The program spurgear.ngc, written entirely in G-code with no assistance from any CAD program, is intended to aid in the design of a straight cut gear with involute shaped teeth. The program could be used to cut a workable prototype out of a flat sheet of stock assuming the user made the teeth large enough to cut with a practical end-mill. All the involute curve math is contained in the program and there's not a lot . A 5/64 end-mill will cut a surprisingly large number of gear shapes, modify the parameters, enable cutter compensation in the program and if no errors are detected it should cut yours.

Here's a screen shot of the program in action. The debug statements execute and print out the tooth specifics at run time so it is necessary to hit the run and pause button a few times to get it executing. The program actually computes some of the values as it draws the part so I print the values after the first tooth is complete.

The variable named precision controls the number of samples taken from the involute curve, .1 means every tenth of a degree of involute curve is used and should result in an accurate gear.

The surface cut in aluminum can be a little rough and that's not much of a problem since the gear rolls against its partner. The variable fudgewidth can be used to allow a final path to "polish" the rolling surface if carefully done (do one full depth pass around the part cutting full depth on the side of the end-mill after completing the plunge cuts)

Mostly the program is an exercise in g-code programming. You don't need any other software to make a gear and the code can be modified for other purposes you may need.

This is very much a work in progress, I had originally planned on using subroutines to make only the involute teeth of the gear. The way the program evolved, it seems that it would be easy to make a subroutine that makes a whole gear and use the interpreter to display a gear train instead of a single gear in order to detect problems with meshing. The program as it exists is under 200 lines long

and is a compromise regarding meshing. Aside from the dimensions, there is no easy way to address most mesh problems .

As the code exists now, I'll leave a copy on the Sherline site, since its short, uncustomized, and could be used for planetary gears or pinions, your choice.