Contents

10/28/2014 Comparison of various types of linear motion design 2

10/29/2014 Learn AutoCAD 2015 3

11/05/2014 Install and learning SolidWorks for 3D modeling 3

11/9/2014 redesign of large cnc 4

12/23/2014 flash HEX file onto arduino for shapeoko 2 grbl software 4

01/31/2015 DC spindle 5

04/11/2015 using gshield on other microcontrollers 6

04/12/2015 Altera FIFOed UART controller 6

04/15/2015 wiring DE0-NANO to gshield 7

04/17/2015 routing algorithm design and arcs in gcode 8

04/18/2015 Adding Z axis to FPGA 10

04/19/2015 updating CNC GUI 10

04/25/2015 Version control Git Eclipse plugin 11

04/29/2015 Generate SSH key and adding to github 14

05/02/2015 Install matplotlib for mac 14

05/09/2015 pulse test, first routing job, and next tasks 14

05/16/2015 updating vhdl pulse generator with acceleration profile 17

05/18/2015 clone remote git repository using egit, and adding to eclipse project 23

09/16/2015 renaming a branch 24

09/16/2015 merging from one branch into another 25

09/17/2015 Identifying checked in files as non source 29

05/30/2015 Testing acceleration generation on stepper motors 30

06/14/2015 resolving pulse generator error VHDL implementation and design 31

06/26/2015 generating accurate gcode coordinates from drawing with specific dimensions 32

07/01/2015 making more accurate drawing to gcode interpretations using autocad 34

07/29/2015 adding another task and a Mutex for shared data between tasks 35

07/30/2015 adding altera FIFOed UART 36

08/14/2015 implementing multithreading and running multiprocessing in python 36

08/30/2015 test cut on foam 44

08/31/2015 optimal settings for cutting acrylic 48

09/07/2015 cnc routing leaf design 49

Large scale RAPTOR

**R**apid

**A**utomated

**P**rototyping

**T**echnology

**O**n

**Rtos**

## 10/28/2014 Comparison of various types of linear motion design

ball screw has low friction, this means it slides easier. When cutting, the table is likely to move unexpectedly.

Acme screw has a jacking thread effect. An acme thread will hold a load in place by design, while ball-screw just rotates away from loads. acme threads can have small leads, ball screws cant because they have to allow room for the balls. Acme screws have the disadvantage of backlash; although correctable with special devices or controllable with repeated adjustment as the screws and threads wear. CNC manufacturers typically use ball screw with oversized diameters to mitigate back driving loads.

The big advantage of acme screw is low cost. Backdriving example: on a vertical axis, when power is off the attached router will tend to fall down, this can be solved by using a finer lead screw.

A rack and pinion system has the advantage of being expandable to larger machines easier. Some of the most advanced CNC machines on the market uses rack and pinion drive system. Although it can also become quite expensive and complex requires gear reduction and other hardware to eliminate radial load on the motor shaft.

Based on simplicity , cost and availability I’ve decided to build a rack and pinion design. This configuration is not common among hobbyists (hobbyists typically use a belt and driven system).

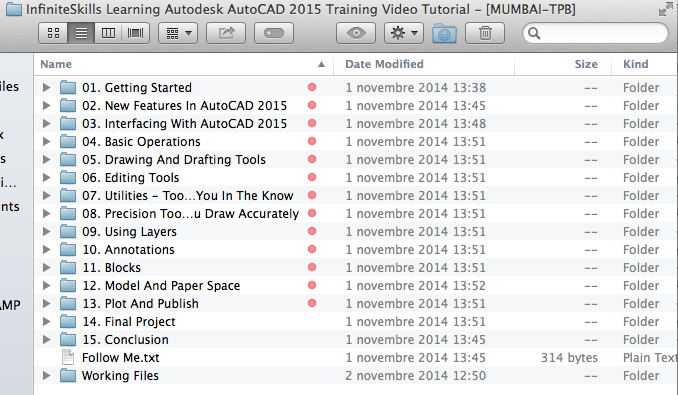
The goal of this design and to create something that can be scaled or upgraded later. Rack and pinion will allow easier scaling in the future and if a more advanced gearing setup is needed later, it can be done later. The point is to get something going quickly to test and learn.

Since the materials for this project will be costly, 3D modeling will provide a good understanding of the end result before production.

## 10/29/2014 Learn AutoCAD 2015

I have had previous experience with AutoCAD about 10 years ago for 2D drawing and 3D modeling.

Having completed the AutoCAD training below:



I remember that this is a great tool for 2D drawing and modeling with limited 3D functionality. It consists mainly of trim, chamfer, fillet and other commands.

Based on forum feedback and research, for more advanced 3D modeling the preferred software is SolidWorks or Inventor. The end result of the research is that SolidWorks will be a better software for modeling and serve as a more useful tool in the future.

## 11/05/2014 Install and learning SolidWorks for 3D modeling

This section will describe my notes on SolidWorks (a software that I have no experience with until today).

## 11/9/2014 redesign of large cnc

research on the forum shows the McMaster rack and pinion is not precise enough for building an accurate cnc machine. Overall, rack and pinion is just going to be very expensive.

<http://www.cnczone.com/forums/diy-cnc-router-table-machines/191798-rack-pinion-vs-ballscrew.html>

ive decided to reduce the cost of the CNC machine by using belts drive instead of rack and pinion. Following the shapeoko 2 design means better support from the opensource community and will provide a base point of where I can learn more regarding cnc machines.

Rather than finding all the components buying the mechanical kit is better price wise.

The shapeoko has limited z axis range, this is something that would require major design modification. Even with a larger maker slide on the z axis the machine would be limited; when the spindle moves up higher than the z axis slide the slide would still be present. This will have to be a later project.

## 12/23/2014 flash HEX file onto arduino for shapeoko 2 grbl software

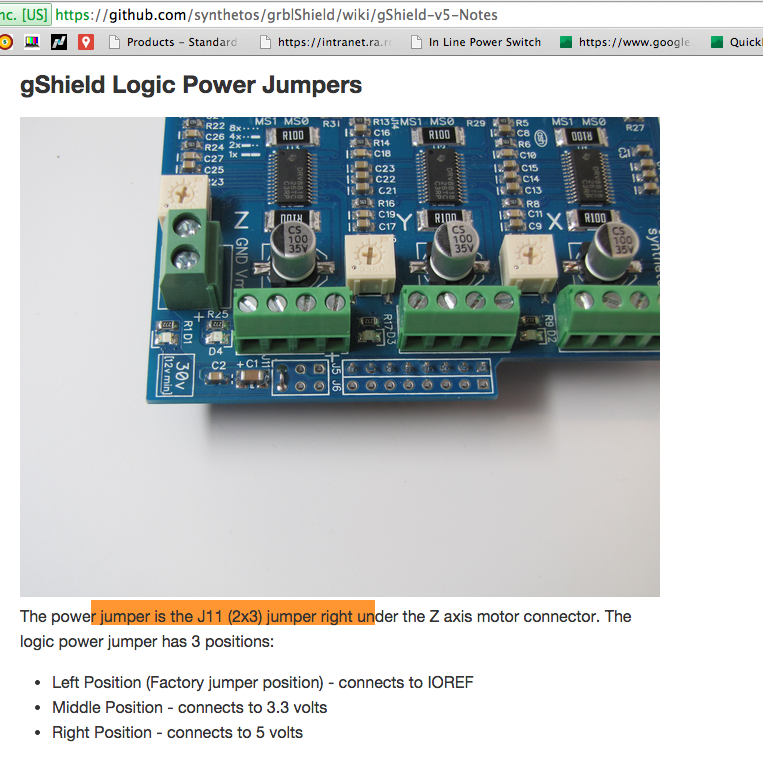
instructions for using GUI interface to flash hex file

<https://github.com/paulkaplan/HexUploader/wiki/Using-HexUploader>

download grbl hex file

<https://github.com/grbl/grbl/downloads>

note: I had to solder a jumper to tell gShield that I want to use 5V signals. See : <https://github.com/synthetos/grblShield/wiki/gShield-v5-Notes>



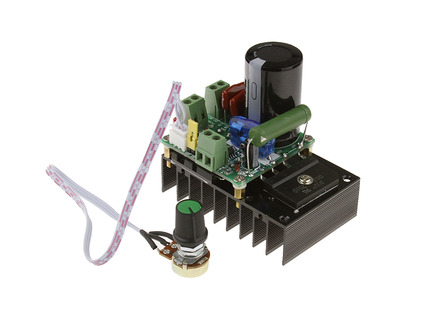
## 01/31/2015 DC spindle

This speed controller is specifically designed for use with our [Quiet Cut, DC spindles](https://www.inventables.com/technologies/300w-dc-spindle). It will not work with the AC powered routers or rotary tools. The speed can be controlled with a potentiometer or an external PWM source such as Mach3. We recommend 48VDC input for best performance with our Quiet Cut DC spindles.

These come with potentiometer on an approximately 1ft (300mm) cable with a knob.

See this [blog post](http://blog.inventables.com/2014/06/shapeoko-upgrade-quite-cut-spindle-with.html) explaining how to use it with a gShield and grbl.

**Note:** This speed controller will only work with our 300W spindles. We are testing controllers for the higher power spindles, but don’t have a firm date yet for when they will be available.



to use PWM to drive spindle speed switch jumper to J21 and provide a 1 to 5KHz PWM signal.

## 04/11/2015 using gshield on other microcontrollers

gshield will work on 3.3v

on the gshield v5b the left position power jumper is actually a trace on the board.

<https://github.com/synthetos/grblShield/wiki/gShield-v5-Notes>

when the left position jumper is left as it is possible to provide 3.3v or 5v to the gshield as Vcc

the enable signal on pin 1 of J7 is normally low to enable the stepper drivers.

Step signal with 30uS low has been tested working with gshield. It may be possible to use a lower pulse width.

## 04/12/2015 Altera FIFOed UART controller

the standard UART module used in qsys does not have a FIFO to buffer the communication.

The FIFOed uart can be found at

<http://www.alterawiki.com/wiki/FIFOed_Avalon_Uart>

as of today v13.1 is the latest available to download but 9.3.1 is used instead because v13 will not work in Quartus 11.

Edit: fifoed uart does not seem to work,

Virtual JTAG uart may be an alternative

<http://idle-logic.com/2012/04/15/talking-to-the-de0-nano-using-the-virtual-jtag-interface/>

## 04/15/2015 wiring DE0-NANO to gshield

wires to connect:

3 direction signals J9

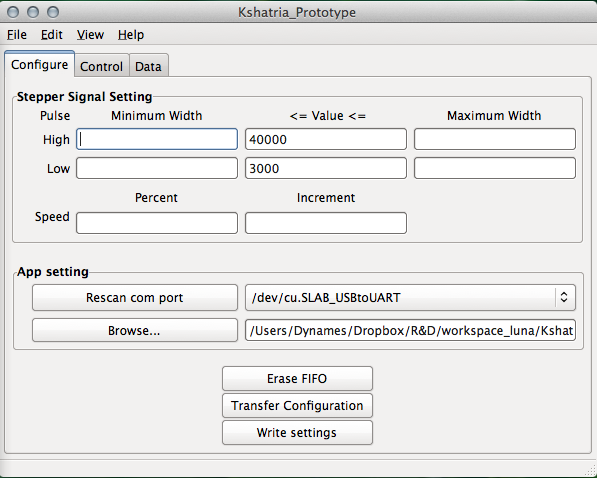
3 step signals J9

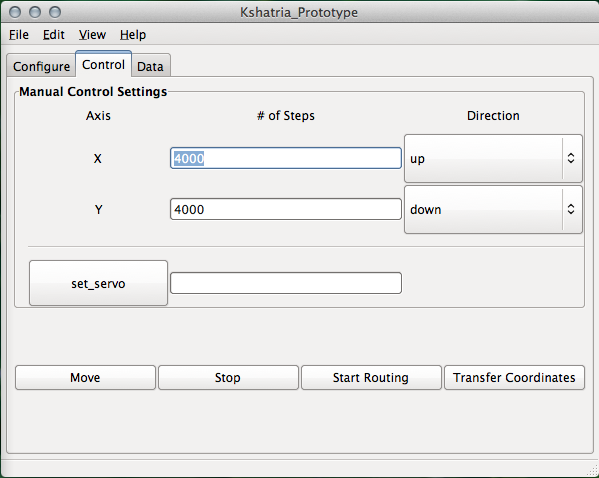
GND J8

Enable J7

Vcc J6 pin 7 to J11 pin 2, 4, or 6

With the Nano wired to the gshield, the following settings were used





DE0-nano runs on 50MHz or 20ns periods

Each increment of pulse width value is one period

20ns\*40000=800micro seconds

20ns\*3000 = 3us

settings:

3us Low pulse width

800us for high pulse width

sending 4000 pulses to x and z axis

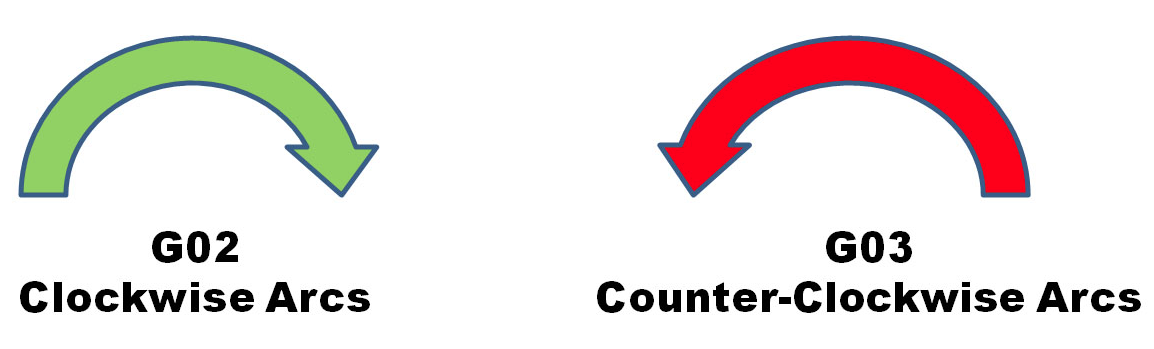
at these settings speed of the stepper motors seemed to be moving at a good pace.

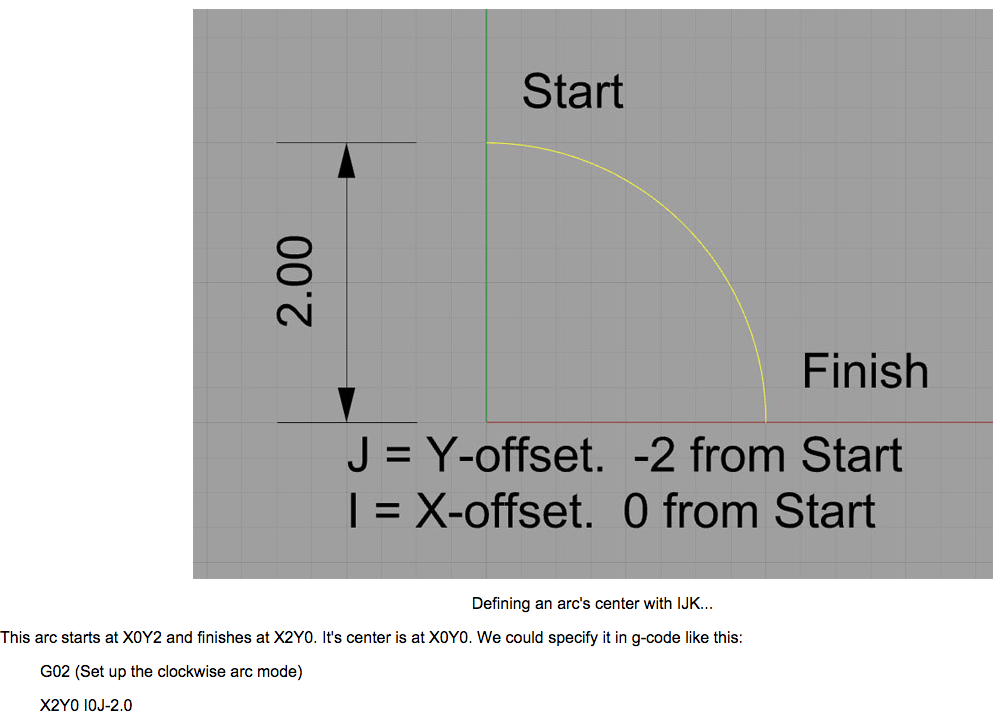
This speed would be a good initial point to start testing speed limits.

## 04/17/2015 routing algorithm design and arcs in gcode

coordinate input to the software is in Gcode format. The Kshatriya prototype contains only basic routing instructions and supported only straight lines.

Future software updates will contain support of commands with I and J variables which store arc information.





detailed information on ARC routing can be found here:

<http://www.cnccookbook.com/CCCNCGCodeArcsG02G03.htm>

From reading information regarding federate, it seems that there is an added benefit to increase the RPM of the spindle as the bit plunges deeper into some materials.

Design for routing arc’s and circle’s:

Bresenham’s algorithm can be implemented via FPGA to efficiently and accurately plot a circle. From gcode G02 ,G03, I, and J the mid point and end point of the arc is known. FPGA can start at a starting point and constantly calculate its position with respect to the mid point while sending a pulse through x and/or y axis; this algorithm would be basically walking the curve. In this way, it is possible to achieve a max resolution that is limited only by distance traveled for a single step.

Upgrading previous RAPTOR to support arc’s, circle’s, curves:

The previous implementation of RAPTOR will only support straight line coordinates. As a quick and easy way to start routing circles, a separate program may be created that parse a GCode file with arc/circle coordinate support and translate into linear coordinates.This is done by interpolating the ARC. This program will of course generate a much larger gcode file than the original. The size of this translated gcode file will of course be exponentially proportional to the resolution of the curve/arc/circle. This method will be less efficient than the FPGA method described above but will be quicker to implement because its not as complex.

## 04/18/2015 Adding Z axis to FPGA

previous implementation only had X and Y axis stepper motors. Today I added functionality to support Z axis control too.

The control register bits , previously:

Workhorse:Users:Dynames:Desktop:Screen Shot 2015-04-18 at 03.55.48.png

Control register bits, now with Z axis control bits:

Workhorse:Users:Dynames:Desktop:Screen Shot 2015-04-19 at 00.14.42.png

the firmware is now updated to receive MoveZ, StepNumZ, and DirZ commands.

## 04/19/2015 updating CNC GUI

Kshatriya GUI sends data over serial and each time I click send the whole GUI would freeze until data sending is complete. Today, I updated Kshatriya to send serial data using a parallel running process (using multiprocess), now I can add messages to a queue and the parallel process will take care of transmitting without halting everything in the main GUI process.

Next Tasks

since the high pulse width max/min and low pulse width max/min is not used I will get rid of them

add support to change xyz high pulse width value individually

instead of clicking a button to transfer configuration, transfer the configuration for an axis only when the move button for that axis is clicked

add buttons to move x, y, and z axis individually

rename move button to move x and y

note: see the post on 08/14/2015 for fixed solution regarding multithreading for the GUI

## 04/25/2015 Version control Git Eclipse plugin

up until now all code has been backed up on DropBox

today’s focus is getting that code versioned in Git

on windows the Nios IDE does not support the latest egit eclipse plugin. Use an older version instead

Eclipse > Help > Install New Software…

Add update site http://download.eclipse.org/egit/updates-2.1

to add existing local files to a repository

<https://help.github.com/articles/adding-an-existing-project-to-github-using-the-command-line/>

on mac (works for eclipse luna)

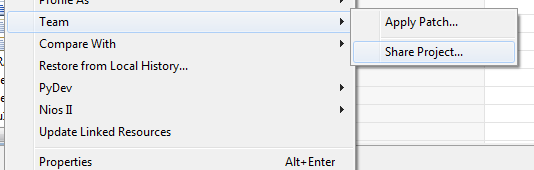
Eclipse > Help > Install New Software…

add update site http://download.eclipse.org/egit/updates

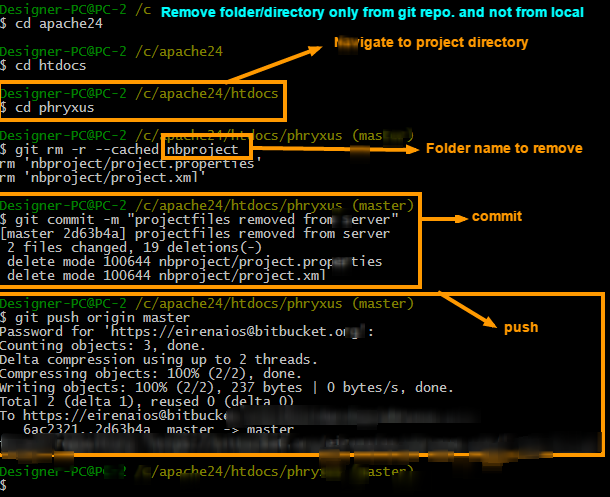
see documentation on egit the Eclipse Git plugin

<http://wiki.eclipse.org/EGit/User_Guide#Overview>

right click on a project then Team > Share Project



to remove a folder from git



to delete a branch from git

<http://matthew-brett.github.io/pydagogue/gh_delete_master.html>

For FPGA files and other non eclipse project based files, creat a blank project in eclipse, then import file system into the blank project. This will make it easier so other files can go through eclipse to push to GitHub instead of using command line (which seems confusing).

To delete a branch, rename or push to a different branch:

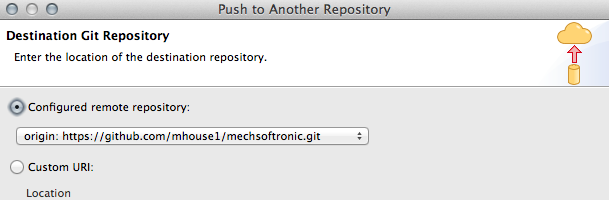
Renaming a branch:

Right Click on project and select Team>Switch To>Other… highlight the local branch you want to rename then click the Rename … button and click Checkout

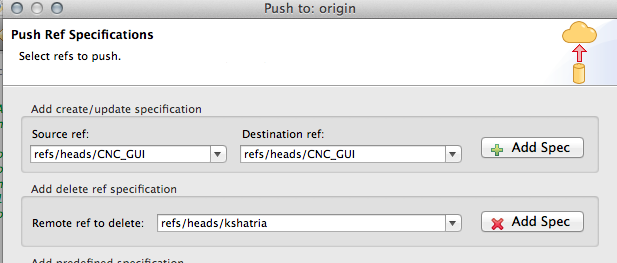
Publishing to new branch and removing old branch:

Right click on project and select Team>Remote>Push…

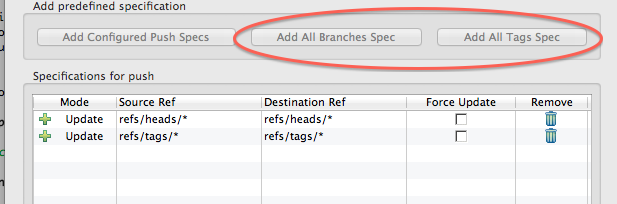
Select the remote repository



next select source and destination to push from and to



click Add spec buttons for create and delete



to push to a folder within a repo branch

Share project > select the git repository and folder within repo

Click on folder to add to index, should see green pluses next to file that is added

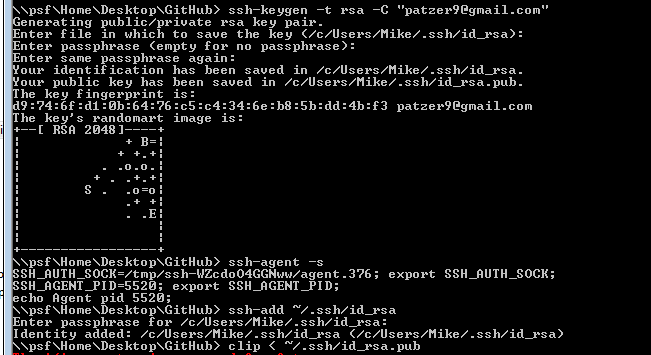
Commit

push

## 04/29/2015 Generate SSH key and adding to github

see here for generating ssh keys

<https://help.github.com/articles/generating-ssh-keys/#platform-windows>



## 05/02/2015 Install matplotlib for mac

install matplotlib-1.3.1-py2.7-python.org-macosx10.6.dmg

<http://sourceforge.net/projects/matplotlib/files/matplotlib/>

will also need numpy

install numpy-1.8.0-py2.7-python.org-macosx10.6.dmg

<http://sourceforge.net/projects/numpy/files/NumPy/>

there are newer versions of the packages above but I have not tested them

## 05/09/2015 pulse test, first routing job, and next tasks

connected the XY axis belts to the stepper motors today. Testing done to measure how many steps per mm.

Testing done on the axis:

Sent 1000 steps, this equals 22mm of travel.

1000/22 = 45.45 pulses per mm

More accurate measurement will be done later but for now the 44.45 pulses per mm will be the starting point.

The table below shows the range of speeds tested

|  |  |  |
| --- | --- | --- |
| Pulse configuration | Fast movement | Slow movement |
| High pulse width count | 15000 | 400000 |
| Low pulse width count | 3000 | 3000 |

Routing:

This is the first routing job done on using the firmware developed so far

The GCode sent is the same GCode used on the prototype laser cutter

GCode:

G90

G21

G0 X0. Y0.0

M05

G1 X2.71428571429 Y8.14285714286

M03

G1 X13.5714285714 Y19.0

G1 X8.14285714286 Y19.0

G1 X2.71428571429 Y24.4285714286

G1 X8.14285714286 Y29.8571428571

G1 X29.8571428571 Y29.8571428571

G1 X35.2857142857 Y24.4285714286

G1 X29.8571428571 Y19.0

G1 X24.4285714286 Y19.0

G1 X35.2857142857 Y8.14285714286

G1 X32.5714285714 Y8.14285714286

G1 X21.7142857143 Y19.0

G1 X21.7142857143 Y8.14285714286

G1 X19.0 Y8.14285714286

G1 X19.0 Y29.8571428571

G1 X19.0 Y8.14285714286

G1 X16.2857142857 Y8.14285714286

G1 X16.2857142857 Y19.0

G1 X5.42857142857 Y8.14285714286

G1 X2.71428571429 Y8.14285714286

M05

M05

G1 X0.0 Y0.0

G1 X16.2857142857 Y21.7142857143

M03

G1 X10.8571428571 Y21.7142857143

G1 X8.14285714286 Y24.4285714286

G1 X10.8571428571 Y27.1428571429

G1 X16.2857142857 Y27.1428571429

G1 X16.2857142857 Y21.7142857143

M05

G1 X21.7142857143 Y21.7142857143

M03

G1 X27.1428571429 Y21.7142857143

G1 X29.8571428571 Y24.4285714286

G1 X27.1428571429 Y27.1428571429

G1 X21.7142857143 Y27.1428571429

G1 X21.7142857143 Y21.7142857143

M05

M05

G1 X0.0 Y0.0

M03

G1 X0.0 Y38.0

G1 X38.0 Y38.0

G1 X38.0 Y0.0

G1 X0.0 Y0.0

M05

G0 X0.000 Y0.000

M05

M02

Note: only the GUI GCode parser currently reads just the XY values into a list and sends over to firmware. Firmware moves from one point to the next linearly. The idea is to draw the shape, drawing to scale would require more configuration later. In this job, the GCode parser reads all xy values and scaled it up by multiplying all values by 5000.

First routing notes:

It seems that when moving along the y axis the whole machine shakes a lot more, I suspect that this is because the belt is unable to grip the stepper motor gears well enough. Also the pulses sent to the stepper motors are a fixed pulse width so this means no acceleration or deceleration for movements, adding acceleration would most likely smooth things out better.

Solutions to grip the gear better:

Replace the y axis gear with a larger diameter

Replace the two y axis mounting plate with a longer plate, this will allow room for the motors to shift upwards and slightly grip the gears better.

Or modify the y axis mounting plates so the motors can slide downwards, then the belts would be turned so that the teeth is facing up. Instead of the belt gears resting on the top of the gear it would be gripping it from the bottom

Next tasks:

Add acceleration profile to pulse generator

Add parameter for number of layers to cut

Add parameter for thickness of layers

~~Add parameter for setting base pulse configuration for routing~~

~~Calibrate routing distance~~

Parse gcode to determine when path is cutting vs not cutting

Configure firmware to read routing status (router on/off)

## 05/16/2015 updating vhdl pulse generator with acceleration profile

Parameters needed for pulse generator with acceleration

Starting\_speed : the speed to start accelerating from

Speed\_change : the speed change rate

Algorithm:

Start at starting\_speed as actual speed, for each pulse decrease from actual speed by variable speed\_change. A deceleration section should be calculated so if there is not enough pulses to be sent to reach target speed then deceleration should start when the number of steps allowed to accelerate equals number of steps remaining. If target speed is reach then hold that speed until deceleration begins.

The following series of sequences has been tested working in the pulse generator module updated to include acceleration feature:

--test even number of steps target speed not reached

wait for clk\_50mhz\_period\*5;

pulse\_width\_high <= x"00000005";

pulse\_width\_low <= x"00000002";

number\_of\_steps <= x"00000008";

speed\_start <= x"0000000F";

speed\_change <= x"00000002";

wait for clk\_50mhz\_period;

start\_generation <= '1';

wait for clk\_50mhz\_period;

start\_generation <= '0';

wait for 2500 ns;

--test even number of steps target speed reached

wait for clk\_50mhz\_period\*5;

pulse\_width\_high <= x"00000005";

pulse\_width\_low <= x"00000002";

number\_of\_steps <= x"0000000F";

speed\_start <= x"0000000F";

speed\_change <= x"00000002";

wait for clk\_50mhz\_period;

start\_generation <= '1';

wait for clk\_50mhz\_period;

start\_generation <= '0';

wait for 4000 ns;

--test odd number of steps target speed reached

wait for clk\_50mhz\_period\*5;

pulse\_width\_high <= x"00000005";

pulse\_width\_low <= x"00000002";

number\_of\_steps <= x"0000000E";

speed\_start <= x"0000000F";

speed\_change <= x"00000002";

wait for clk\_50mhz\_period;

start\_generation <= '1';

wait for clk\_50mhz\_period;

start\_generation <= '0';

wait for 4000 ns;

--test force target speed if next acceleration increment would

--make actual\_speed faster than target speed. Test passes if

--target speed reached for acceleration and successful deceleration from target speed

wait for clk\_50mhz\_period\*5;

pulse\_width\_high <= x"00000005";

pulse\_width\_low <= x"00000002";

number\_of\_steps <= x"0000000F";

speed\_start <= x"0000000F";

speed\_change <= x"00000003";

wait for clk\_50mhz\_period;

start\_generation <= '1';

wait for clk\_50mhz\_period;

start\_generation <= '0';

wait for 4000 ns;

--test initial speed faster than target speed

--if initial speed is faster than target speed then hold target speed

wait for clk\_50mhz\_period\*5;

pulse\_width\_high <= x"0000000F";

pulse\_width\_low <= x"00000002";

number\_of\_steps <= x"0000000F";

speed\_start <= x"00000005";

speed\_change <= x"00000003";

wait for clk\_50mhz\_period;

start\_generation <= '1';

wait for clk\_50mhz\_period;

start\_generation <= '0';

wait for 6000 ns;

--test initial speed faster than target speed and even number of steps

--if initial speed is faster than target speed then hold target speed

wait for clk\_50mhz\_period\*5;

pulse\_width\_high <= x"0000000F";

pulse\_width\_low <= x"00000002";

number\_of\_steps <= x"00000010";

speed\_start <= x"00000005";

speed\_change <= x"00000003";

wait for clk\_50mhz\_period;

start\_generation <= '1';

wait for clk\_50mhz\_period;

start\_generation <= '0';

wait for 6000 ns;

--test initial speed and speed change are zero

wait for clk\_50mhz\_period\*5;

pulse\_width\_high <= x"00000005";

pulse\_width\_low <= x"00000002";

number\_of\_steps <= x"00000020";

speed\_start <= x"00000000";

speed\_change <= x"00000000";

wait for clk\_50mhz\_period;

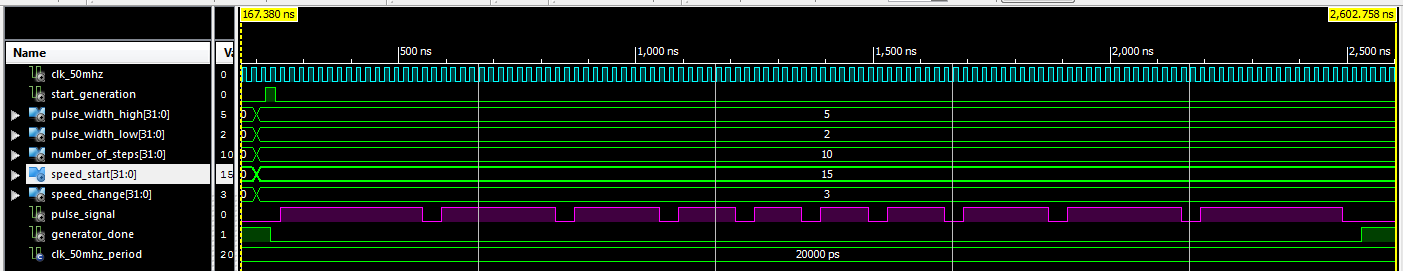
start\_generation <= '1';

wait for clk\_50mhz\_period;

start\_generation <= '0';

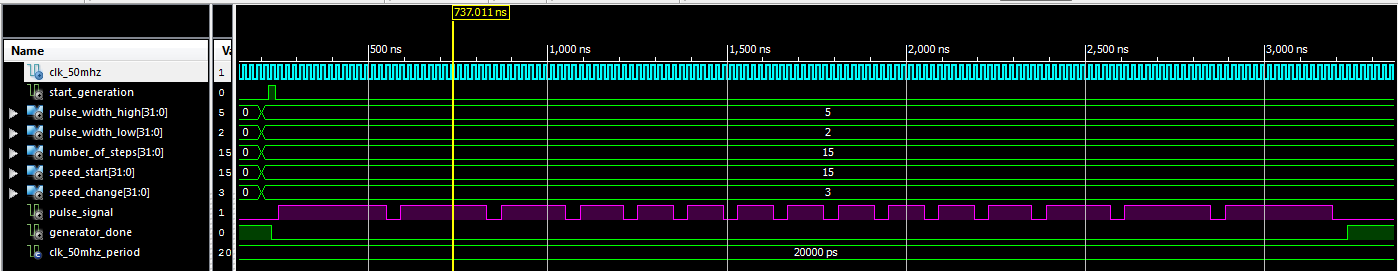
wait for 5000 ns;

Test bench done in Xilinx ISE ISim



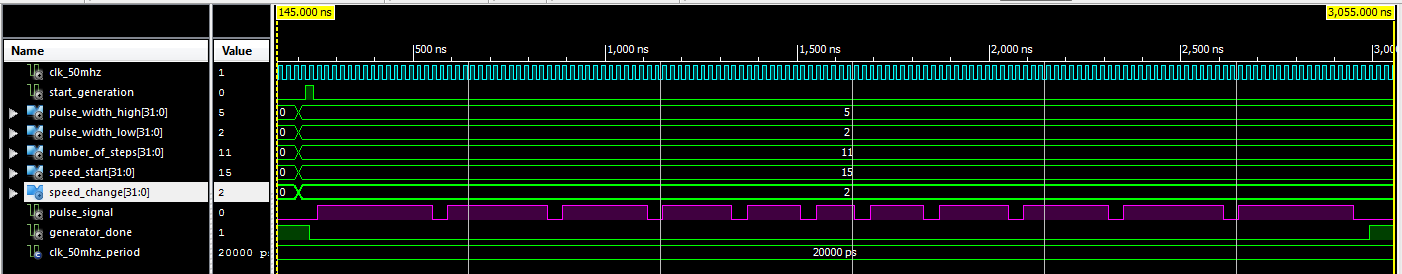
Simulation: when pulses never reach target speed

As we can see, acceleration ramp never reach target speed deceleration begins starting at pulse #6



Simulation: when pulses reach target speed

as we can see, when target speed is reached speed is held constant



simulation: when odd number of steps sent

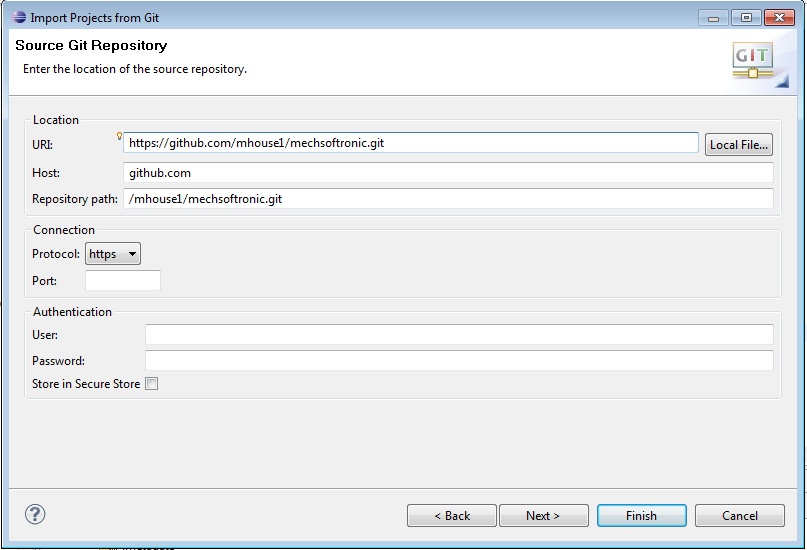
as we can see the 6th pulse is the smallest pulse, deceleration starts after that.

## 05/18/2015 clone remote git repository using egit, and adding to eclipse project

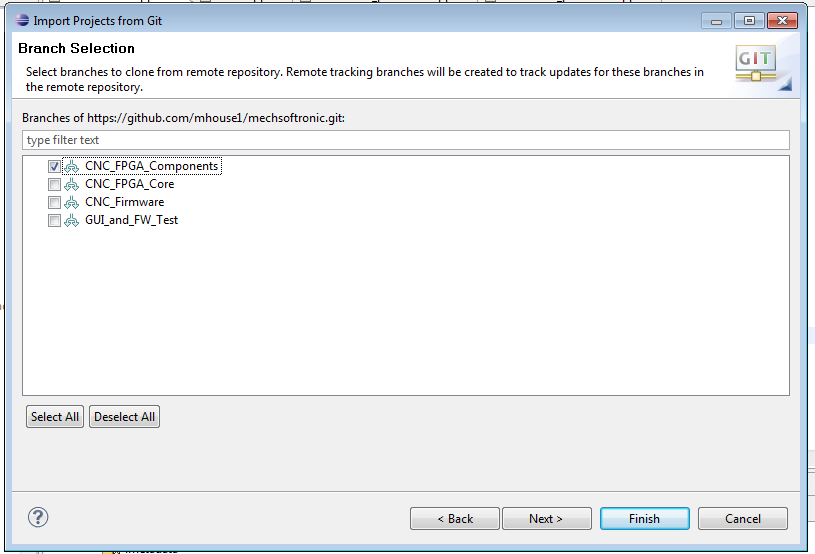
in eclipse…

File > Import > Git > Project from Git > URI > Next >

Then enter the following information for URI, host and repository path then click next



After clicking next, select the branch to clone



once the branch is brought into the project workspace, right click on project and select “share project …”, and now you can make changes and submit to that branch

### 09/16/2015 renaming a branch

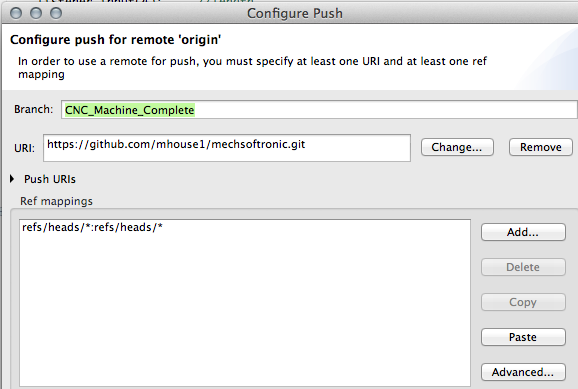
right click on the project > Team > Advanced >Rename branch …

renaming is actually creating a new branch with the same content.

Now to commit it as a new branch

Team > Push to Upstream

Will need to click configure to set the push as the name of new branch.

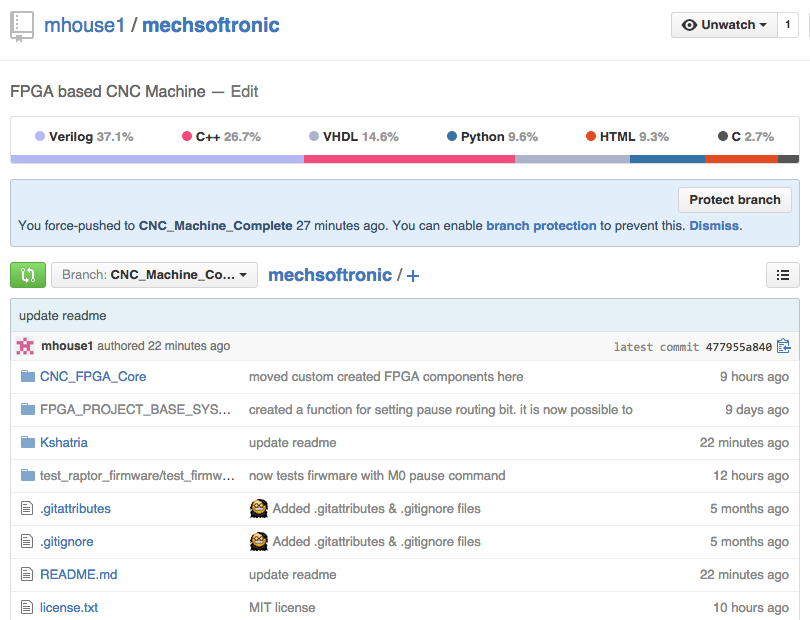


### 09/16/2015 merging from one branch into another

|  |  |
| --- | --- |
| Currently the project has several branches with its own commit histories. Each branch has its own type of source. I want to merge the other branches into one. | Workhorse:Users:Dynames:Desktop:Screen Shot 2015-09-16 at 18.59.41.png |
| Fetch from upstream to get the latest of other branches that you want to merge | Workhorse:Users:Dynames:Desktop:Screen Shot 2015-09-16 at 19.00.43.png |
| Select merge | Workhorse:Users:Dynames:Desktop:Screen Shot 2015-09-16 at 19.01.30.png |
| Select the branch to merge into current branch and click ‘Merge’ | Workhorse:Users:Dynames:Desktop:Screen Shot 2015-09-16 at 19.02.49.png |
| Click Team > Push to Upstream to push to GitHub | Workhorse:Users:Dynames:Desktop:Screen Shot 2015-09-16 at 19.06.28.png |
|  |  |
|  |  |
|  |  |

### 09/17/2015 Identifying checked in files as non source

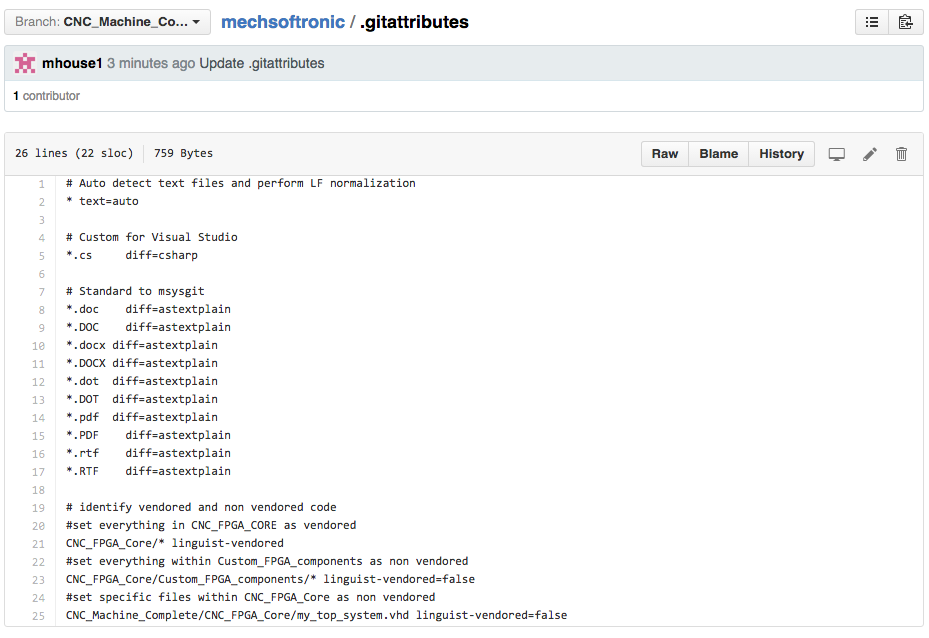
The GitHub statistics bar show the percentage of a languages that make up the project.



sometimes this data is wrong because it determines generated files as source. In the figure above HTML is generated source. To identify HTML as generated source we will modify the .gitattributes file.

In the figure a modified .gitattributes file specifies which files are vendored and non vendored.

For more details on linguist see: <https://github.com/github/linguist#overrides>



## 05/30/2015 Testing acceleration generation on stepper motors

Will use the following base settings to start testing at:

Acceleration = 400000

Speed change = 1000

Pulse width high = 70000

Pulse width low = 10000

Number of steps = 2000

Oscilloscope trigger

Trigger > Menu > Mode = Edge , Slope = rising, Sweep = normal

## 06/14/2015 resolving pulse generator error VHDL implementation and design

Previously the pulse generator has been tested working with acceleration working for x and y axis. For some reason the same design does not work for z axis, it just produces a pulse with constant pulse width. I suspect that I don’t fully understand some aspects of VHDL. Maybe it’s the timing things ive herd about.

The problem: I’ve created 3 instances of pulse generator module and have connected them exactly the same. With my top module created it seems to work properly for x and y axis. When I make updates to the top module such as add another instance or add anything more to the top module the 3 pulse generator instances behaves differently.

From research : <http://www.edaboard.com/thread314515.html>

You've got both std\_logic\_unsigned and numeric\_std. Use numeric\_std only then change all the std\_logic\_vector signals that you do "+ 1" to unsigned.

realistically VHDL is not that hard to learn (I just read the Perry book cover to cover), but you have to start by reading a book and not just trying to pick up what you need (like you can with some software languages and most scripting languages).

*I assumed that VHDL was easy enough to just learn by reading a little bit online combined with trial and error.*

Update 08/14/2015 it turns out problem was caused because on restart the FPGA registers that is not reset is in an unknown state, for example since I did not specify a default value for a signal or register it may be default to either high or low.

To solve this I added a reset input signal to the pulse generator module that triggers default values to be set to bring the pulse generator to a known state.



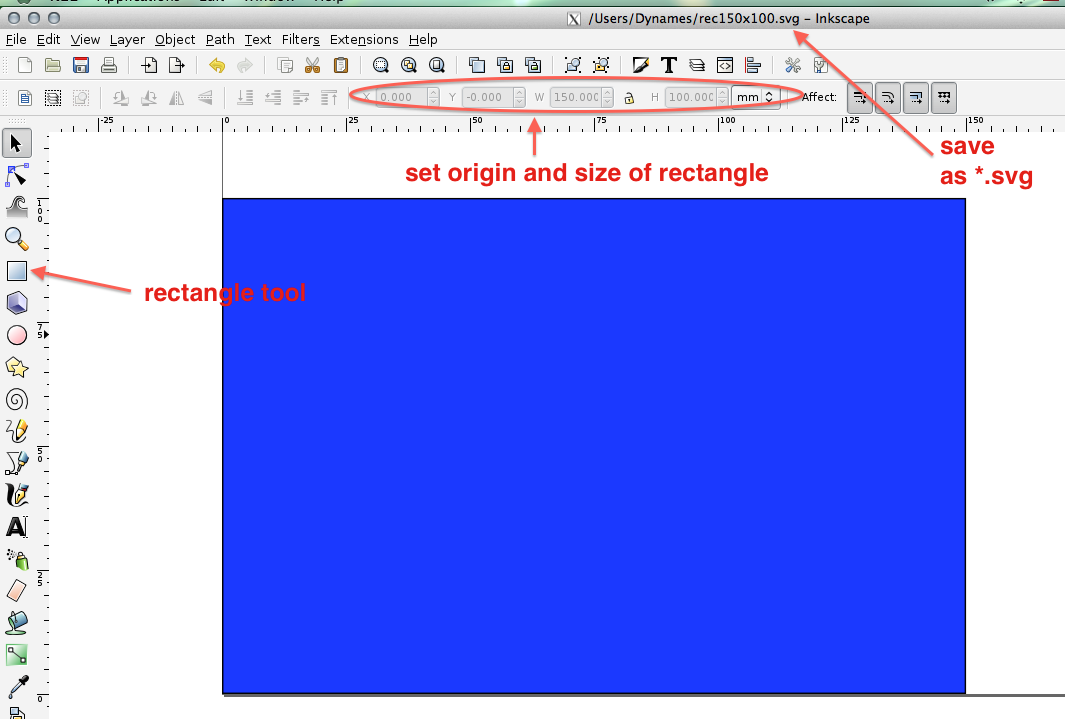
08/21/2015 the above solution works for single axis. This solution would not work for multiple axis when for example the xy axis is drawing a diagonal that is not 45 degrees. When one axis moves slower than the other this would introduce a skew because one axis is accelerating faster than the other.

A work around to this would be to bring out the acceleration state signal and have an external module create varying clock speeds based on acceleration state. The varying clock speed would clock both xy axis so they are synchronized.

## 06/26/2015 generating accurate gcode coordinates from drawing with specific dimensions

Example: drawing an 150mm x 100mm rectangle using inkscape

File > Document Properties > Page > Units > mm



in makercam, set units to mm, and set px/inch to 90 then open the svg.

Makercam > edit > preferences > px/in = 90 for inkscape

Note: since we specified origin of 0,0 in inkscape, so now the drawing is set to origin

Note: the svg does not use the exact coordinates of the drawing it converts the drawing to pixles and approximate the location so the gcode gets close to the original size but not the actual. In the future I’ll have to create an application that can draw and generate gcode to produce better results.

Sample Generated GCode:

(Generated by PartKam Version 0.05)

G21 G90 G40

(follow path 1)

G0 Z15

T0 M6

G17

M3

G0 X0.1411 Y100.1411

G1 Z-1.5 F800

G1 X150.141 Y100.14099999999999 F1500

G1 X150.141 Y0.141

G1 X0.141 Y0.141

G1 X0.141 Y100.14099999999999

G0 Z15

## 07/01/2015 making more accurate drawing to gcode interpretations using autocad

I am not happy with the approximate drawing gcode described in the previous post. I have the option of either creating my own software the draws and generate gcode or look for other options. For now I will opt for option two, as option 1 would be more time consuming.

autoCAD is the preferred method of drafting designs, if I can utilize autocad and its output files this would be a powerful combination.

From searching I’ve stumbled upon this post:

Source: <http://forums.ni.com/t5/LabVIEW/AutoCad-to-G-Code-conversion-vi/td-p/70817>

AutoCAD did sell a software library called ObjectDBX giving programmers access to the DWG format. If you have more money to spend than time to program, you may want to look at that.  
If you're programming from scratch, DXF may be the easier way to go. It's in ASCII so you can read the file in any editor or word processor. An easy way to get started is to use AutoCAD to create a very simple drawing, e.g. a square with corners at known locations, generate a DXF file, read that DXF file and look for the coordinates of the squares corners. Each corner is a VERTEX in DXF. Than add a circle to your drawing and look for that. The drawings for the pieces you want to CNC will be much more complicated, but it's best to start with simple examples.

Parsing the DXF seems to be my next step.

Opensource applications are available to do this; dxf2gcode, which is programmed using python with pyqt4.

It seems pyQt4 is worthwhile to learn

Source: <http://ubuntuforums.org/archive/index.php/t-1997496.html>

Which one(Pyqt,Pyslide,PyGTK ,wxPython) would be better option for beginners when it needs to work on GUI.  
  
You spelt PySide wrong. PySide is pretty much the same as PyQt4, but it seems that PySide is better implemented for Python 2.x and PyQt4 is better implemented for Python 3.x, although the difference is probably negligible since both use Qt.  
  
When I started, I tried to use Glade with GTK (don't know if it was PyGTK or GTK3 or whatever), but I just found it really frustrating. wxPython was even worse for what I wanted to do. As soon as I found Qt everything went really easy. So at least for me, as a beginner, Qt was the optimal choice.  
  
PyQt4 also comes with its own database classes, networking, etc. And all of that of course connects really easy to widgets. So you can save some serious time by going that way. Also check out the book linked above "Rapid GUI programming with Python and Qt", which is really an excellent book.

Will clone and modify this project to my liking: <http://sourceforge.net/projects/dxf2gcode/>

This project utilize pyqt4 which will require python 3.3 so ill skip this for now

<http://www.pythonschool.net/pyqt/distributing-your-application-on-mac-os-x/>

for now ill be using a prebuilt program <http://sourceforge.net/projects/dxf2gcode/files/?source=navbar>

## 07/29/2015 adding another task and a Mutex for shared data between tasks

Up until now gcode is sent over serial and translated into movement before routing begins, this would work until there is a large gcode file. Today, the plan is to add a third task to start routing while gcode is being sent.

Plan: task 2 will build up data and be the producer, task 3 will be the consumer and do routing operation. A mutex will ensure the resource is not being accessed at the same time by task 2 and 3.

The implementation will be based off of code found on github:

<https://gist.github.com/alexjaw/b24a77de5f615b2ca16f>

for MicroC/OSII reference manual see:

<http://www.win.tue.nl/~mholende/automotive/uCOS-II-RefMan.pdf>

First define the mutex

//define a mutex

OS\_EVENT \*global\_state\_mutex;

note that the MUTEX\_PRIORITY should have a lower number (higher priority) than task priority

**#define** TASK1\_PRIORITY 10

**#define** TASK2\_PRIORITY 9

**#define** TASK3\_PRIORITY 8

**#define** MUTEX\_PRIO 7

/\* main must create the mutex \*/

**int** **main**(**void**)

{

INT8U err;

//create mutex for shared memory

global\_state\_mutex = OSMutexCreate(MUTEX\_PRIO, &err);

// Create tasks

OSTaskCreate(task1, NULL, &task1\_stk[TASK\_STACKSIZE-1], TASK1\_PRIORITY);

OSTaskCreate(task2, NULL, &task2\_stk[TASK\_STACKSIZE-1], TASK2\_PRIORITY);

OSTaskCreate(task3, NULL, &task3\_stk[TASK\_STACKSIZE-1], TASK3\_PRIORITY);

OSStart();

**return** 0;

}

For tasks, to wait for and release mutex:

INT8U error\_code;

//OSMutexPend parameter 1 is the mutex pending,parameter2 = 0 (means don’t timeout) is the timeout, parameter 3 is the error code to return

OSMutexPend(global\_state\_mutex, 0, &error\_code);

//the splice function a.splice(a.end(), b);

//moves items of B to end of A (emptying B at the same time)

//this operation is O(1)

**if** (!error\_code)

{

global\_machine\_route.splice(global\_machine\_route.end(),listener.routes);

}

**else**

{

**printf**("task 2 waited too long\n");

}

//release mutex

OSMutexPost(global\_state\_mutex);

## 07/30/2015 adding altera FIFOed UART

## 08/14/2015 implementing multithreading and running multiprocessing in python

Multiprocessing is the ability to process multiple things at the same time, sometimes utilizing more than 1 CPU cores. This is done in python using (import multiprocessing) module.

The threading module uses threads, the multiprocessing uses processes. The difference is that threads run in the same memory space, while processes have separate memory. This makes it a bit harder to share objects between processes with multiprocessing. Since threads use the same memory, precautions have to be taken or two threads will write to the same memory at the same time. This is what the global interpreter lock is for.

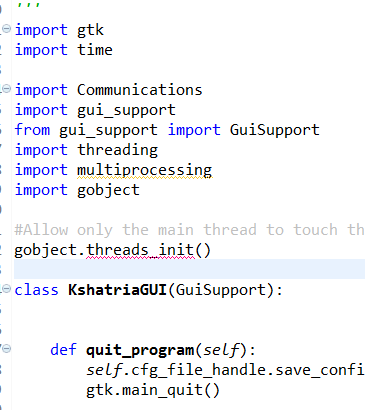
Multithreading allows a process to have multiple threads running in the process at the same time doing their own tasks. While sharing the same memory.

The CNC machine GUI uses threads to push string messages into a queue to be consumed by another thread that transmits the message to the CNC controller. This allows the transmission of message to happen in the back ground so the GUI user does not have to wait for transmission to complete before sending another command.

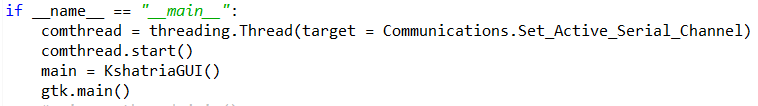
In this case we can also have multiple queues to be consumed by the thread, one for fast and one for slow messages. The user clicks gcode transfer button and it is sent slowly. While the slow messages is transmitting if we wish to interrupt the transmission we can send a fast message.

To implement this in the GUI, remember that the threads must not run within the GUI thread since it is its own thread.

For this to work we must call the gobject.threads\_init()



Notice that the thread we want is seprate from gtk.main() and not creating in a function call inside KshatriaGUI()



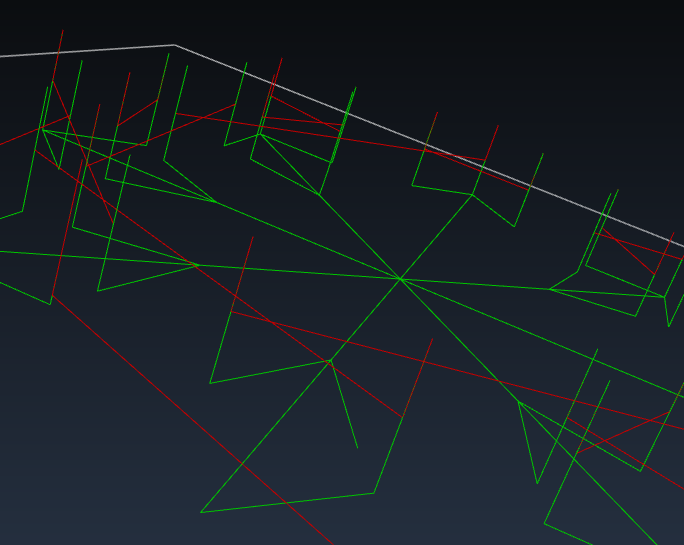
08/24/2015 generating single layer gcode in DXF2GCODE

open dxf file for the snowflake

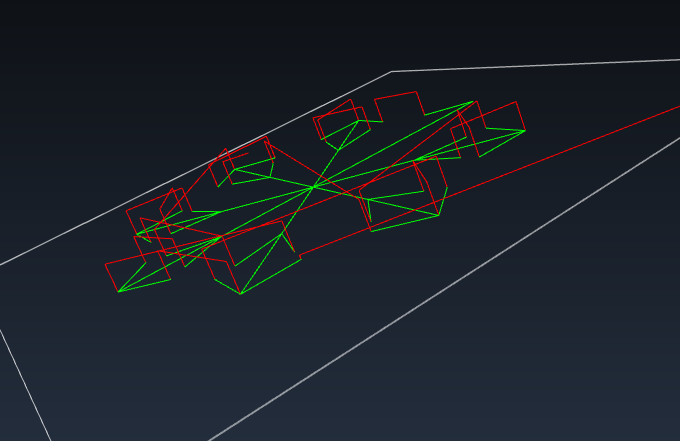
select snowflake layer

safety margin is the amount z to go up before coming down and routing

In the picture below safety margin is 10 and retraction area is 5, as we can see it goes up twice the distance of red lines from point to point

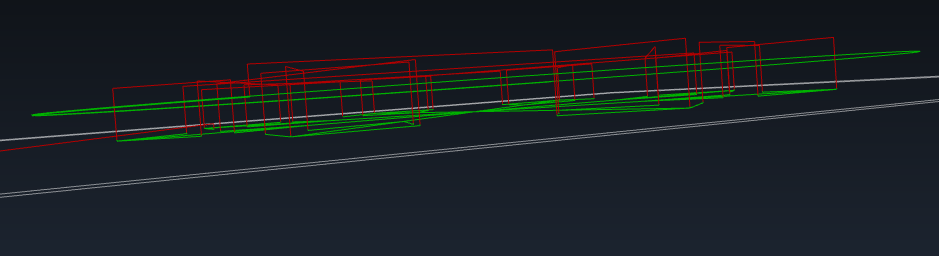


setting safety margin to zero will create equal amount of upward movement between points.

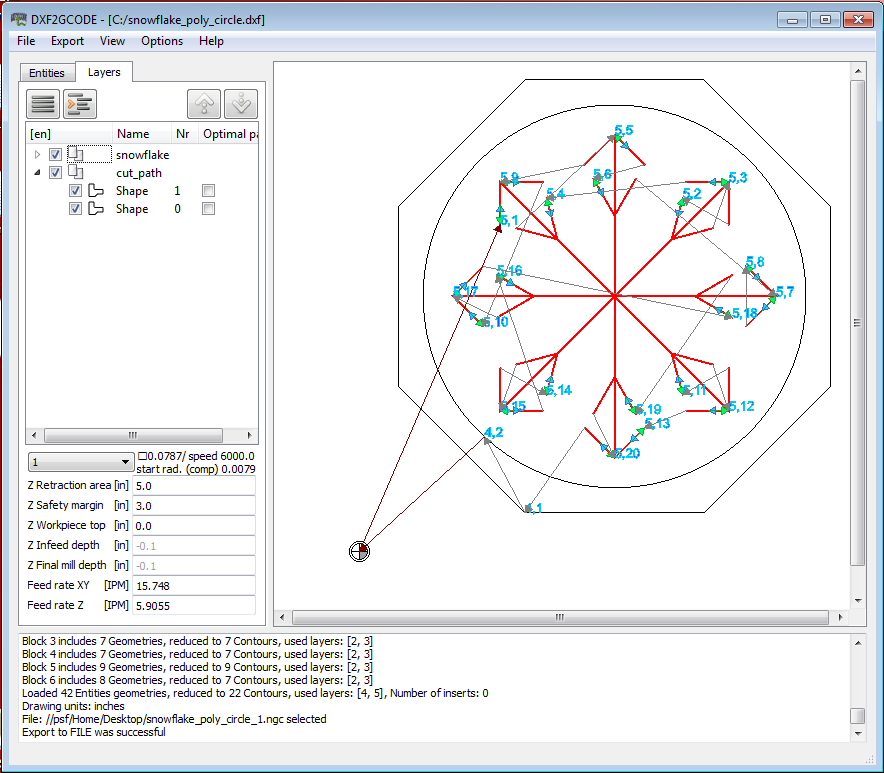


Feedrate xy and federate z is ignored for now (those settings will be over written by firmware).

The workpiece top parameter can be used for example to indicate something should start at a higher height, in this case the circle is set with z workpiece top = 3

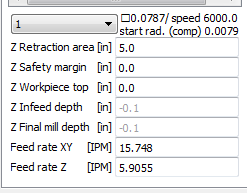


to generate gcode routing for 1 layer make sure in feed depth is the same as final mil depth.

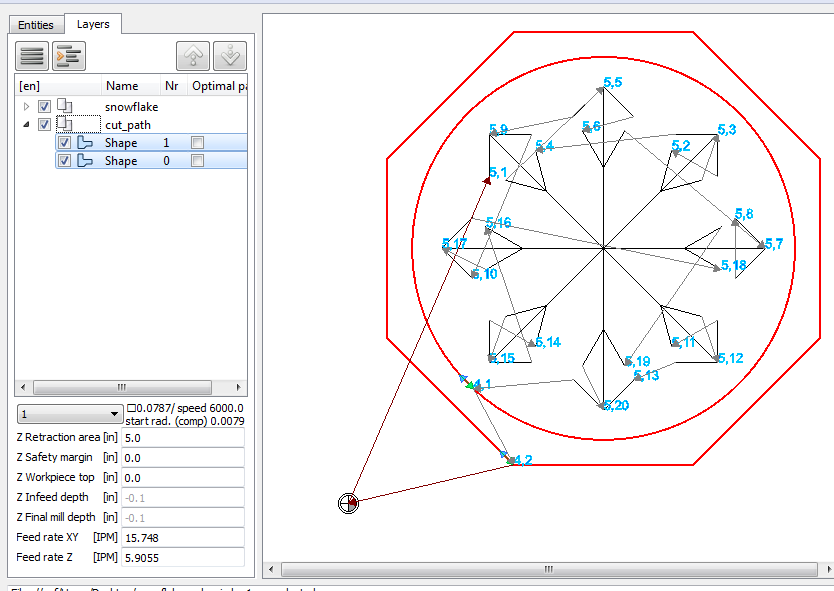


in this case we only want to route a single layer to quickly see the end result

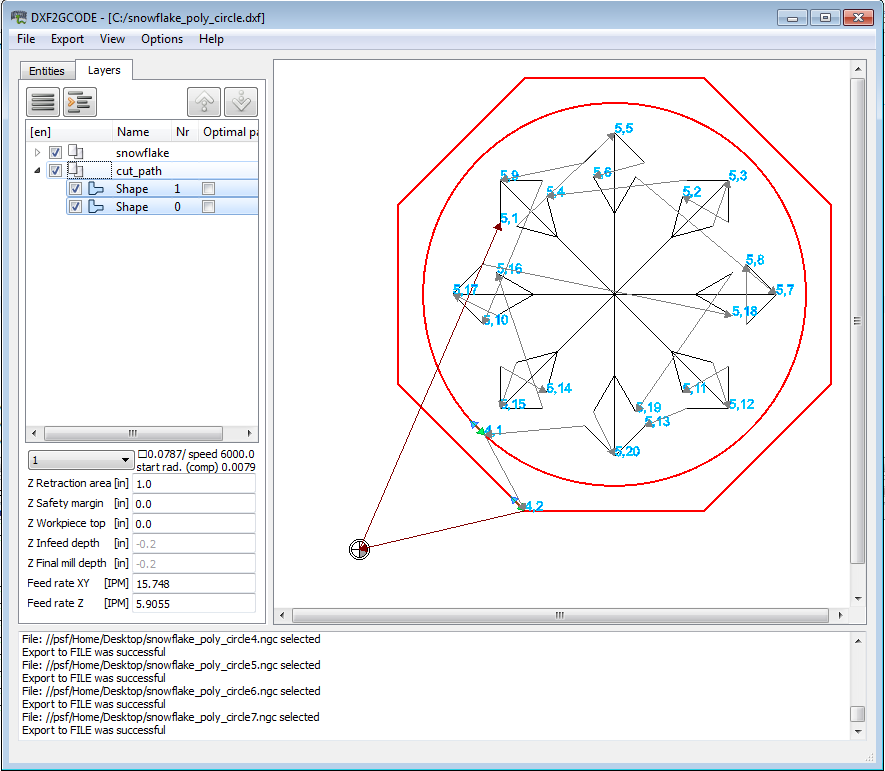
for the snow flake layer the following settings were used

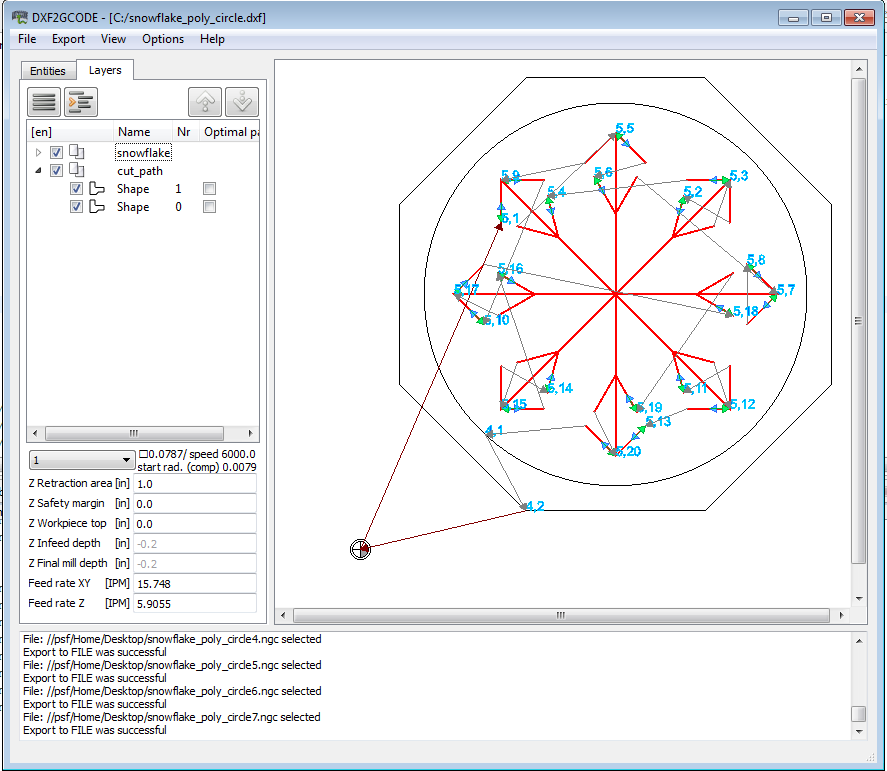


for the circle and polygon layer (highlighted in red) the following settings were used.

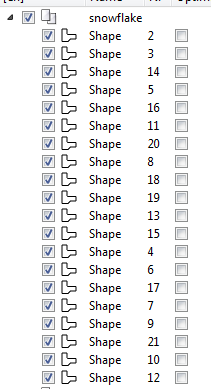


to cut one layer these are the settings used, note that retraction area was set to 1mm and infeed depth set to -0.2 and final mill depth set to -0.2, safety margin and workpiece top set to zero. Federate is ignored. The retraction area of 1.0 allow the router to be set just touching the surface at the beginning of routing then retract 1.0 before moving to first coordinate.





the optimal path contains the shape order:

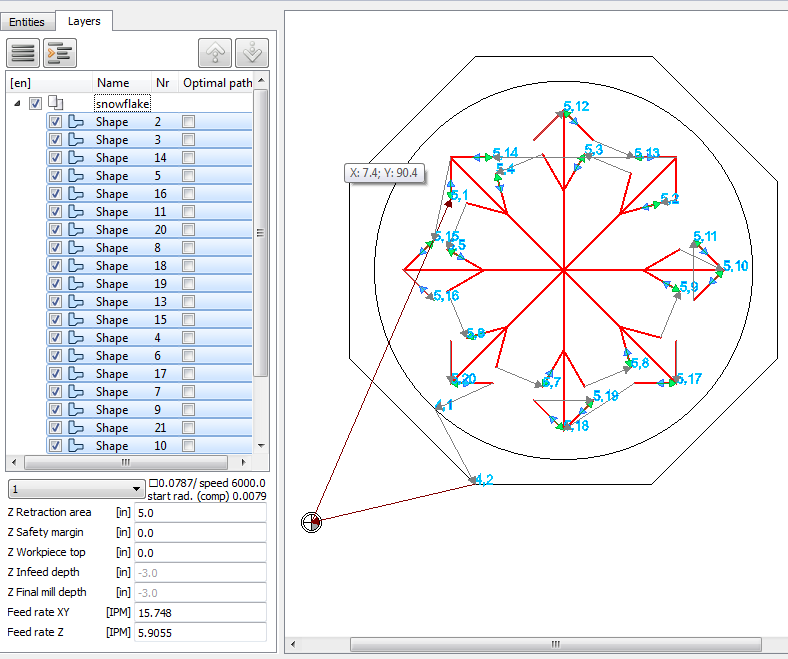


## 08/30/2015 test cut on foam

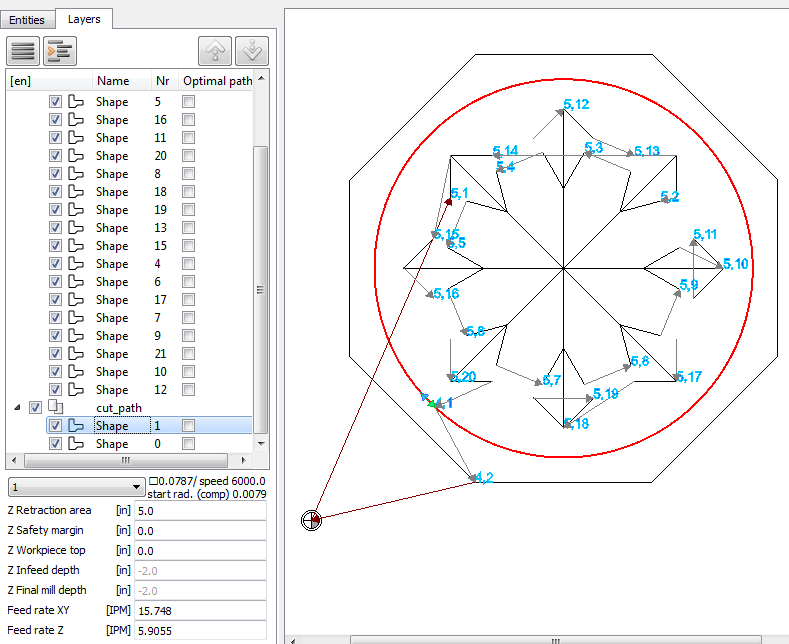
to test the depth of cuts I generated gcode for quickly cutting full depth in one sweep.

The following settings were used to generate a file called “snowflake\_full\_depth\_1.ngc.

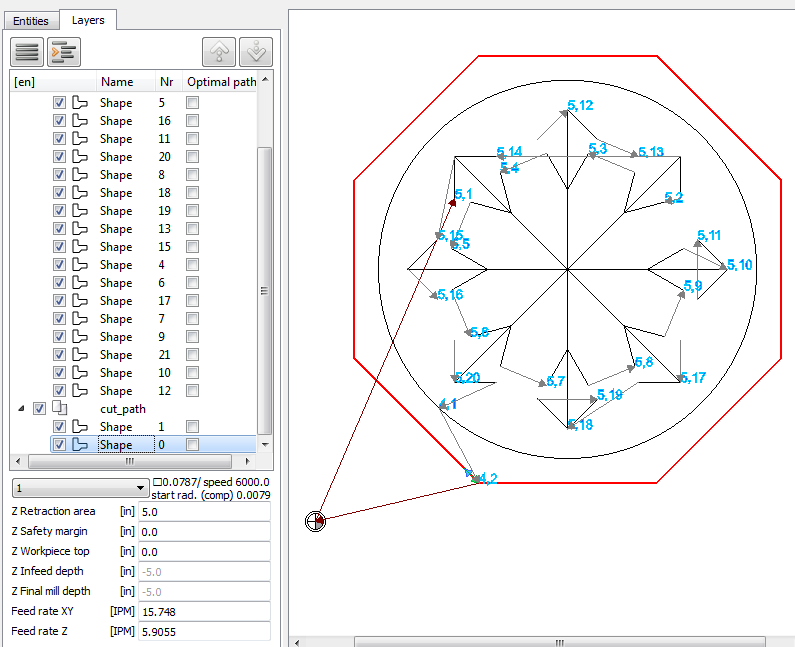
Note that the snowflake part has infeed/final mill depth of -3.0 ; so it does not cut all the way through the acrylic sheet (which has a thickness of -5.0 mm).



Note that the circle part has infeed/final mill depth of -2.0

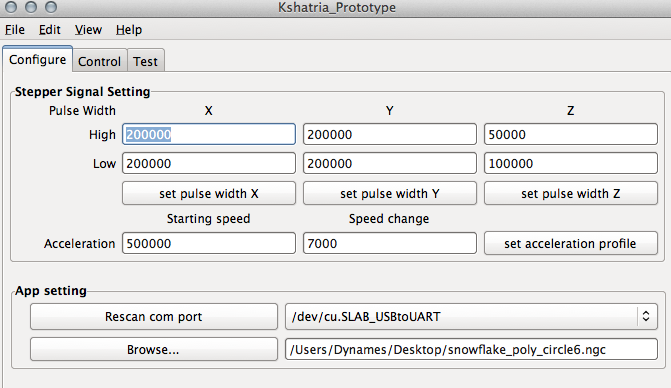


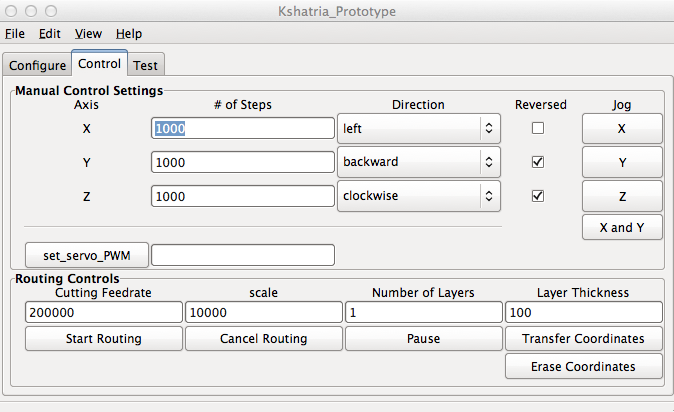
Note that the snowflake part has infeed/final mill depth of -5.0 ; I’ve set -5.0 because this should be the thickness of the acrylic sheet.



as of today only the z axis has acceleration enabled.

Settings in the GUI:





## 08/31/2015 optimal settings for cutting acrylic

use single flute spiral upcut bit,

signle flute does a better job of removing bits of cut acrylic away from the area.

Want depth 2mm for snowflake

Depth 3mm for circle

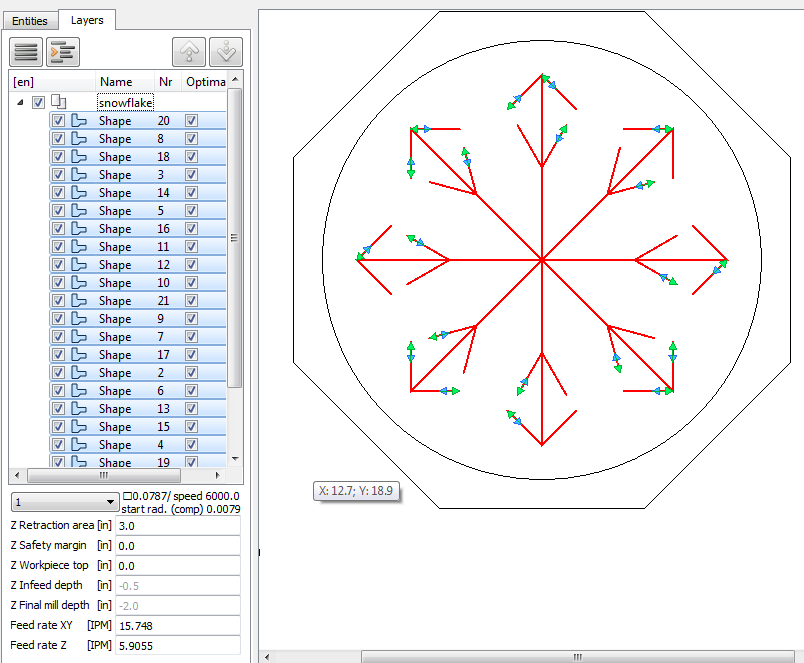
Depth 6mm for hexagon

Step down (Z Infeed depth) 0.5mm for all

Retraction 3.0mm for all

Gcode file: snowflake\_acrylic\_1.ngc

Settings used for snowflake,



## 09/07/2015 cnc routing leaf design

want depth 2mm for leaf

depth 2mm for circle

depth 5.5mm for hexagon

step down (Z Infeed depth) 0.5mm for leaf

stepdown (Z Infeed depth) 0.5mm for circle and hexagon

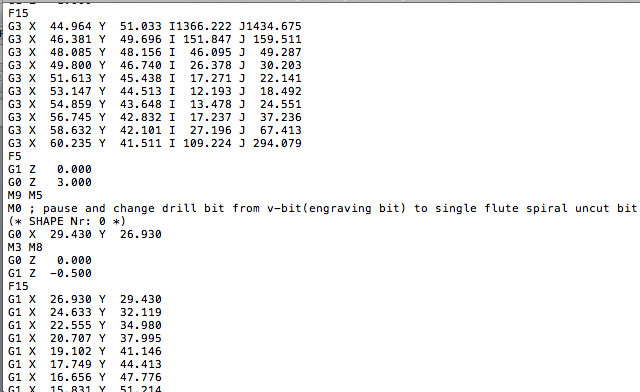
z retraction 5mm for all

z retraction 10mm for circle

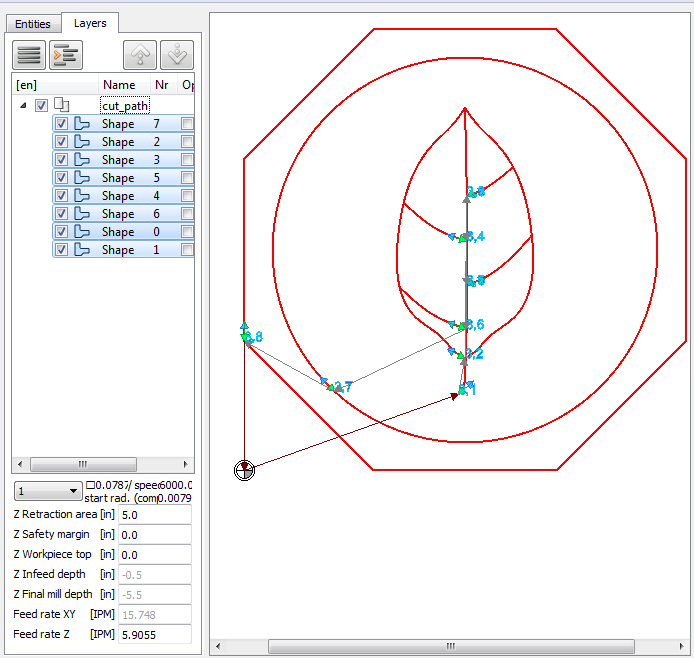
z retraction 3.0 mm for hexagon

gcode file: leaf\_origin\_1.ngc

~~modified generated gcode and inserted M0 command before shape 0 gcode to pause the router and allow time to change from engrooving bit to single flute spiral upcut bit.~~



note: M0 was later removed because the groove bit require a flatter surface. A future task will be to figure out a way to automate surface leveling.



to change the starting point of a plot, right click on the line move mouse close to where you want to start and select: “set nearest start point”

