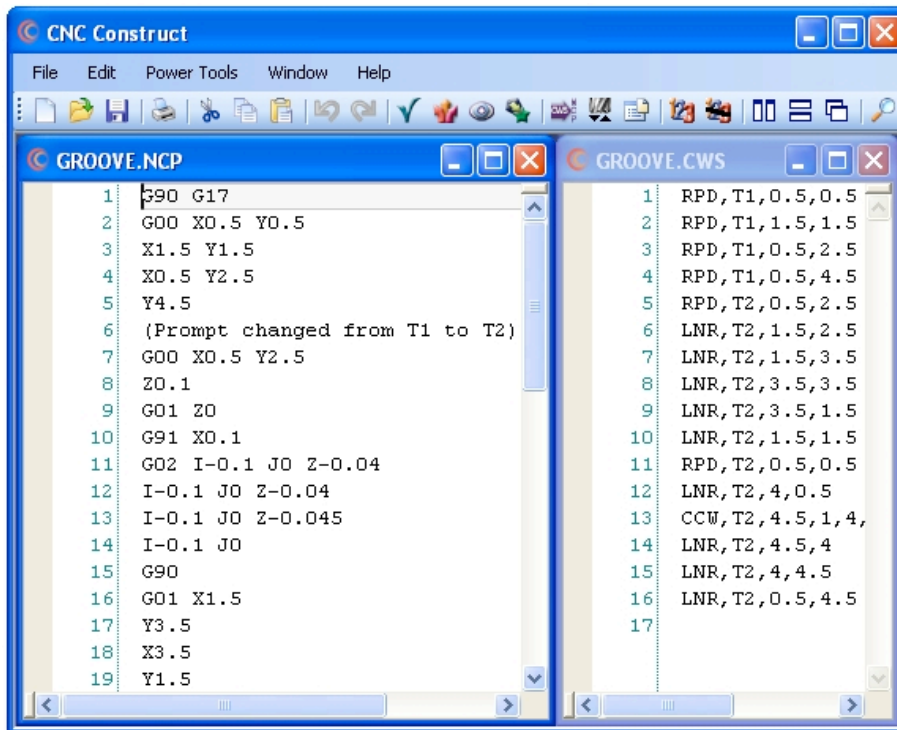


Figure 6. Results of automatically adding cutter on/cutter off commands to tool 2.

Step 5) Add preparatory commands from template:

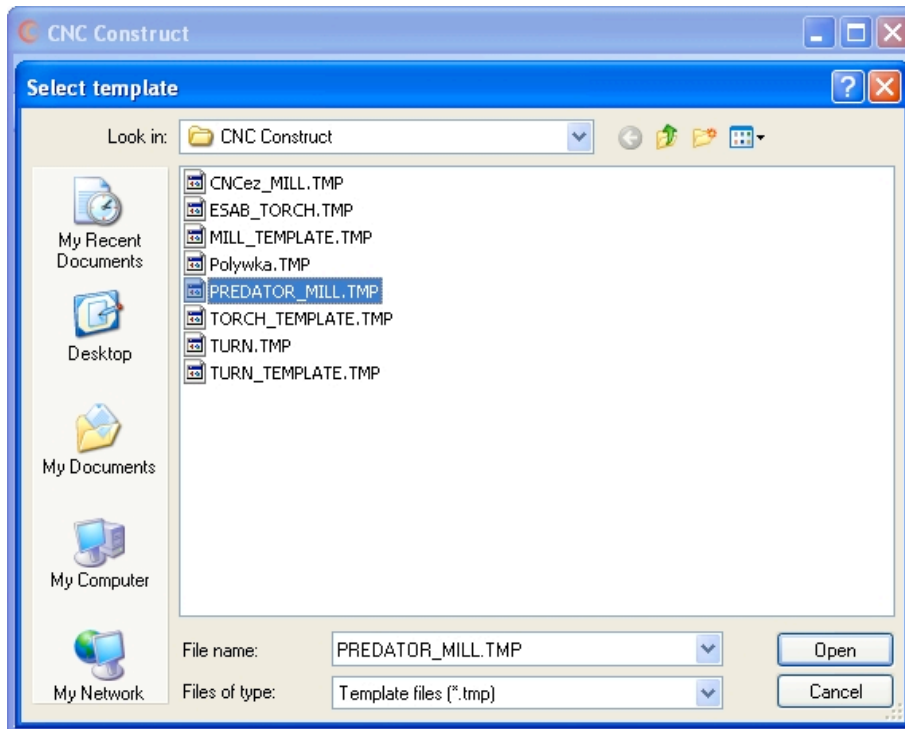


Figure 7. Choices of template files are shown.

Step 6) Open the template file, (which in this example includes the tool library,) and manually edit canned cycles, tool numbers, compensation registers, speeds, feeds, etc. After the edits are complete, use CNC Construct to number the NC program.

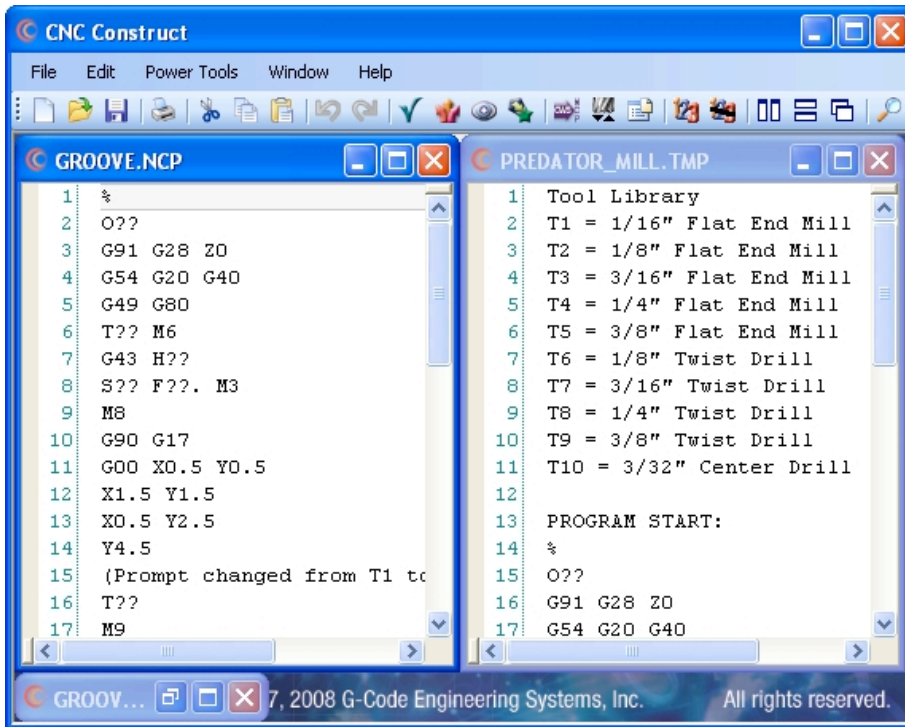


Figure 8. The NC program must be edited with correct tools, speeds, feeds, etc.

The result of this manual programming task is shown in Appendix A. For reference, the template file used for this example is shown in Appendix D. Note that the template file uses the “?” symbol to identify what commands need edits. This symbol is used to prompt the programmer for the most common edits (speeds, feeds, tool numbers, compensation registers, etc.) required.

Step 7) Simulate or verify the CNC program with software such as that found in Valentino¹⁰. As mentioned in the Abstract, this programming example can also be viewed as a video on the internet at <http://www.CNC-Construct.com>.

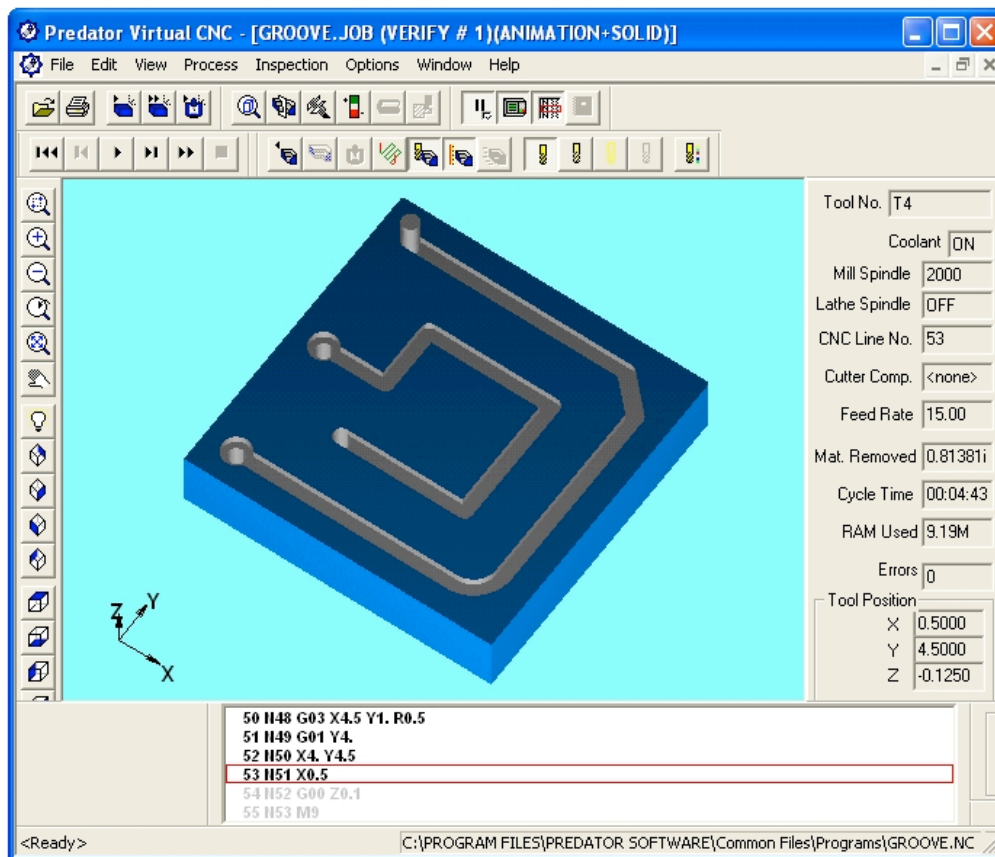


Figure 9. Graphical simulation of G-code program.

Conclusion:

With language based programming, CAD/CAM, conversational programming, and STEP-NC, technologies have been developed and continue to evolve to automate, avoid, and, even eliminate G-code programming.

Unlike the aforementioned technologies, CNC Construct was engineered to **advance** G-code programming and can save time, eliminate errors, and, when used with CAD, eliminate math. These are the three primary problems associated with manual programming.

The G-code programs for a 2D, a 2½D, and a turning operation were each analyzed and broken up into three categories: 1) cutter coordinate/cutter motion, 2) cutter on/cutter off, and 3) preparatory commands that include program beginning, tool change, and program end commands.

The hardcopy coordinate table that has long been associated with manual programming was shown to be missing cutter motion information, center coordinate information for arc motion, and tool information. This information was incorporated in an advanced coordinate work sheet computer file.

The coordinate work sheet file can be used by CNC Construct to automatically format the cutter coordinates/cutter motion data to G-code, to graphically verify the tool path defined by the cutter coordinates/cutter motion and to automatically modify (scale, copy, rotate, etc.) the cutter coordinates/cutter motion.

CNC Construct has incorporated a simple procedure that can take advantage of the inherent power of wire frame CAD to accurately create points and lines that correspond to cutter coordinates and cutter motion. CNC Construct can extract the objects from the CAD data file and automatically create the coordinate work sheet (*.cws) file.

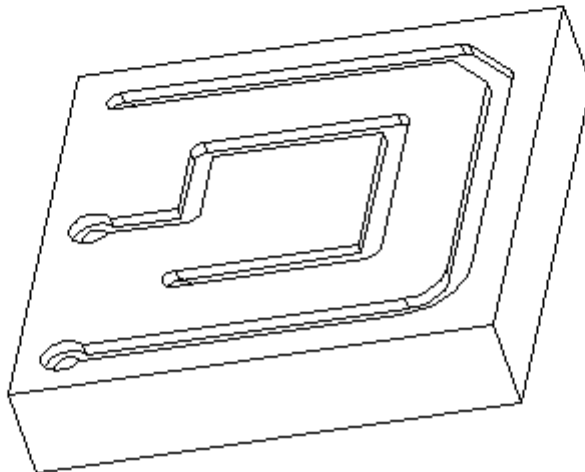
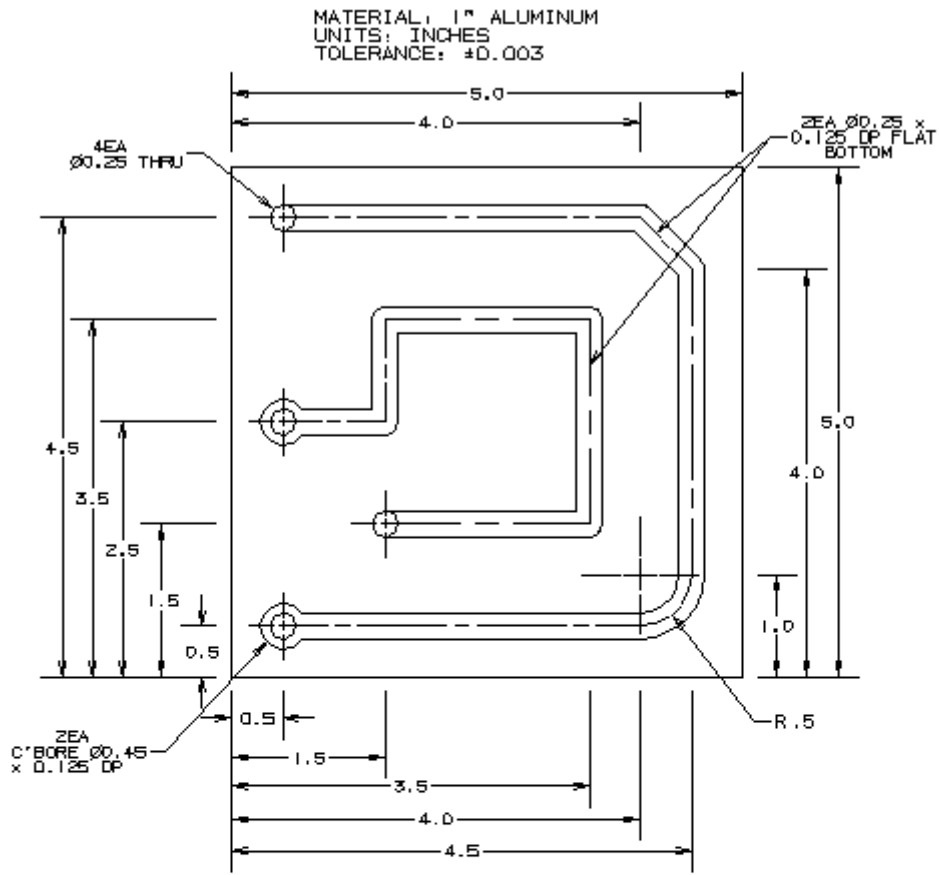
CNC Construct can prompt for and add cutter on/cutter off commands to G-code programs, and add the preparatory commands which includes the program beginning, tool change, and program end commands from user-defined templates to G-Code programs.

References:

- [1] James V. Valentino and Joseph Goldenberg, Introduction to Computer Numerical Control, 3rd Edition (New York, Prentice Hall, 2002), p.442-432.
- [2] Frank Nanfara, et. al., The CNC Workshop – A Multimedia Introduction to Computer Numerical Control (Kansas, Schroff Development Corporation, 2002), p247-297.
- [3] Mike Lynch, Computer Numerical Control for Machining, (New York, McGraw-Hill, Inc., 1992) p291-322.
- [4] Lynch, p98
- [5] Peter Smid, CNC Programming Handbook: a comprehensive guide to practical CNC programming, 2nd Edition (New York, Industrial Press Inc., 2003) p40.
- [6] Nanfara, et. al., p98.
- [7] Frank W. Wilson, Editor-in-Chief, Numerical Control in Manufacturing prepared by the American Society of Tool and Manufacturing Engineers (New York, McGraw-Hill Book Company 1963), p164.
- [8] Lynch, p196.
- [9] Smid, p46.
- [10] James V. Valentino and Joseph Goldenberg, p445-511.

Appendix A-1

G-code Programming Example

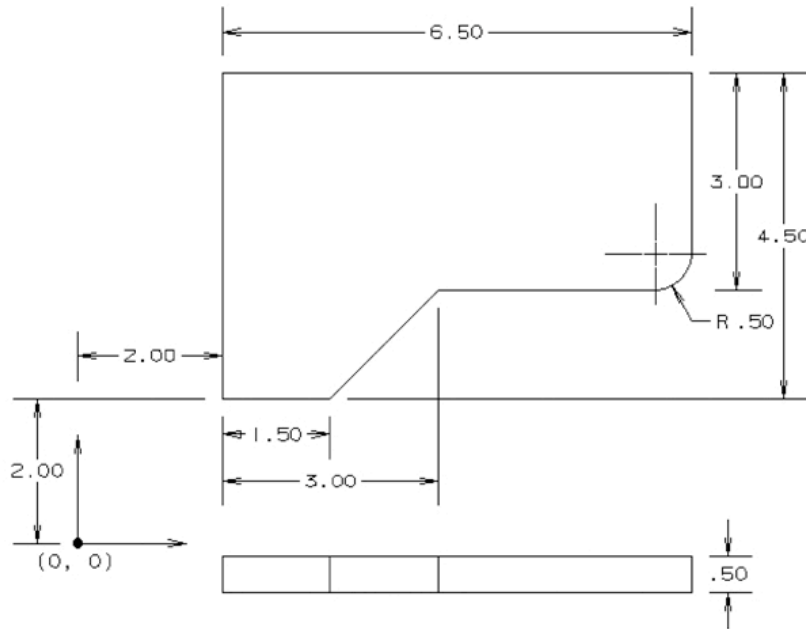


G-code Program:

```
%
O10
N01 G91 G28 Z0
N02 G54 G20 G40
N03 G49 G80
N04 T8 M6
N05 G43 H8
N06 S3000 F12. M3
N07 M8
N08 G90 G17
N09 G00 X0.5 Y0.5
N10 Z0.1
N11 G83 Z-1.1 Q0.25 R0.1
N12 X1.5 Y1.5
N13 X0.5 Y2.5
N14 Y4.5
N15 (Prompt changed from T1 to T2)
N16 T4
N17 M9
N18 M5 G91 G28 Z0
N19 G40 G49 G80 G90
N20 M6
N21 G43 H4
N22 S2000 F15. M3
N23 M8
N24 G00 X0.5 Y2.5
N25 Z0.1
N26 G01 Z0
N27 G91 X0.1
N28 G02 I-0.1 J0 Z-0.04
N29 I-0.1 J0 Z-0.04
N30 I-0.1 J0 Z-0.045
N31 I-0.1 J0
N32 G90
N33 G01 X1.5
N34 Y3.5
N35 X3.5
N36 Y1.5
N37 X1.5
N38 G00 Z0.1
N39 X0.5 Y0.5
N40 G01 Z0
N41 G91 X0.1
N42 G02 I-0.1 J0 Z-0.04
N43 I-0.1 J0 Z-0.04
N44 I-0.1 J0 Z-0.045
N45 I-0.1 J0
N46 G90
N47 G01 X4.
N48 G03 X4.5 Y1. R0.5
N49 G01 Y4.
N50 X4. Y4.5
N51 X0.5
N52 G00 Z0.1
N53 M9
N54 G00 G80 Z15. M5
N55 G91 G28 Z0
N56 G40 G49 G90
N57 M30
%
```

Appendix A-2

APT Programming Example:



```
PARTNO      PART71
MACHIN/BRIDGE, 3
CLPRINT
OUTTOL/0.002
SETPT=POINT/0,0,1
P1 = POINT/2,2,0.5
P2 = POINT/3.5,2,0.5
P3 = POINT/5,3.5,0.5
P4 = POINT/8,4,0.5
P5 = POINT/8.5,6.5,0.5
P6 = POINT/2,6.5,0.5
PL1 = PLANE/P1, P2, P3
PS = PLANE/PARALEL,PL1,ZSMALL,0.5
C1 = CIRCLE/CENTER,P4,RADIUS,0.5
L1 = LINE/P2, P3
L2 = LINE/P3,RIGHT, TANTO, C1
L3 = LINE/P5, LEFT, TANTO, C1
L4 = LINE/P5, P6
L5 = LINE/P6, P1
L6 = LINE/P1, P2
CUTTER/.5
FEDRAT/7,IPM
LOADTL/1,SETOOL,0,0,3
SPINDL/1000,CLW
COOLNT/FLOOD
```

FROM/SETPT
GO/TO, L1, TO, PS, ON L6
GORGT/L1, TO, L2
GORGT/L2, TANTO, C1
GOFWD/C1, TANTO, L3
GOFWD/L3, PAST, L4
GOLFT/L4, PAST, L5
GOLFT/L5, PAST, L6
GOLFT/L6, PAST, L1
GOTO/SETPT
SPINDL/OFF
COOLNT/OFF
END
FINI

Appendix B:

Example Coordinate Sheet

Sheet 1 of 2

TOOL OFFSET COORDINATE TABLE

Customer ABC Manufacturing

Tool Radius .375

Table Prepared By WEN Date 8-19-60

Part No.
1 4 6 1 7

Roughing _____

Finishing X

Pt. No.	X	Y	Z	Pt. No.	X	Y	Z
0	5.000	10.000		18	10.300	12.763	
1	9.400	10.800		19	9.868	12.390	
2	9.906	10.937		20	9.828	12.016	
3	10.000	9.312		21	9.868	12.150	
4	8.947	8.780		22	9.968	12.018	
5	8.947	8.750		23	7.093	11.969	
6	8.947	7.125		24	9.896	11.328	
7	9.962	7.125		25	10.657	11.154	
8	7.437	8.344		26	9.907	10.938	
9	10.000	8.344		27	12.341	11.500	
10	10.000	9.311		28	9.855	11.260	
11	10.000	8.344		29	9.108	11.188	
12	12.563	8.344		30	9.804	11.467	
13	10.308	7.125		31	10.500	11.750	
14	11.035	7.125		32	9.868	12.154	
15	11.035	10.000		33	9.154	12.780	
16	10.937	11.923		34	10.093	11.562	
17	10.000	11.875		35	8.791	11.122	

Appendix C-1

2-1/2D Mill Program

```
%
O00010
(DRAWING: BADGE)
(T01,H01 = FLY CUTTER)
(T02,H02 = 3/16" END MILL)
(T03,H03 = 1/8" DRILL)
G91 G28 Z0
G17 G20
G40 G49
G80 G90
T01
M06
G43 H01
(FACE)
S2000 F10. M03
M08
G00 X-0.8 Y0.27
Z0.1
G01 Z0
X4.8
Y1.73
X-0.8
G00 Z0.1
T02
M05
M09
G91 G28 Z0
G40 G49
G80 G90
M01
M06
G43 H02
(POCKET)
S2000 F10. M03
M08
G00 X-0.15 Y.9688
Z0.1
G01 Z-0.045
X.5245
G02 X.524 Y.969 I.666 J.031
G01 X.635
G02 I0.6655 J0.0312
G01 X0.635
G02 X0.8632 Y1.4497 I0.555 J0.0312
I0.1139 J-0.1567
X1.5168 I0.3268 J-0.4497
I-0.1139 J-0.1567
X1.7187 Y0.8282 I-0.3268 J-0.4497
I-0.1843 J0.0599
X1.19 Y0.4441 I-0.5287 J0.1718
Y0.8316 I0 J0.1938
G01 Y1.
Y0.8316
G02 Y0.4441 I0 J-0.1937
X0.6613 Y0.8282 I0 J0.5559
I0.1843 J0.0599
X0.635 Y1.0312 I0.5287 J0.1718
G01 X-0.15
G00 Z0.1
X4.15
G01 Z-0.045
G01 X3.4755
G02 I-0.6655 J-0.0312
G01 X3.365
G02 X3.3387 Y0.8282 I-0.555 J-0.0312
I-0.1843 J0.0599
X2.81 Y0.4441 I-0.5287 J0.1718
Y0.8316 I0 J0.1938
G01 Y1.
Y0.8316
G02 Y0.4441 I0 J-0.1937
X2.2813 Y0.8282 I0 J0.5559
I0.1843 J0.0599
X2.4832 Y1.4497 I0.5287 J0.1718
I0.1139 J-0.1567
X3.1368 I0.3268 J-0.4497
I-0.1139 J-0.1567
X3.365 Y0.9688 I-0.3268 J-0.4497
G01 X4.15
G00 Z0.1
X2.2726 Y.4663
G01 Z-0.045
X2. Y0.1938
X1.7274 Y.4663
G00 Z0.1
T03
M05
M09
G91 G28 Z0
G40 G49
G80 G90
M01
M06
G43 H03
(DRILL)
S2000 F5. M03
M08
G00 X0.2202 Y1.
Z0.1
G01 Z-0.6
```

G00 Z0.1

X3.7798

G01 Z-0.6

G00 Z0.1

M05

M09

G91 G28 Z0

G40 G49

G80 G90

M30

%

Appendix C-2

2D Torch Program

G70
G90
G92 X0.Y0.
G00 X7.25 Y5.56
M67
M70
G01 Y4.19
G02 X7. Y3.94 I7. J4.19
X7. Y3.94 I7. J5.56
M73
M69
G00 X18.87 Y5.56
M67
M70
G01 Y4.19
G02 X18.62 Y3.94 I18.62 J4.19
X18.62 Y3.94 I18.62 J5.56
M73
M69
G00 X30.5 Y5.56
M67
M70
G01 Y4.19
G02 X30.25 Y3.94 I30.25 J4.19
X30.25 Y3.94 I30.25 J5.56
M73
M69
G00 X42.12 Y5.56
M67
M70
G01 Y4.19
G02 X41.87 Y3.94 I41.87 J4.19
X41.87 Y3.94 I41.87 J5.56
M73
M69
G00 X53.75 Y5.56
M67
M70
G01 Y4.19
G02 X53.5 Y3.94 I53.5 J4.19

X53.5 Y3.94 I53.5 J5.56
M73
M69
G00 X65.37 Y5.56
M67
M70
G01 Y4.19
G02 X65.12 Y3.94 I65.12 J4.19
X65.12 Y3.94 I65.12 J5.56
M73
M69
G00 X77. Y5.56
M67
M70
G01 Y4.19
G02 X76.75 Y3.94 I76.75 J4.19
X76.75 Y3.94 I76.75 J5.56
M73
M69
G00 X88.62 Y5.56
M67
M70
G01 Y4.19
G02 X88.37 Y3.94 I88.37 J4.19
X88.37 Y3.94 I88.37 J5.56
M73
M69
G00 X1.79 Y0
M67
M70
G01 X3.07 Y1.28
G03 Y1.63 I2.89 J1.45
G02 X12.56 Y5.56 I7. J5.56
G01 X12.81 Y5.54
X13.06 Y5.56
G02 X24.19 I18.62 J5.56
G01 X24.44 Y5.54
X24.69 Y5.56
G02 X35.81 I30.25 J5.56
G01 X36.06 Y5.54

X36.31 Y5.56
G02 X47.44 I41.87 J5.56
G01 X47.69 Y5.54
X47.94 Y5.56
G02 X59.06 I53.5 J5.56
G01 X59.31 Y5.54
X59.56 Y5.56
G02 X70.69 I65.12 J5.56
G01 X70.94 Y5.54
X71.19 Y5.56
G02 X82.31 I76.75 J5.56
G01 X82.56 Y5.54
X82.81 Y5.56
G02 X82.81 Y5.56 I88.37 J5.56
G01 X82.56 Y5.58
X82.31 Y5.56
G02 X71.19 I76.75 J5.56
G01 X70.94 Y5.58
X70.69 Y5.56
G02 X59.56 I65.12 J5.56
G01 X59.31 Y5.58
X59.06 Y5.56
G02 X47.94 I53.5 J5.56
G01 X47.69 Y5.58
X47.44 Y5.56
G02 X36.31 I41.87 J5.56
G01 X36.06 Y5.58
X35.81 Y5.56
G02 X24.69 I30.25 J5.56
G01 X24.44 Y5.58
X24.19 Y5.56
G02 X13.06 I18.62 J5.56
G01 X12.81 Y5.58
X12.56 Y5.56
G02 X3.07 Y1.63 I7. J5.56
M73
M69
G00 X0.Y0.

Appendix C-3

Turning Program

%	Z0.1
O00025	X1.4
(DRAWING NAME: DGS-0018)	G01 Z-0.565
(T909 = .125" WIDE CUT OFF)	G00 X1.5 Z0.1
(T202= 1/64"R PROFILE TOOL)	X1.3
(T101= 18TPI THREAD INSERT)	G01 Z-0.565
(T1010=14TPI NPT THREAD)	G00 X1.4 Z0.1
(T707= CENTER DRILL)	X1.2
(T404 = 19/64" DRILL)	G01 Z-0.565
G18 G20 G40 G54 G64	G00 X1.3 Z0.1
G80 G90 G96 G99	X1.1
T909	G01 Z-0.565
(FACING/THREAD RELIEF)	G00 X1.2 Z0.1
G50 S1200	X1.
G96 S250	G01 Z-0.565
F0.005	G00 X1.1 Z0.1
M03	X0.9
M08	G01 Z-0.565
M31	G00 X1. Z0.1
G00 Z0.125	X0.8
X1.5	G01 Z-0.565
G01 X0	G00 X0.9 Z0.1
G00 X1.5	X0.7
Z-0.45	G01 Z-0.565
G01 X0.48	G00 X0.8 Z0.1
G00 X1.5	X0.6
Z-0.5	G01 Z-0.565
G01 X0.48	G00 X0.7 Z0.1
G00 X1.5	X0.55
Z-0.538	G01 Z-0.565
G01 X1.193 Z-0.5	G00 X0.6 Z0.1
G00 X1.5	X0.48
Z-1.075	G01 Z-0.225
G01 X1.12	G00 X0.55 Z0.1
G00 X1.5	X0.408
Z-1.037	G01 Z-0.063
G01 X1.193 Z-1.075	X0.48 Z-0.125
G28 U0	G00 Z0.1
G28 W0	X0
M01	G42 G01 X0.179
T202	X0.48 Z-0.115
(PROFILE)	Z-0.225
G96 S400	G00 G40 X1.5
F0.012	Z-1.23
M03	G01 X1.12 F0.003
M08	G00 X1.4
G00 Z0.1	G01 Z-3.275
Z0	G00 X1.5 Z-1.23
X1.5	X1.3
G01 X0	G01 Z-3.275

G00 X1.4 Z-1.23
X1.2
G01 Z-3.275
G00 X1.3 Z-1.23
G01 X1.12
X1.1 Z-1.35
Z-3.275
G00 X1.2 Z-1.23
G01 X1.12
X1.119
X1.02 Z-1.99
Z-3.275
G00 X1.1 Z-1.94
G01 X0.925
G28 U0
G28 W0
M01
T101
(STRAIGHT THREAD)
G97 S1100
F0.01
M03
M08
G00 Z-0.175
X1.75
X0.55
G76 X0.485 Z-0.5 K0.033 D0.015 F0.055556
G28 U0
G28 W0
M01
T1010
(TAPER THREAD)
G97 S1100
F0.01
M03
M08
G00 Z-1.044
X1.75
G01 X1.5
G00 X1.75
G00 Z-1.2
X1.75
G01 X1.2
G76 X0.925 Z-1.95 K0.05 D0.018 I0.0469
F0.071429
G28 U0

G28 W0
M01
T707
(CENTER DRILL)
G97 S1800
F0.005
M03
M08
G00 Z0.1
X0
G01 Z-0.2
G00 Z0.1
G28 U0
G28 W0
M01
T404
(DRILL)
G97 S1400
F0.008
M03
M08
G00 Z2.
X0
Z0.5
Z0.1
G83 Z-2. Q0.2 R0.1 F0.004
G00 Z0.1
G28 U0
G28 W0
M01
T909
(CUT OFF)
G96 S250
F0.005
M03
M08
M33
G00 Z-1.9
X1.75
G01 X0
G28 U0
G28 W0
G105
(PUSH OUT DISTANCE = 1.91)
M99
M30

Appendix D

PREDATOR_MILL.TMP File

Tool Library

T1 = 1/16" Flat End Mill

T2 = 1/8" Flat End Mill

T3 = 3/16" Flat End Mill

T4 = 1/4" Flat End Mill

T5 = 3/8" Flat End Mill

T6 = 1/8" Twist Drill

T7 = 3/16" Twist Drill

T8 = 1/4" Twist Drill

T9 = 3/8" Twist Drill

T10 = 3/32" Center Drill

PROGRAM START:

%

O??

G91 G28 Z0

G54 G20 G40

G49 G80

T?? M6

G43 H??

S?? F??. M3

M8

PROMPT CHANGE:

T??

M9

M5 G91 G28 Z0

G40 G49 G80 G90

M6

G43 H??

S?? F??. M3

M8

PROGRAM END:

M9

G00 G80 Z15. M5

G91 G28 Z0

G40 G49 G90

M30

%

!!