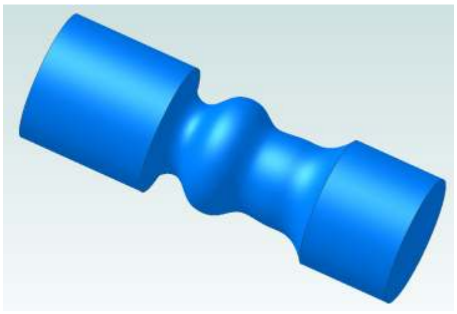
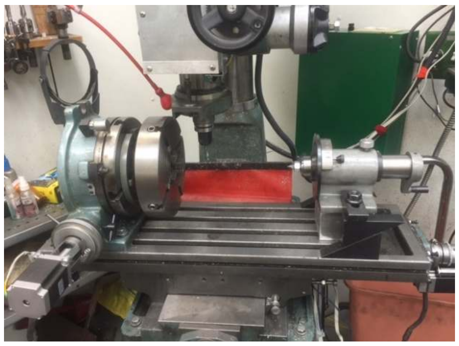
2 ¾ D Machining On a 4 Axis RF-30 Mill/Drill, version 1.4

By R. G. Sparber



It would not be hard to make this part with a 5 axis screw machine and the related 3D software tools. The workpiece is fixtured so it can be turned. The cutter can be both spun and oriented. The result is an amazingly versatile (and expensive) machine.



My challenge was to machine such parts with standard 2 ½ D software tools and a 4 axis Computer Numerical Control (CNC) driven mill/drill.

I have X, Y, Z, and A axis control. My A axis lets me rotate the workpiece. Unlike a 5 axis mill, I am unable to machine features like undercuts or angled holes without changing the fixturing.

What is Probably New?

Very likely… nothing.

What Might Be Unusual?

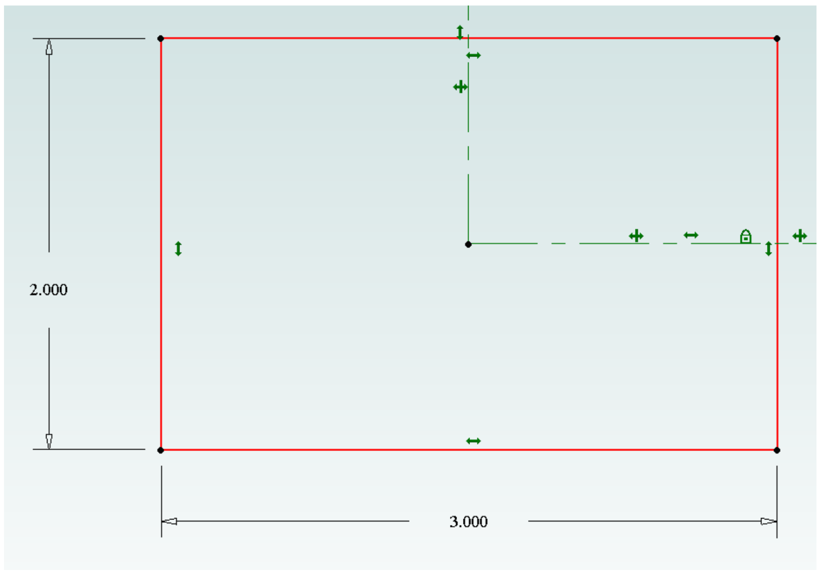
By adding a single wire to my CNC control box, I am able to drive the A axis in two different ways. I can index this axis in the standard way using the A*ddd* command where *ddd* is the angular position in degrees. But I can also make the A axis continuously rotate by using the spindle control command S*rpm* where *rpm* is a number that corresponds to the RPM of the A axis. Once the A axis is set to turn, no A commands are needed in the subsequent command lines. I end up with mill capabilities similar to a CNC lathe. The software manipulation part of this came

[1] You are free to distribute this article but not to change it.

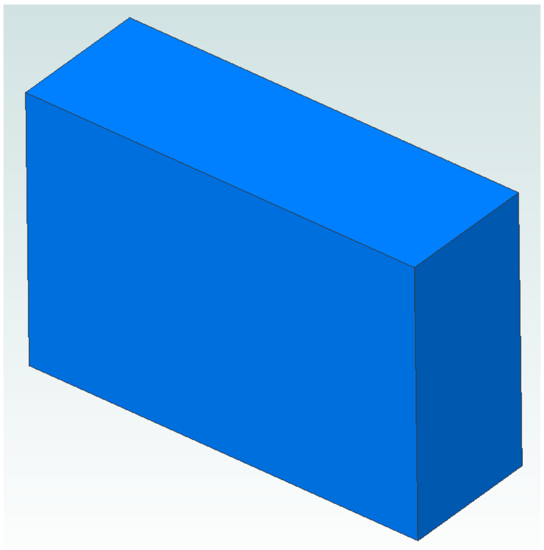
from "Martin who lives is Leasingham". This dual control of the A axis should be independent of the software although it was only tested with Mach3.

2 ½ D

In CNC there is a concept called "2 ½ D". Let me explain it first.

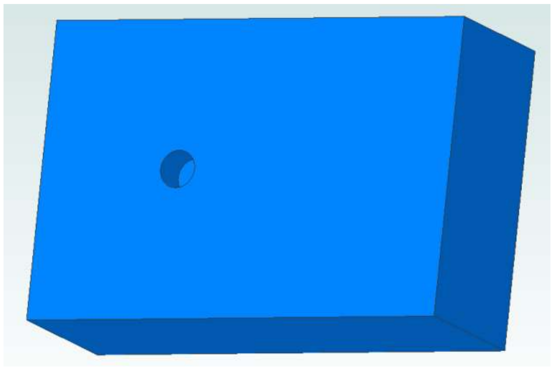


I have drawn a 2 by 3 rectangle on a XY plane. It is a two dimensional object: 2D.

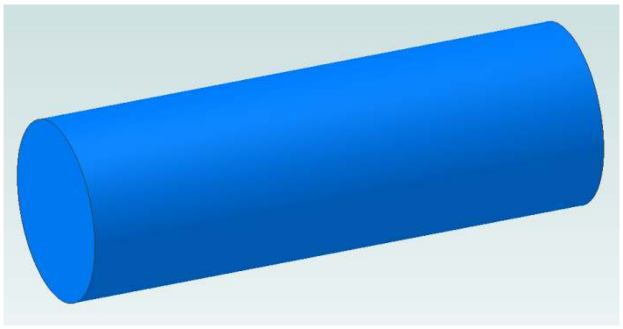


By "extruding" my rectangle along the Z axis, I have created a 1 by 2 by 3 rectangular solid. Although it really is a three dimensional object, notice that my Z dimension is uniform. I can define this object by first setting the thickness and then deal with a 2D object. For this reason, we call objects that can be extruded "2 ½ D".

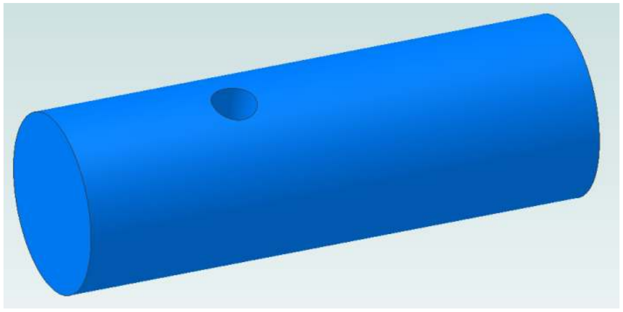
2 ½ D objects are easily machined on a mill using 3 axes. First I would set the 1 dimension using the Z axis. Then I could drive the X and Y axes to define my 2 by 3 surface.



After machining all 3 dimensions of the block, I can mill a hole. The hole is at a specified XY location and its depth is set by changing the Z axis. So this is also 2 ½ D.



Here I have drawn a circle of diameter 1 and extruded it into a cylinder of length 3. This too can be considered a 2 ½ D object.



The command needed to mill this hole is identical to what was used on my block.

But these two cases are not identical. With my block, the hole could be milled after moving around the XY plane. With the cylinder, I would first machine the outside diameter and then lay it down. Repositioning the workpiece, in my humble opinion, takes us from 2 ½ D to a ***slightly*** higher dimension.

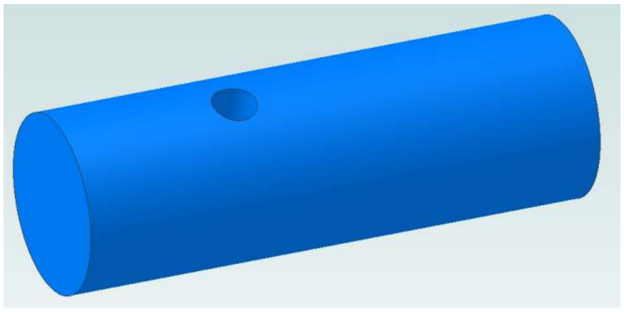
If I was running a 5 Axis Screw Machine, then the part would not need to be repositioned. My cutter would be moved within its 5 axes to mill this hole.

So what do we call it when a workpiece is machined 2 ½ D, repositioned, and again machined 2 ½ D? At least within this article, I will call it "**2 ¾ D**".

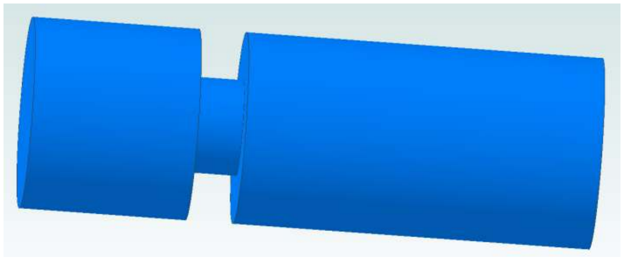
In the above example, I manually repositioned the workpiece. My question also applies when a 4th axis is moving the part.

2 ¾ D

Manually repositioning the cylinder is one example of 2 ¾ D. But the broader case is setting either a new position or a new ***velocity***.



Say I leave the spinning cutter down in this hole and start to rotate the part. My rotational velocity went from zero to a finite number.



I get a reduced diameter section as if the part was on my lathe and a parting tool was used. However, I'm using the side and bottom of my end mill here and running on the centerline of the part.

The standard way to do this operation is to specify the depth of the cut and the number of degrees of rotation. This causes the end of the cutter to spiral down to its final diameter. Then we go another full revolution to insure uniform diameter. For example,

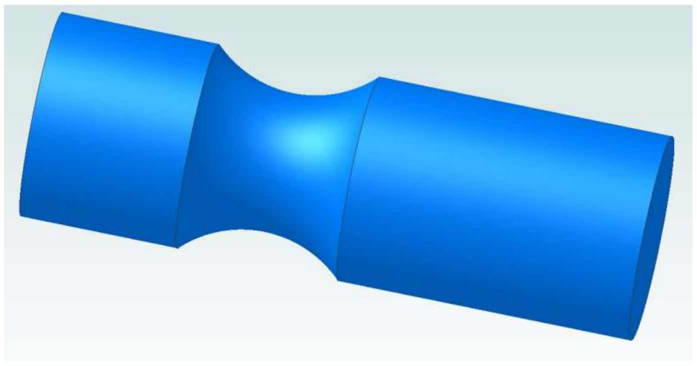
```
G0 Z 2 (raise the end mill off the part)
G0 A0  (rotate the part to 0°)
G1 F1 A360 Z0.25  (rotate the part 1 rev while lowering the end mill to 0.25")
G1 A720  (with the end mill at 0.25 inches, turn 1 more rev)
```

An alternate approach is to hack a line of 2 ½ D code:

```
G0 Z2 (raise the end mill off the part)
(start rotation)
G1 F1 Z0.25  (lower the end mill down to 0.25")
(pause to allow one full rotation)
```

This says to start the cylinder rotating and keep it rotating. Then feed the cutter down to its final depth. When done, pause to insure the cutter visits the full perimeter of the part. My A DRO does not increment.

In other words, I can use my CAM software tools to define a profile and then add a few lines of code manually to shape cylinders.



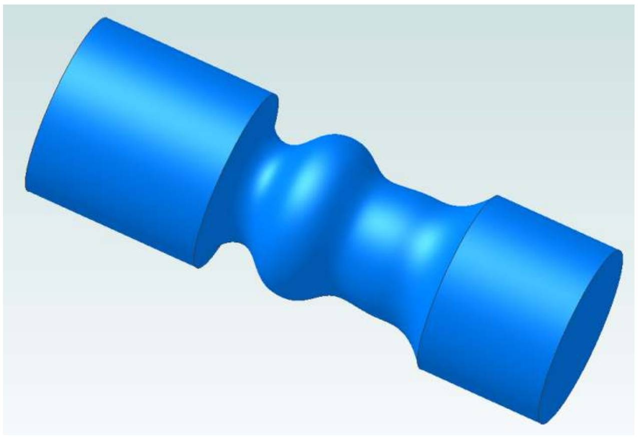
Look what I get when I move the cutter along the Y axis so it is side milling. The cutter is lowered so its bottom is just below the center of rotation of the part.

My XY movement defines a radius. Since the part is turning continuously, I get this saddle.

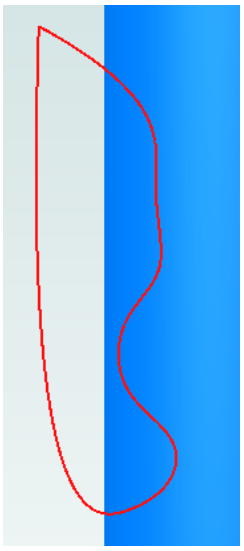
The saddle can be cut by adding an index command to each line of code that defines the arc.

Using my alternate approach, I start by drawing a 2D radius and end up with a saddle. More on this later.

OK, so much for boring shapes.



Drawing in 2D provides almost limitless flexibility. No need to just draw a radius, I can draw any shape line as long as it does not have undercuts[2].



First I drew a freehand 2D figure. In my drawing program[3] I used Revolve Cut to create the rendering shown above. But in order to generate the complex g-code, I simply treat this as a 2D object cut into a rectangular block. The longitudinal axis of the block is the center of rotation of my cylinder. The thickness of the block equals the radius of my workpiece.

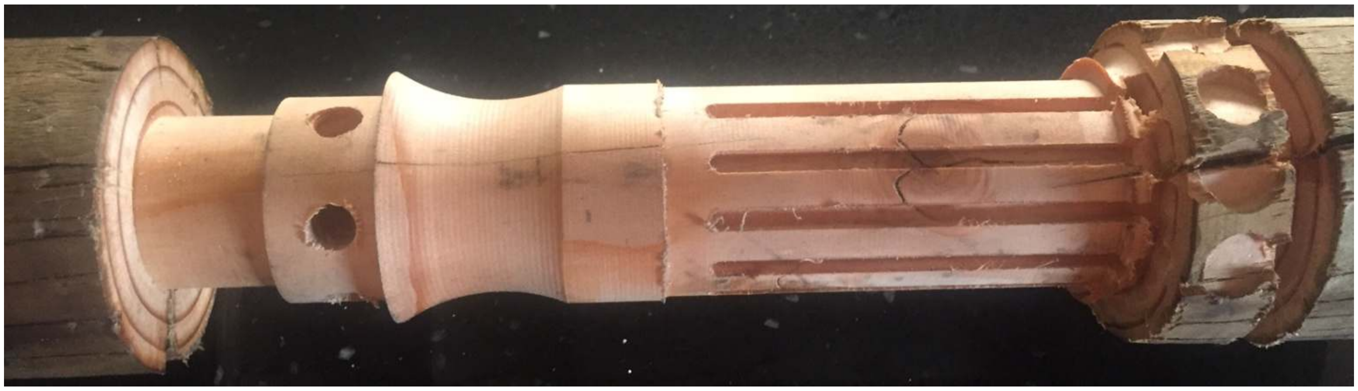
I add a continuous rotate command in front of the g-code and bracket it with pause commands to insure the ends are square.

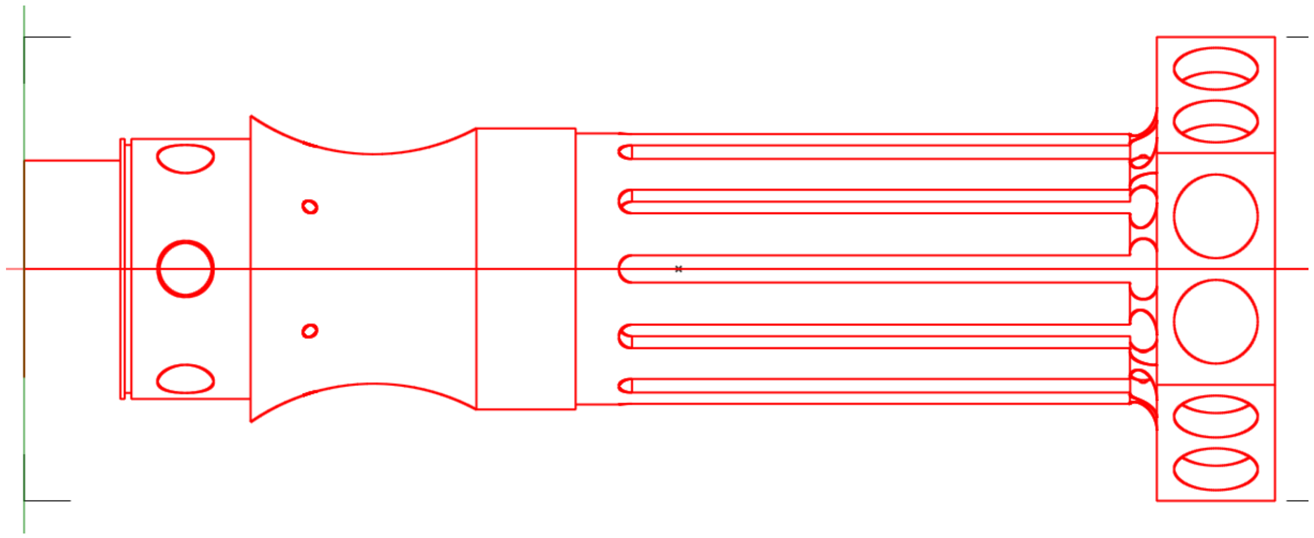
If I used the conventional rotate format, I would need to add an index command to each line of code. Given the complex shape of this figure, that would be a tall order. Of course, you could buy 3D CAM software to do this task automatically. I can't justify the added expense but this is common in commercial shops.

So far it is all talk, how about some action?

[2] An undercut cannot be side milled.
[3] I use Alibre PE which has since been rebranded as Geomagic Design™.

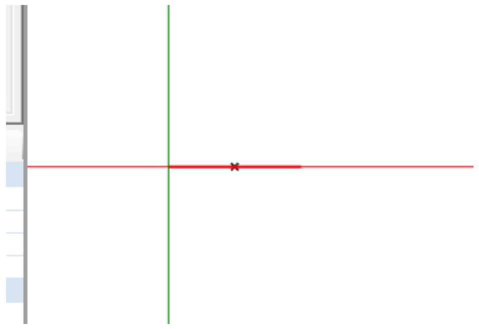
Shop Experience





Here is the result of a test run that employs both side milling and end milling. I am machining an old wooden stake in order to find my programming errors. Plenty of bugs to be seen here but you can also see some sharp corners and a nice contour.

Below the picture is the view from CamBam. Notice that in the lower left corner it appears that a horizontal line is missing. It has been moved to be on the centerline and becomes the prototype for all cylindrical machining.

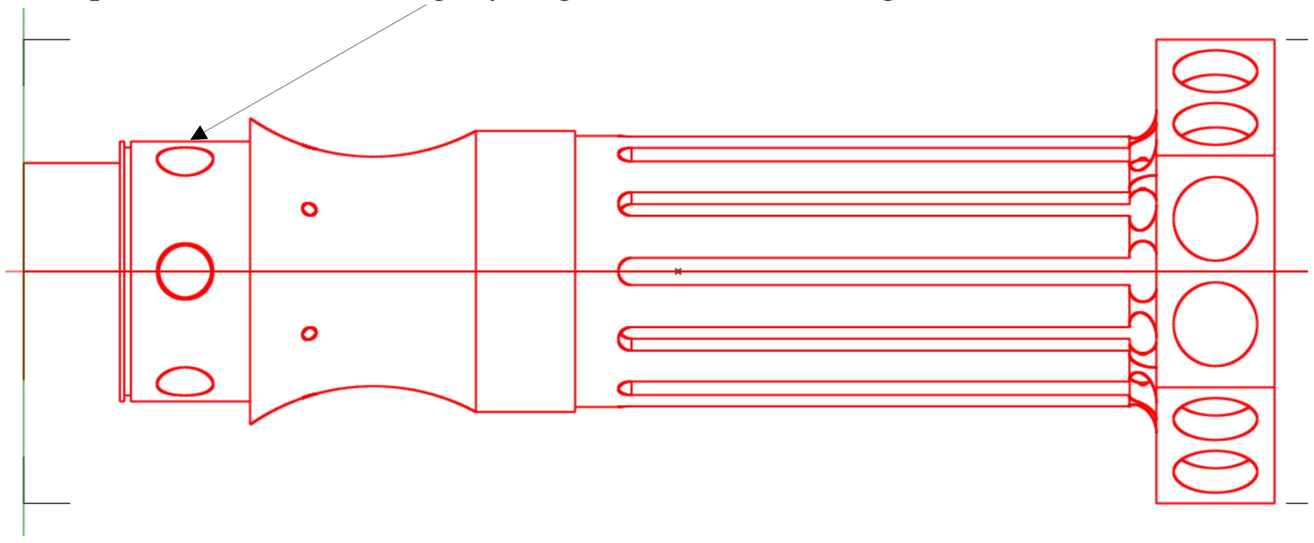


Starting on the left, I run my ¼ inch end mill along the centerline. The g-code generated from the relocated line is my starting point.

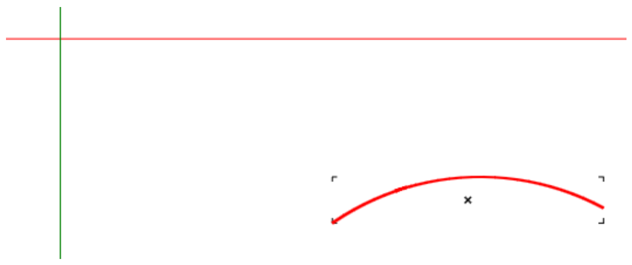
```
G0 Z1.25 (Z raised to safe plane)
M3 S5000 (sets A axis to about 2 RPM)
G0 X-.125 Y-0.0  (position 1/4 inch end mill to origin offset by dia on X axis and
at OD plus 1/8 inch; run on center line)
G1 F1.0 Z0.925 (feed down 0.2 inches from OD)
G4 P45 (hold position for more than 1 revolution to square up start shoulder)
G1 F0.2 X0.309 (rough feed to end of line but move to 0.444 - 0.125 - 0.01 to
compensate for end mill diameter)
G4 P45 (hold position for more than 1 revolution to square up end shoulder)
```

By changing my X start and end points, I can define the location of the cylinder being machined. Z controls its radius.

A pause at the start and end gives me the uniform shoulders. Then I raise the cutter up a little and cut the slightly larger diameter following section[4].



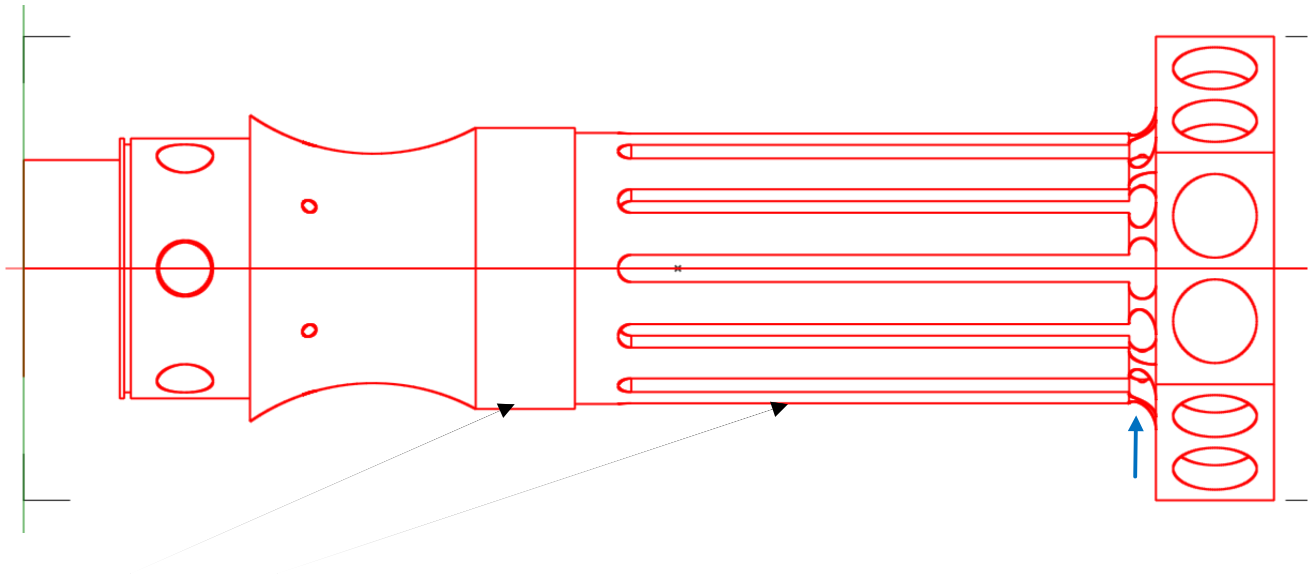
[4] The narrow slot will be cut manually on my lathe.



Next is the saddle. This requires me to move the cutter off of the centerline and start side milling. With the bottom of the cutter slightly lower than the center or rotation, I simply follow this 2D curve.

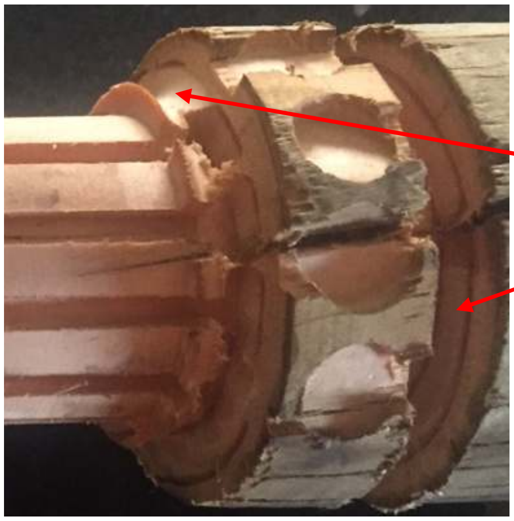
Here is where things start to get interesting. The g-code for this curve already includes the fact that I am side milling with a ¼ inch end mill. I feed down 0.2 inches per pass. You can see, below, that many lines of code are needed to define this curve. None of these lines are modified. My added lines are in red. If the workpiece was not continuously turning, I would need to have an A*ddd* command on each line where *ddd* is the number of degrees moved during the this line segment. In place of the two lines I have added would be a command to turn one full revolution with the cutter at the last position as shown on page 4.

```
( polyline 7 roughing )
G0 X1.1211 Y-0.8163 ( 1st start of poly 7)
G0 Z1.1875 (prepare to feed down)
G1 F1.0 Z0.925 (feed down for first pass)
G4 P45 (pause at start of line to cut clean start circumference)
G2 F0.2 X1.3313 Y-0.7102 I0.4924 J-0.7141 (moving along poly 7 as A axis turns
at about 2 RPM)
G3 X1.3372 Y-0.708 I-0.0436 J0.1267
G1 X1.3391 Y-0.7073
G3 X1.3405 Y-0.7067 I-0.0496 J0.1245
G3 X1.3453 Y-0.7048 I-0.0457 J0.126
G3 X1.3509 Y-0.7029 I-0.0412 J0.1275
G3 X1.3524 Y-0.7024 I-0.0392 J0.1281
G1 X1.3584 Y-0.7005
G3 X1.3593 Y-0.7002 I-0.0407 J0.1277
G3 X1.3636 Y-0.6989 I-0.0357 J0.1292
G3 X1.3643 Y-0.6988 I-0.0328 J0.1299
G1 X1.3727 Y-0.6966
G1 X1.3772 Y-0.6955
G1 X1.3786 Y-0.6952
G3 X1.3848 Y-0.6937 I-0.0292 J0.1308
G2 X2.0236 Y-0.7661 I0.2287 J-0.8368
G4 P45 (pause at end of line to cut clean end circumference)
```



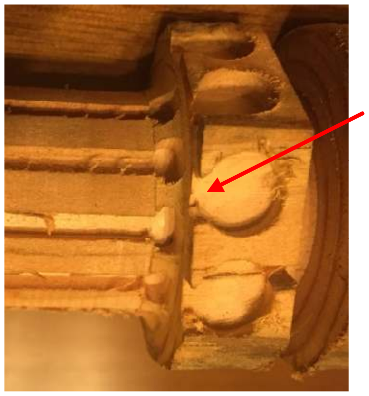
A third and forth uniform diameter comes next so I move my cutter back to the centerline and start above the part.

At the end of the forth uniform section is a fillet (blue arrow). The end mill is again moved off center so I am side milling as was done at the saddle.

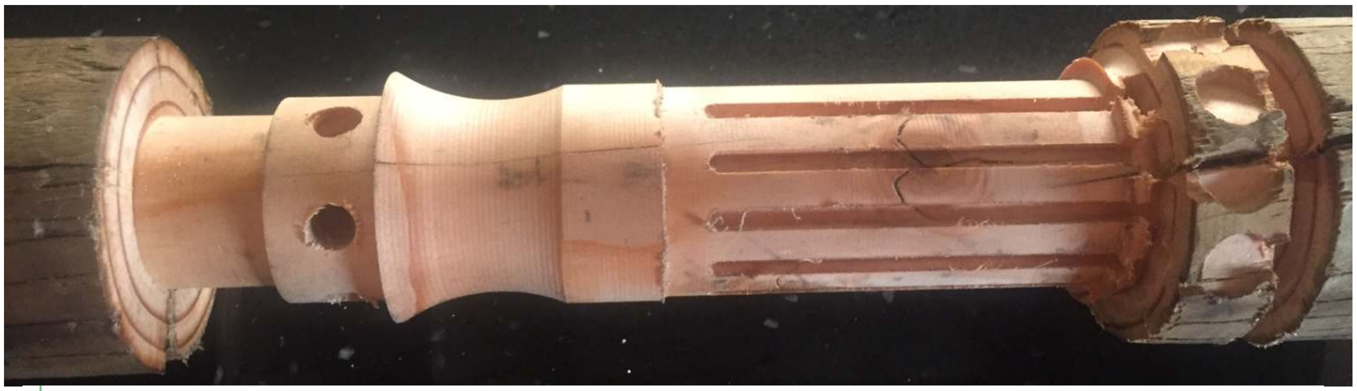


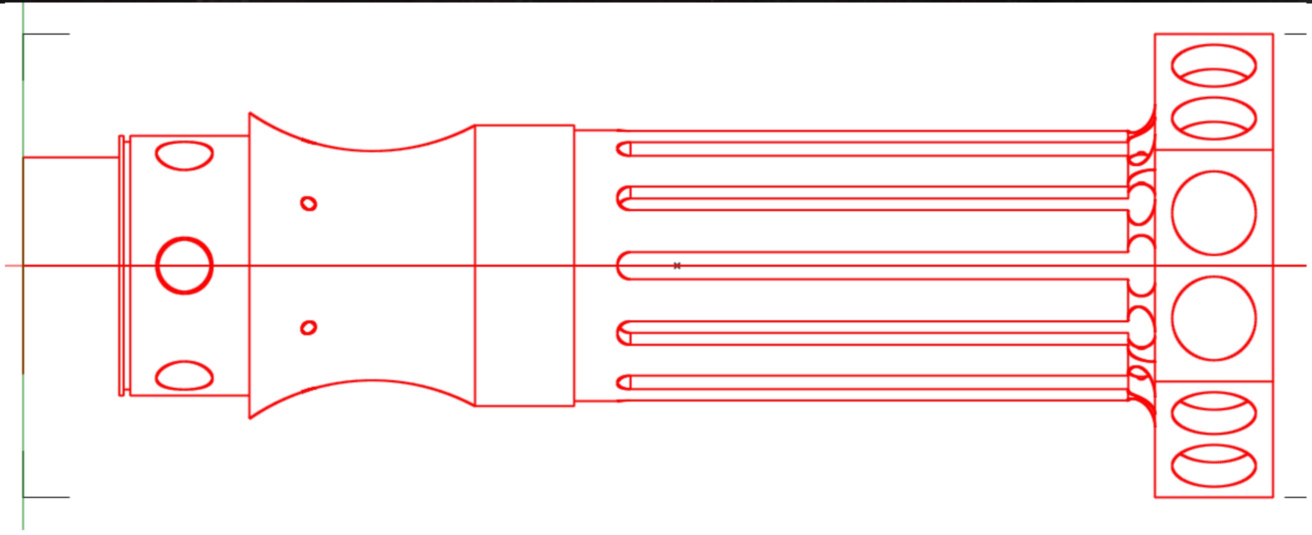
The fillet is supposed to transition from the forth uniform diameter to the side wall of the largest diameter.
An error in both X and Y caused me to miss that mark.

The final cut with the part continuously rotating is on the far right. It defines the right face of the largest diameter and was cut with the end mill back on the centerline. I cut down to a 1 inch diameter.



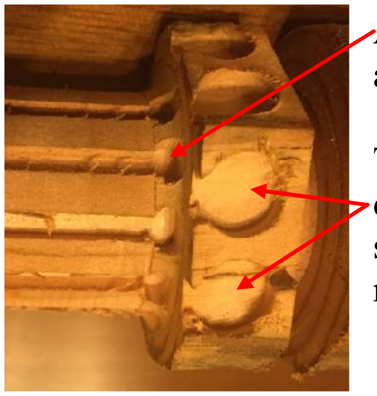
On my second try I fixed the fillet but put it in the wrong place. Note that the pockets break through on the left side. Two steps forward and one back.





After executing an M5 command which stops both the spindle and the continuous A axis rotation, I changed to an 1/8 inch end mill. Then I executed a M3 S0 command to stop A axis continuous rotation but keep the cutter turning. G0 A0 moves me to zero degrees.

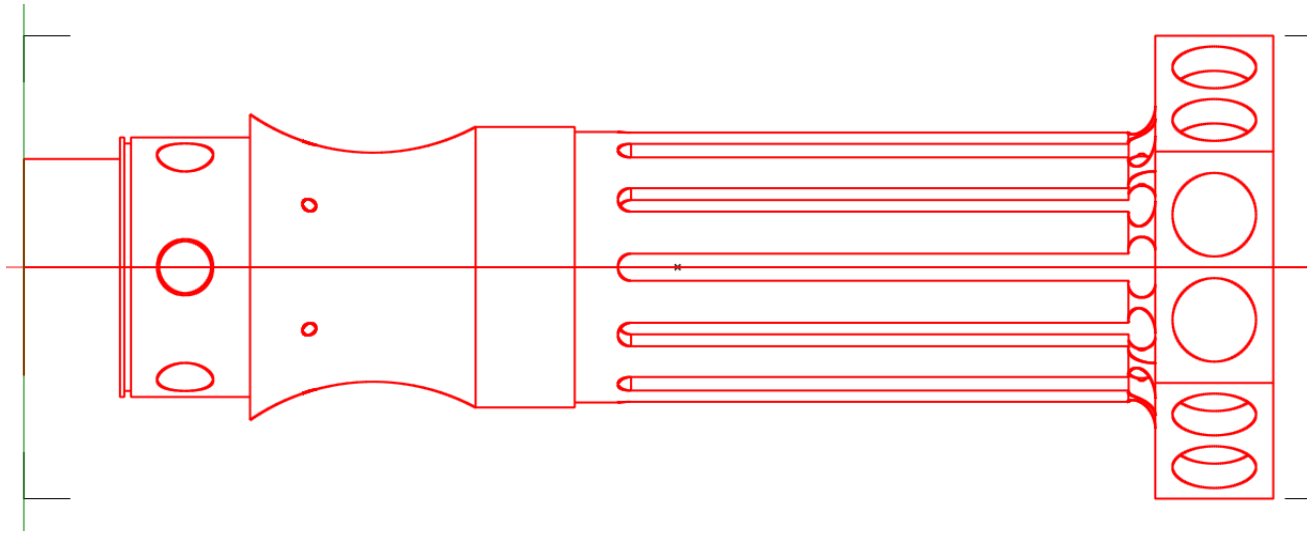
With the cutter at the centerline, I cut two concentric holes on the left and the slot in the middle.



At the right end of the slot, I move forward and backwards along the Y axis.

The cutter is then raised enough to get on top of the large diameter on the right. Then it milled two pockets that are symmetric around the centerline. These operations are repeated every 60 degrees[5].

[5] 0, 60, 120, 180, 240, and 300 degrees.



With the cutter out of the way, I execute an A30 which indexes the part to 30 degrees. Then I repeat the slot cutting operations. These operations are also repeated every 60 degrees[6].

In all cases, I first deal with the A axis and then machine as if I was doing 2 ½ D.

The file is 44K bytes but most of it is copy and paste.

[6] 30, 90, 150, 210, 270, and 330 degrees.

Advise on Generating Code

Since this is semiautomatic g-code generation, errors can easily creep in. It helps to have a shop drawing in front of you and label each feature both on the drawing and in the code.

For each end mill feature, I verify the following:

- Y at the centerline
- Start and end of path along X axis assuming no tool offset. Then think about how to offset the end mill at the start and end. If the feature has a larger diameter than the last feature, you can set zero tool offset at the start. If the feature has a smaller diameter than the last feature, you must shift the tool over so you do not start by milling the past feature. The same logic applies at the end of the path.
- Down feed rate
- Depth of cut
- Total side milling feed rate sets both the A axis RPM and X feed rate:

$$\boldsymbol{RPM\ of\ A\ axis = \frac{f}{3D}} \qquad \textbf{(1)}$$

$$\boldsymbol{X\ axis\ feed\ rate = RPM\ of\ A\ axis\ \times c} \qquad \textbf{(2)}$$

Where

$\boldsymbol{f}$ is the desired feed rate experienced by the cutter

$\boldsymbol{D}$ is the diameter of the workpiece at the cut

$\boldsymbol{c}$ is the radial depth of cut

See http://rick.sparber.org/TTFR.pdf for details.

- Pause at start and end of path to allow a full revolution of the part.

For each side mill feature, I verify the following:

- Down feed rate
- Depth of cut. Although you are side milling a cylinder, you can treat it as if it were a block.
- Total side milling feed rate sets both the A axis RPM and X and Y feed rates:

$$\boldsymbol{RPM\ of\ A\ axis = \frac{f}{3D}} \qquad \textbf{(1)}$$

$$\boldsymbol{X\ and\ Y\ axis\ feed\ rate = RPM\ of\ A\ axis\ \times c} \qquad \textbf{(2)}$$

Where

$\boldsymbol{f}$ is the desired feed rate experienced by the cutter

$\boldsymbol{D}$ is the diameter of the workpiece at the cut

$\boldsymbol{c}$ is the radial depth of cut

- See http://rick.sparber.org/TTFR.pdf for details. The finish cut should be at maximum RPM and the feed rate should be set to leave an acceptably small ripple.
- Pause at start and end of path to allow a full revolution of the part.

X and Y movement is defined by a block of g-code generated by a CAM program. No changes should be needed. The code should include the correct tool offset.

The Hardware Hack

You may be wondering about this continuous rotate command. I have a Variable Frequency Drive (VFD) on my spindle motor. RPM is set with a dial within easy reach. I never felt the need to set the RPM using g-code. I just use the code to turn the spindle on and off. This is done with the M3 and M5 commands.

Mach3 permits me to define a spindle motor as using a Pulse/Direction interface. In other words, the spindle motor would be controlled just like my axis stepper motors. I then wire the spindle's Pulse output to my A axis stepper driver's Pulse input[7]. I also have a connection from my A axis output to the driver's Pulse input. This unorthodox arrangement gives me the ability to free rotate or index the A axis.

To free rotate the A axis at 1 RPM, I execute

M3 S2300

M3 says to turn on the spindle. It now also starts pulses to be sent to my A axis driver. The S2300 defines the rate at which pulses are sent. Empirically I found that 2300 pulses per minute translates to about 1 RPM.

For the full story, see http://rick.sparber.org/CNCHW.pdf starting at page 18.

[7] See the Appendix for a purely data way to make this connection if you are using Mach3.

Acknowledgments

Thanks to Martin from Leasingham for opening my eyes to this approach and reviewing the article.

Thanks to Dan Mauch for his data only approach.

I welcome your comments and questions.

If you wish to be contacted each time I publish an article, email me with just "Article Alias" in the subject line.

Rick Sparber
Rgsparber.ha@gmail.com
Rick.Sparber.org

Appendix

Dan Mauch sent me this approach that does the same thing with just data that I did with both data and a wire: dual control of the A axis. However, he has sorted out the S command so it directly shows RPMs.

"I have a simpler method for controlling the indexing and RPM.

- In Config>ports and Pins>motor outputs for both the A axis and Spindle:
  - For motor output set pin 8
  - For direction set pin 9
  - use port 1
- In Spindle setup put check marks on
  - use spindle motor output and
  - step and direction.
- In General Config
  - Set A axis to Angular and
  - have a check mark in G92.1.
- In motor tuning we need to set steps per degree. Given that you know the steps per revolution for the A axis, calculate the steps per degree: $\frac{steps\ per\ revolution}{360\ degrees} = SPD$. Set the A axis to SPD steps per degree. For example, I have a 2000 steps per rev driver/motor combo $\frac{2000}{360} = 5.55555$. In motor tuning under spindle set 20000 with a velocity of 60. Not sure why I have it there but is works. If I enter M3 S10 I get 10 RPM.

In Mach3 if you use the M3 command it will not increment the A Axis DRO. However, you will see the A axis RPM under spindle RPM.

Sometimes you need to set the A Axis DRO from within a G code tool path. I use the g92 command."