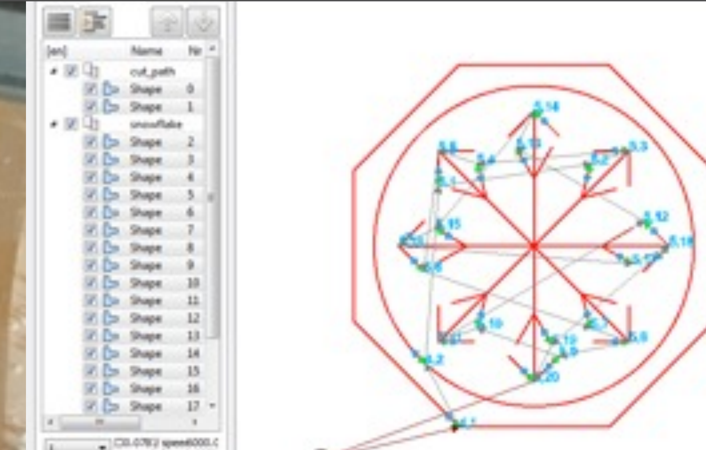
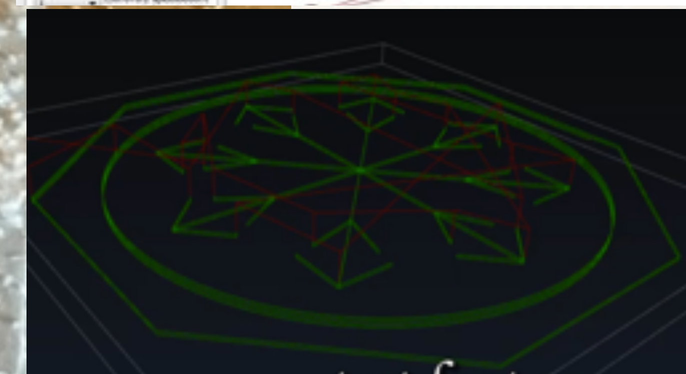




FPGA/Nios Based CNC Controller

https://github.com/mhouse1/mechsoftronic

10/28/2015

What is CNC?

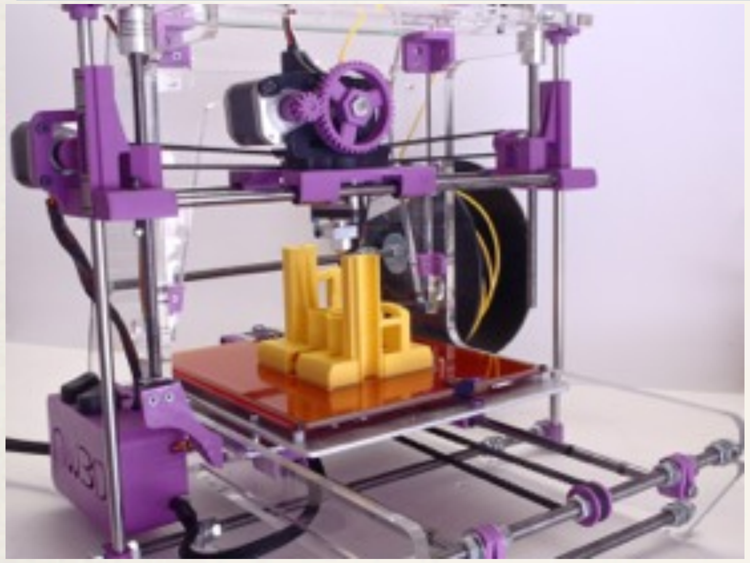
- Computer Numeric Control
  - Take drawing from CAD (Computer Aided Design) tools, convert to a series of numbers (GCode), control movement using GCode
- CNC Machine
  - Mechanical, software, and electronic components that make up a tool that can create objects designed in CAD
- CNC Controller
  - A device with firmware that interprets GCode and outputs motor control signals to control how an object is shaped

Sample GCode

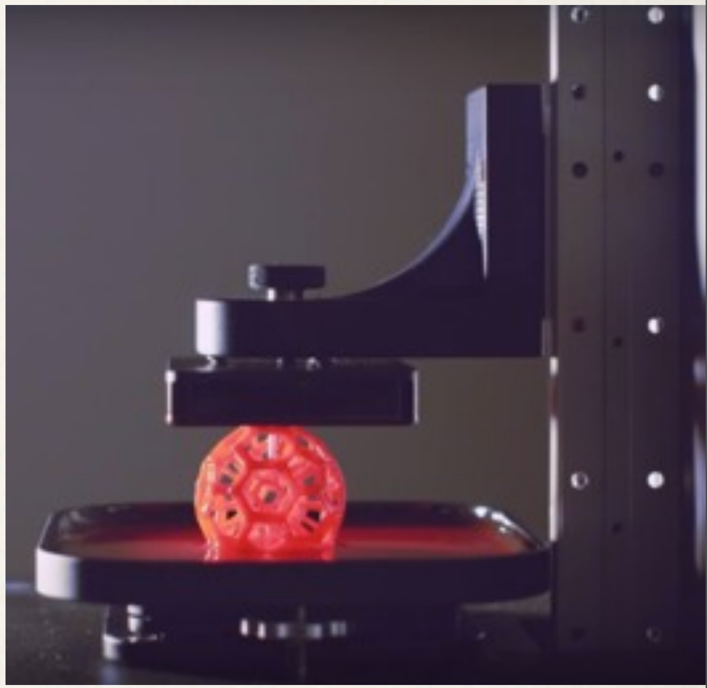
```
(Generated with: DXF2GCODE, Version: PyQt4 Beta, Date: $Date: Tue Jun 9 17:40:00 2015 +0200 $)
(Created from file: //psf/Home/Desktop/snowflake1.dxf)
(Time: Wed Jul 29 07:46:24 2015)
G20 (Units in inches) G90 (Absolute programming) G64 (Default cutting) G17 (XY plane) G40 (Canc
G0 Z   0.591

(*** LAYER: snowflake ***)
T1 M6
S6000

(* SHAPE Nr: 1 *)
G0 X  28.180 Y  28.180
M3 M8
G0 Z   0.000
F5
G1 Z  -0.500
F15
G2 X  91.820 Y  91.820 I  31.820 J  31.820
G2 X  28.180 Y  28.180 I -31.820 J -31.820
F5
G1 Z  -1.000
F15
G2 X  91.820 Y  91.820 I  31.820 J  31.820
G2 X  28.180 Y  28.180 I -31.820 J -31.820
F5
G1 Z  -1.500
F15
G2 X  91.820 Y  91.820 I  31.820 J  31.820
G2 X  28.180 Y  28.180 I -31.820 J -31.820
F5
G1 Z  -2.000
F15
G2 X  91.820 Y  91.820 I  31.820 J  31.820
G2 X  28.180 Y  28.180 I -31.820 J -31.820
F5
G1 Z   0.000
G0 Z   0.000
M9 M5

(* SHAPE Nr: 2 *)
G0 X  33.130 Y  76.870
M3 M8
```

3D Print vs Milling

what kind of output do you need?



- extrusion
  - layer by layer
  - low structural integrity
  - slow



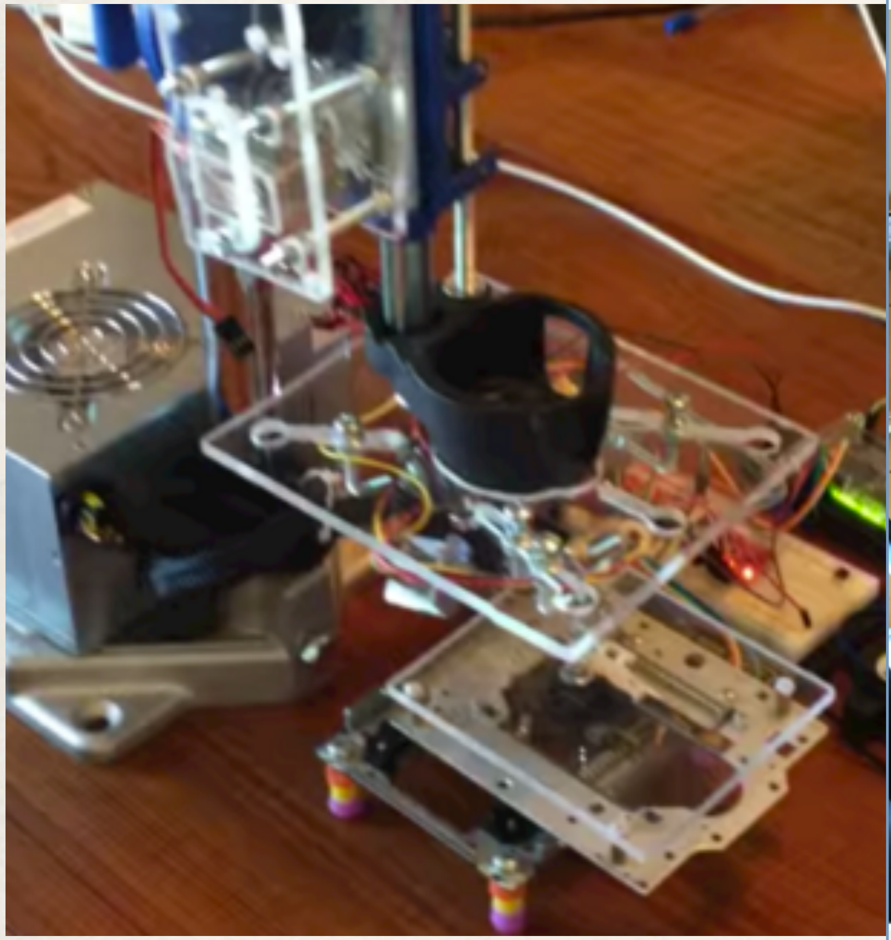
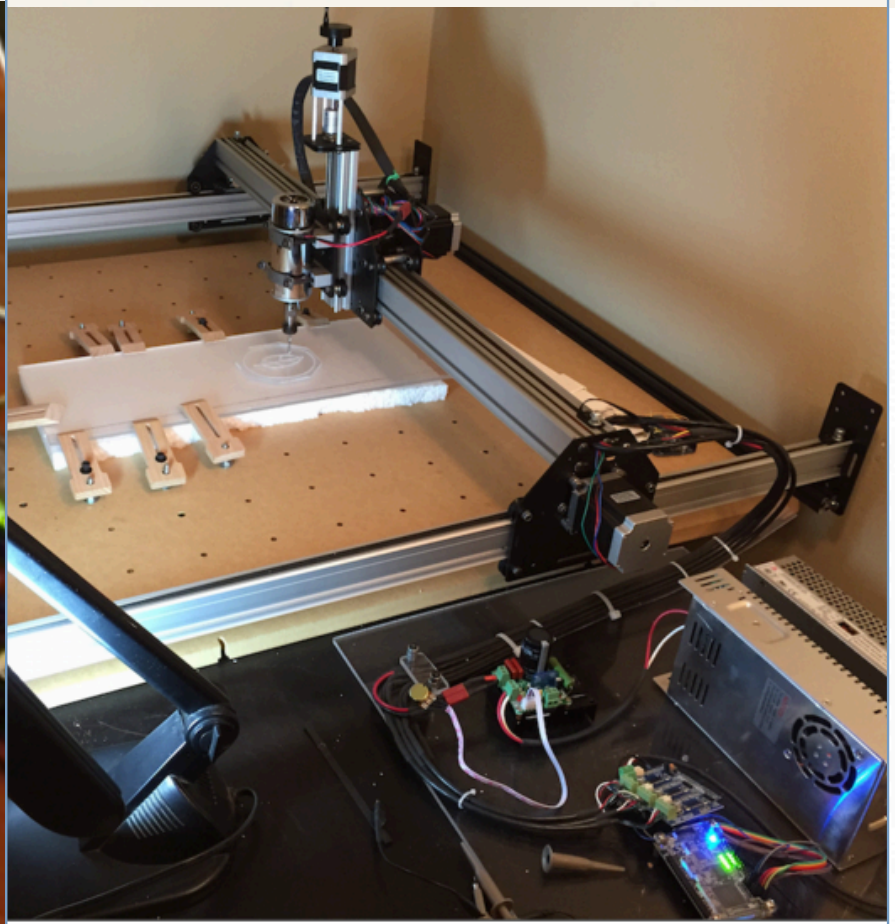
- photopolymerization:
  - layer by layer
  - better structural integrity
  - faster
  - create more complex shapes

- milling (what this project is)
  - high structural integrity: plastic, wood, metals
  - fast
  - less complex shapes

Project Goals

- Research, Design, and Build a CNC Machine
- Create CNC controller and make everything OpenSource
    - Use FPGA to create jitter free movement
    - great system integration by managing hardware and software
- End result is a tool usable at home to make designs from CAD quickly and accurately
- adaptable to any mechanical platform

Adaptable Mechanical Platform

MKI ($0)	MKII ($1000)	MKIII (~$1600)
		

Why not just buy one?

- Cost
  - high cost even for a CNC mill with small milling area
- Customizable
  - add on laser cutter, plastic extruder, plasma cutter, exacto knife, ... anything you want
- Full understanding of how the machine works and its limits

- base system for future use
  - FPGA based systems are complex and painful to setup but is powerful; this project will be used as a starting point for future projects

Design

Linear motion hardware			
ball screw	acme	rack and pinion	belt drive
✶almost no backlash ✶low friction (good for moving but bad for stopped) ✶will tend to rotate away from loads ✶must use oversized loads to mitigate back driving loads ✶expensive	✶will hold a load in place by design ✶backlash; unexpected play due to clearance or looseness of mechanical parts ✶limited size ✶not so expensive	✶easily expandable to larger machines ✶complex; requires proper gear reduction and other hardware to eliminate radial load on the motor shaft ✶can become expensive if performance desired	✶can accommodate semi large machine ✶simple ✶not so great at high speeds ✶typically used by hobbyists because its one of the cheaper options for performance

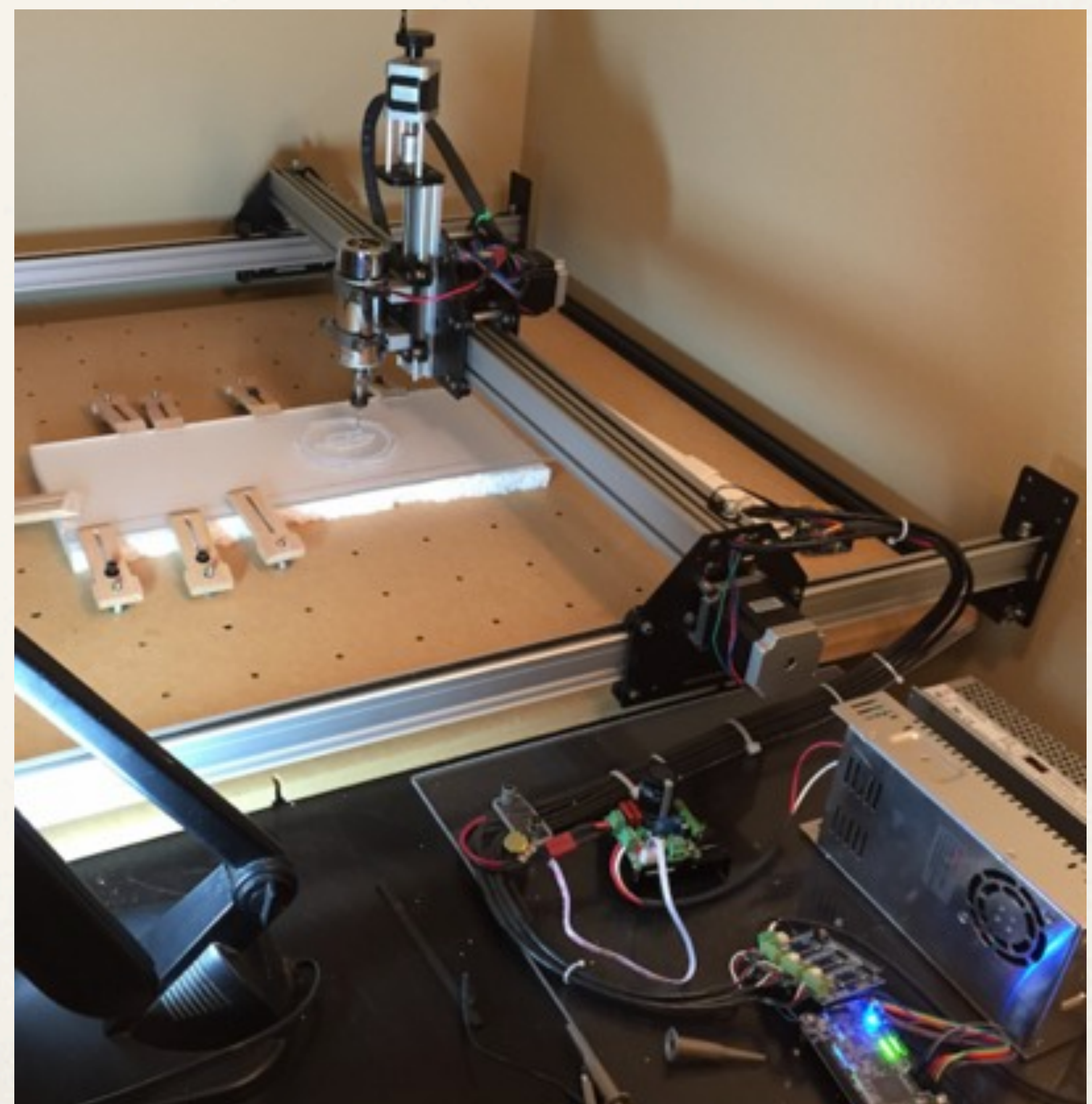



Hardware

Modified OpenHardware Shapeoko 2 1000mm x 1000mm rails

	A	B	C
1	Mechanics Summary		
2	**Item**	**Qty**	**Total**
3	Stepper Motor - NEMA 17	1	$16.94
4	MakerSlide	4	$87.36
5	Flat Washer	2	$5.00
6	Socket Head Cap Screw	2	$6.18
7	Socket Head Cap Screw	1	$3.14
8	Hex Nuts	1	$2.00
9	Flat Washer	2	$4.00
10	Aluminum Spacers	2	$5.22
11	GT2 Belting - Open Ended	15	$29.85
12	Aluminum GT2 Pulley	3	$19.41
13	End Mill Starter Set, 1/8 in Shank	2	$39.90
14	Desktop 3D Carving (CNC Mill) Kit - Shapeoko 2	1	$300.00
15	Button Head Cap Screw	2	$4.70
16	Knob with Crank	1	$8.40
17	Clamp Set	2	$59.80
18	Carbide V Bit	1	$13.50
19	Carbide V Bit	1	$13.50
20	Microswitch	6	$15.00
21	Enclosed Power Supply	1	$42.25
22	ER11-A Collets, Nuts and Wrenches	1	$6.65
23	ER11-A Collets, Nuts and Wrenches	1	$5.84
24	Quiet Cut Spindle	1	$79.85
25	Spindle Speed Controller	1	$35.00
26	Enclosed Power Supply	1	$64.75
27	Waste Board with Threaded Inserts	1	$103.00
28	gShield	1	$49.99
29	Stepper Motor - NEMA 23	3	$117.00
30	Subtotal:	$1,138.23	
31	Shipping Cost:	$106.93	
32	**Grand Total:**	**$1,245.16**	

- Altera DE-0 Nano development board
  - Using Nios II/e core (free)
- gShield OpenHardware stepper motor driver



note: quantity above indicate number of sets

Software:Firmware

Table below lists program size for various project configurations

Firmware type	Program Size (KBytes)
Total size available	32736
CNC c project	31
CNC c++ project	364
MicroC/OS II in CNC c++ replace iostream with stdio.h	432
MicroC/OS II hello_ucosii in C	101

2014

Today, 2015

```
Info: (RAPTOR_03.elf) 457 KBytes program size (code + initialized data).
Info:                 32311 KBytes free for stack + heap.
```

Software:Firmware

Task1

- make sure FIFO don't over flow
  - read serial data from UART FIFO
  - queue data for task 2 to consume
  - sender will be required to send gcode at a slower rate than commands

Task2

- read queued data from task 1 and convert to data packets
- if threshold # of data packets received, signal sender to stop sending
- process data packets
  - either commands
  - or GCode
- GCode is interpreted into steps and dir and queued for Task 3 to consume
- Commands are processed right away
  - see GUI for commands available

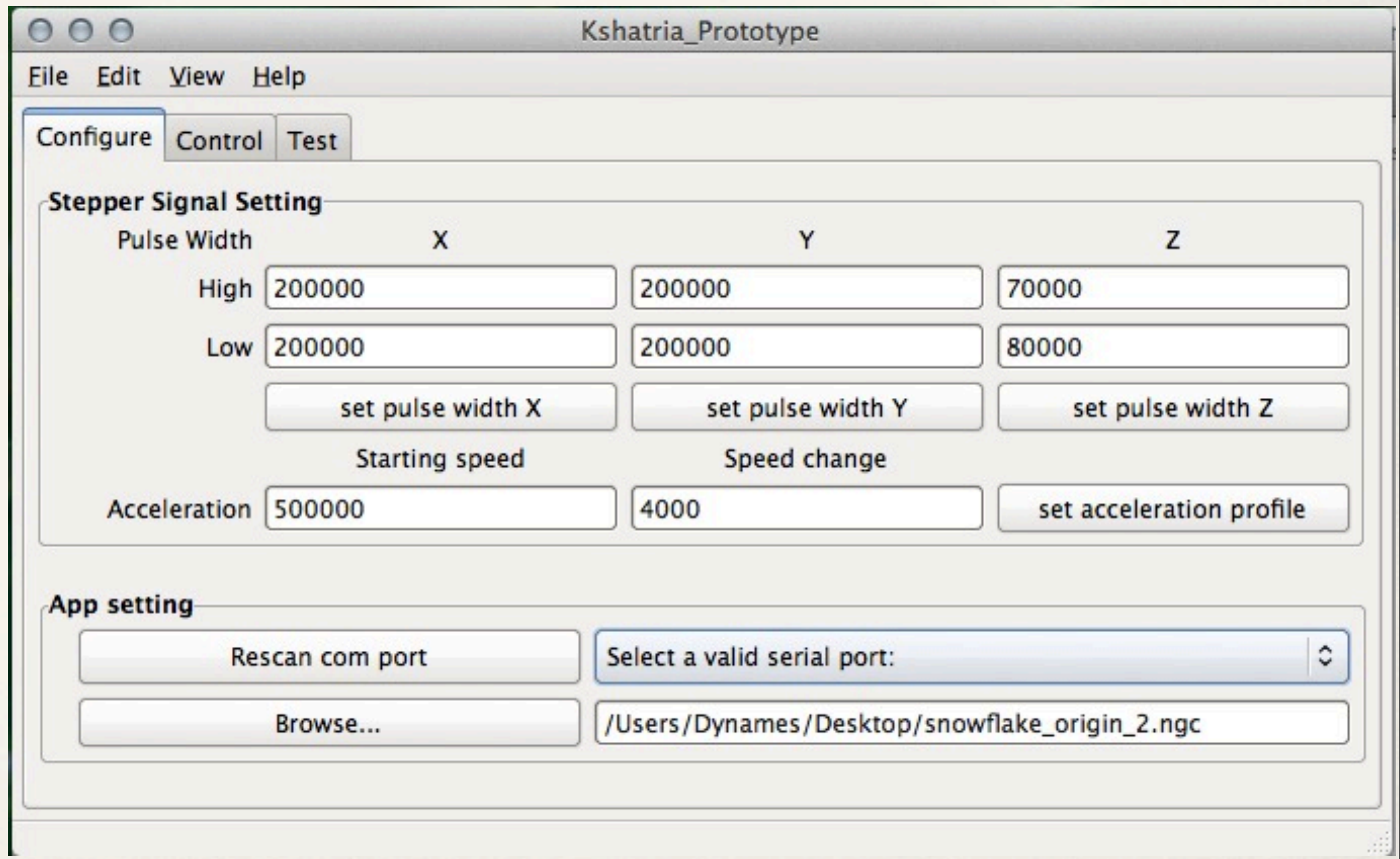
Task3

- dequeues step and direction and writes to hardware for FPGA modules to determine acceleration and synchronization

The 3 tasks run in parallel on MicroC OS II RTOS, this was done because the FPGA is configured with the "free" Nios II/e CPU core where vectored interrupts are not available.

Software:GUI

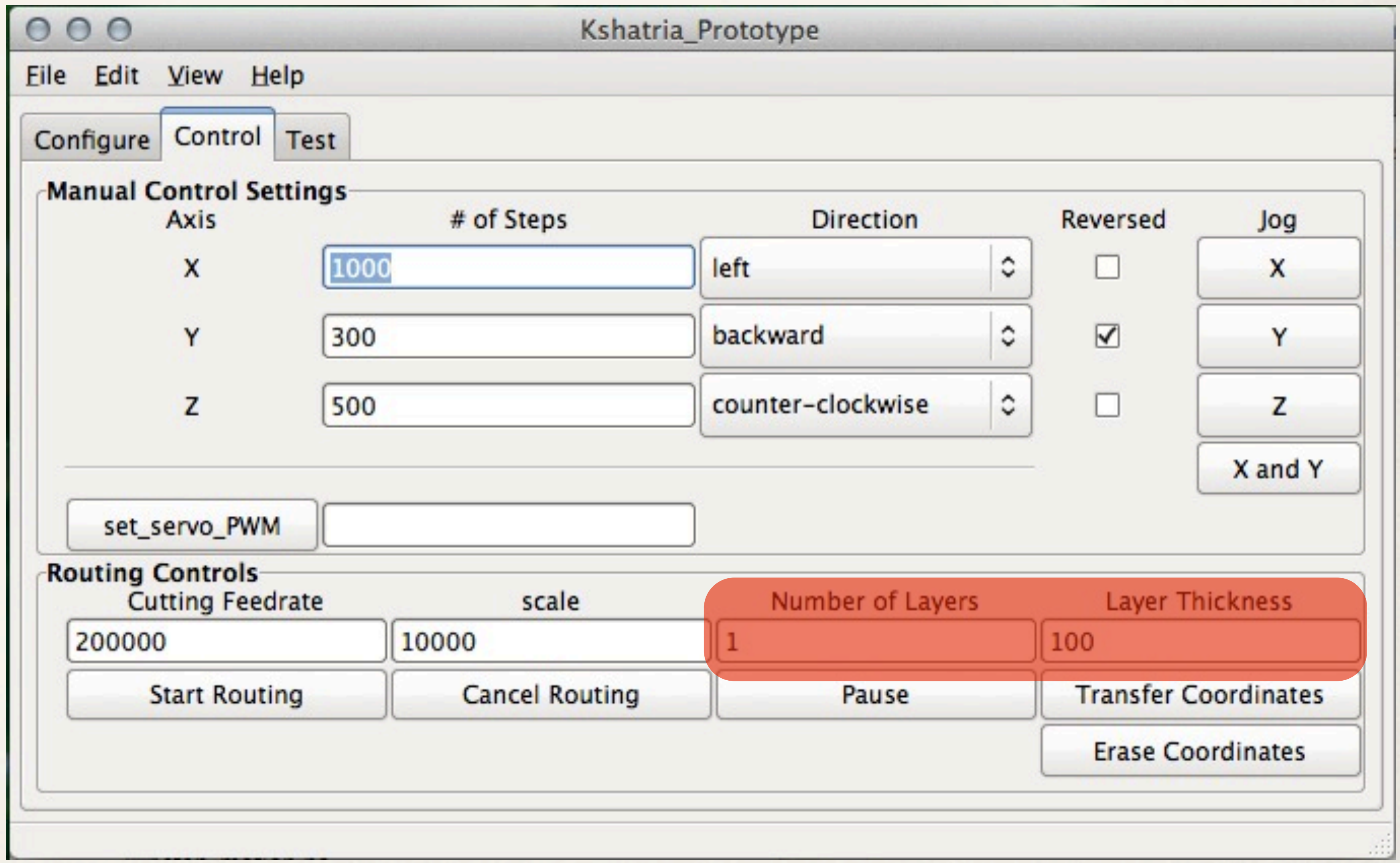
- configure nominal pulse settings for 1 step
    - where 1 step is a square wave
    - each count = 20ns
    - so x-axis in this case would output 4ms of high and 4ms of low per step
- acceleration settings only applies to z-axis



Can also select serial com-port and GCode file to send

Software:GUI

- jog single axis specified number of steps in specified direction
- jog X and Y axis at the same time
- check reverse to reverse axis in software
- PWM will control routing spindle speed (not yet implemented)
- scale will increase drawing scale
- feed-rate will determine maximum speed of the router
- other commands: start, cancel, pause, erase route



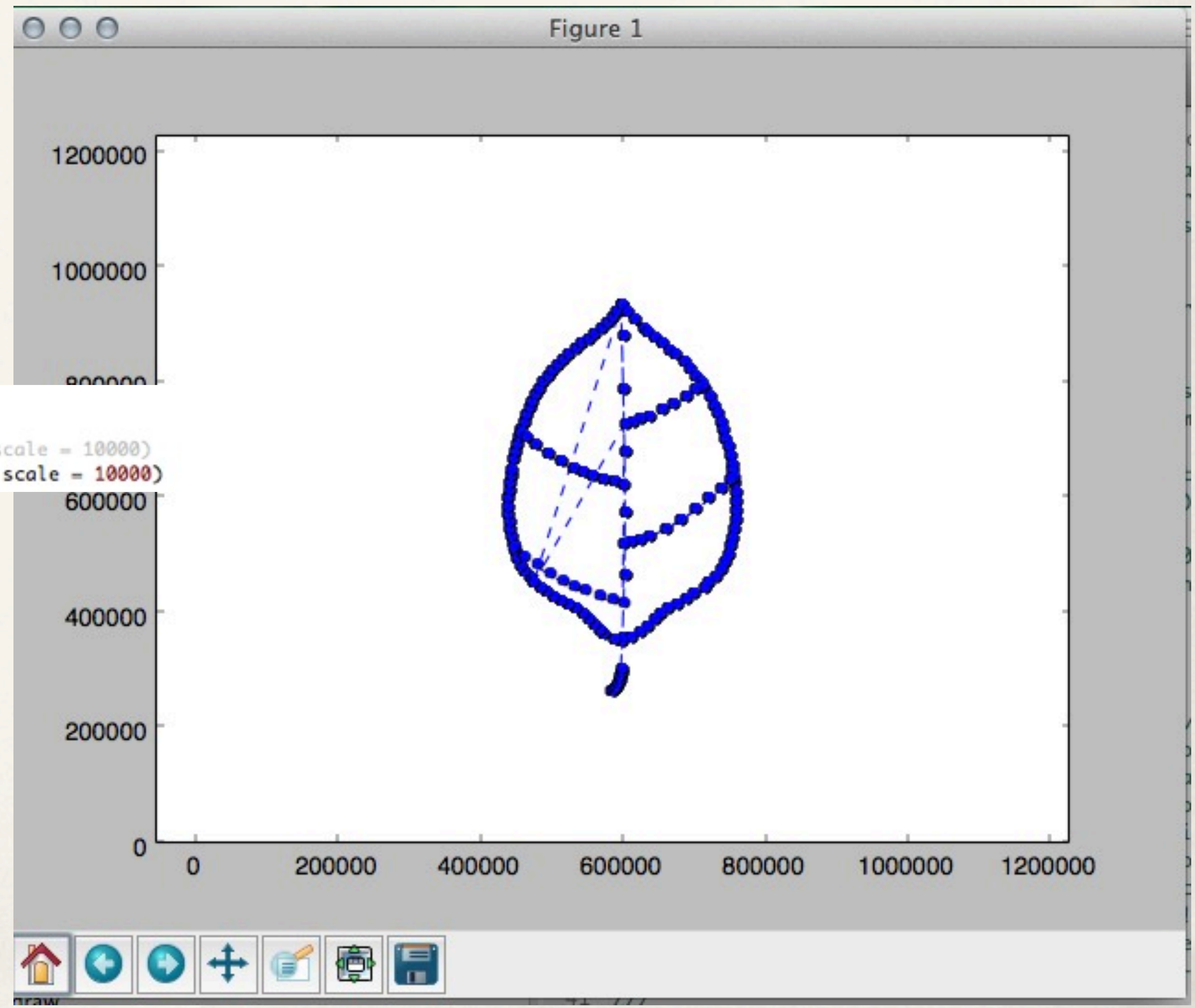
note: highlighted in red is no longer needed and to be removed

Software:GUI

- The parse_gcode.py module doubles as an preview tool if ran individually
  - allows user to preview points with respect to time and position

- To use:
  - modify GCode file name

```
if __name__ == '__main__':
    #get a set of coordinates
    #xycoord = get_gcode_data('snowflake6optimized.ngc', scale = 10000)
    xycoord = get_gcode_data('leaf_engrooving_bit_1.ngc', scale = 10000)
```

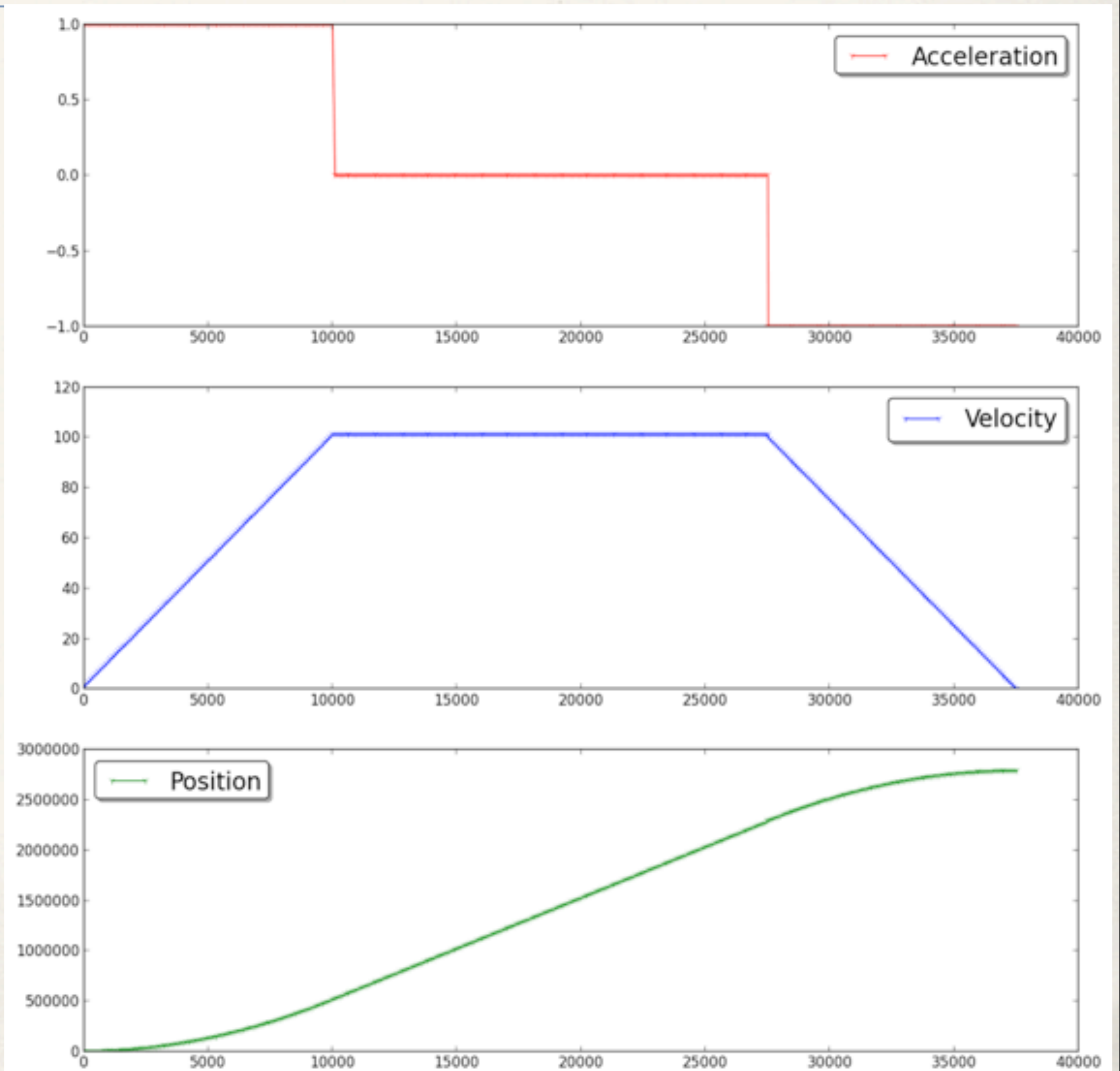


FPGA: special components

- non-standard altera modules (see project log to learn how to install)
  - slave_template
    - memory mapped registers, firmware can read / write to these and FPGA can read / write to it too. used by firmware to control FPGA motion control modules, and other IO
  - FIFOed UART
    - a modified version of the Altera UART module with the added ability to buffer UART data
  - PWM generator
    - used for controling RC servos, and router speed controller
  - pulse generator
    - used to generate step and dir signal that goes to the stepper motor driver

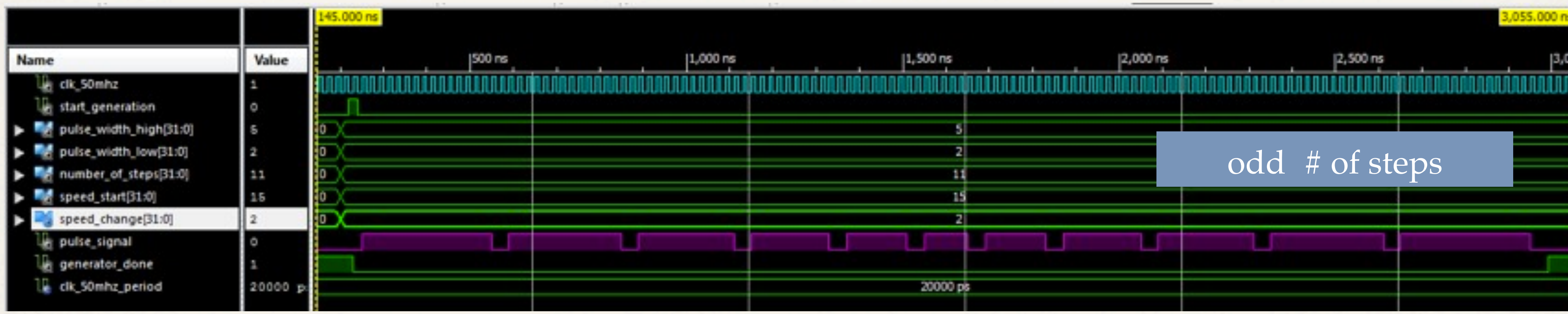
FPGA: motion generator Z-axis

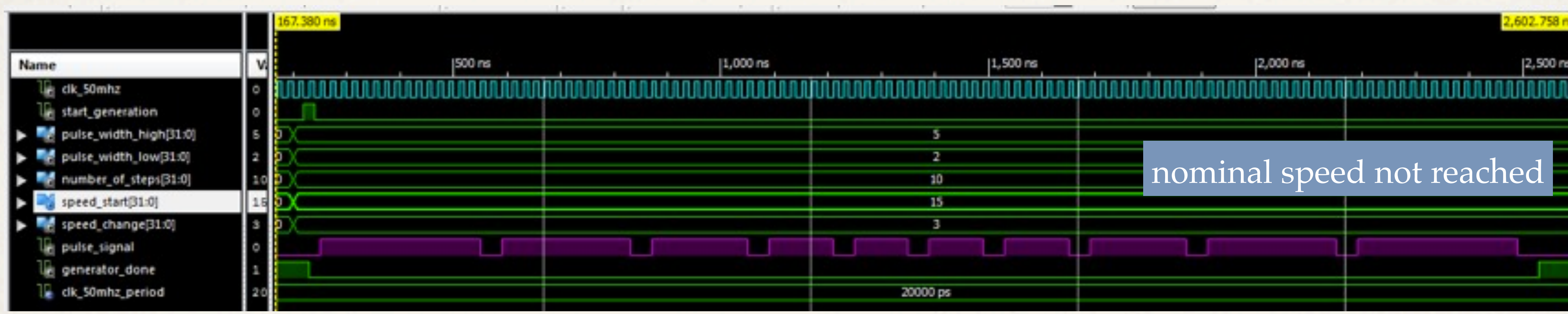
- z-axis acceleration is handled by the FPGA given a nominal speed and a speed change, the curve generated is similar to simulation shown (left)
- This had to be redesigned for XY axis to work properly

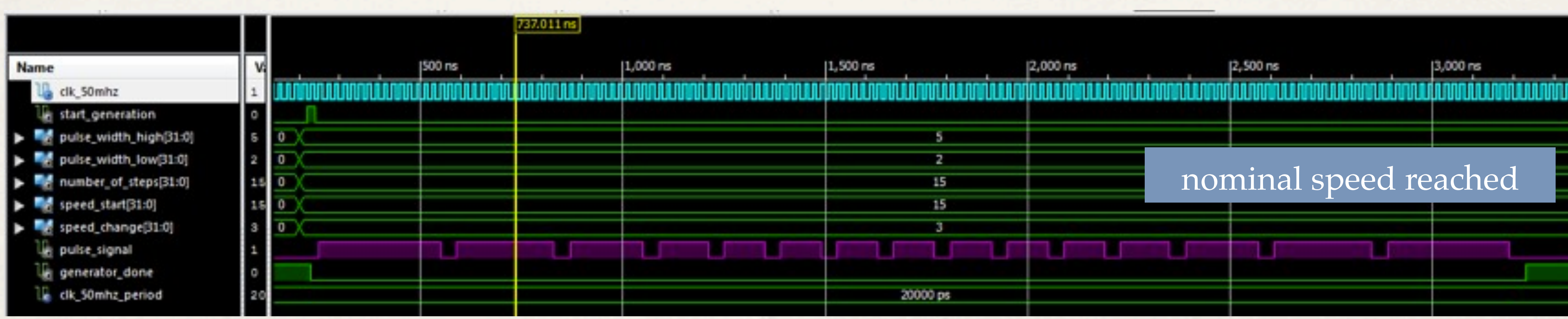


FPGA: motion generator Z-axis

tested using Xilinx test bench







FPGA: motion generator XY axis

- Design and implementation in progress
- currently limited to no acceleration
  - this means xy axis should move much slower
- redesign is basically a clock tick that can be sped up or slowed down used to synchronize xy axis while maintaining acceleration

```
///    FPGA motion control:
///        The design requires a xy movement to be synchronized. This means steps traveled on one axis
///        must be proportional to steps traveled on the other axis even while accelerating and deceleration.
///        number of steps and pulse width constants will be loaded into FPGA registers, the FPGA module would
///        normally generate the steps with a fixed pulse width clocked at a constant clock signal. To convert
///        this fixed pulse width output into something capable of acceleration, speed hold, and deceleration
///        an enable signal will be used to ignore some clock ticks.
///        For example: a clock may have a normal tick (! = 1 clock tick) interval of
///        [--------------------------------constant speed--------------------------------]
///        ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! !
///        We want the clock to look like the following in order to implement acceleration/deceleration.
///        [-------acceleration---------][-----speed_hold-----][---------deceleration----------]
///
///        !      !      !     !    !   ! ! ! ! ! ! ! ! ! !   !    !     !      !       !
///        The above is what i call an accordion clock, it has 3 states: acceleration, speed_hold, and deceleration
///        Luckily, the FPGA module does not have to do all permutations of the 3 states.
///        The order will be one of these state orders:
///        where a = acceleration, h = speed_hold, d = deceleration
///        a     <- acceleration for during output of all steps
///        h     <- speed_hold for during output of all steps
///        d     <- deceleration for during output of all steps
///        a h d <- acceleration to speed hold to deceleration
///        a h   <- acceleration to speed hold
///        h d   <- speed hold to deceleration
///
///        we don't need to go from states like deceleration to acceleration, because that would mean there's a direction
///        change, in that case we'd get a new set of steps to output.
///
///        motion control input:
///            clock
///            enable
///            number of steps to accelerate
///            number of steps to hold_speed
///            number of steps to decelerate
///            axi1_total #of steps to travel
///            axi1_direction_axis
///            axi2_total #of steps to travel
///            axi2_direction_axis
///            master_axi
///
///        motion control output:
///            step_signal
///            direction
```

Project Status

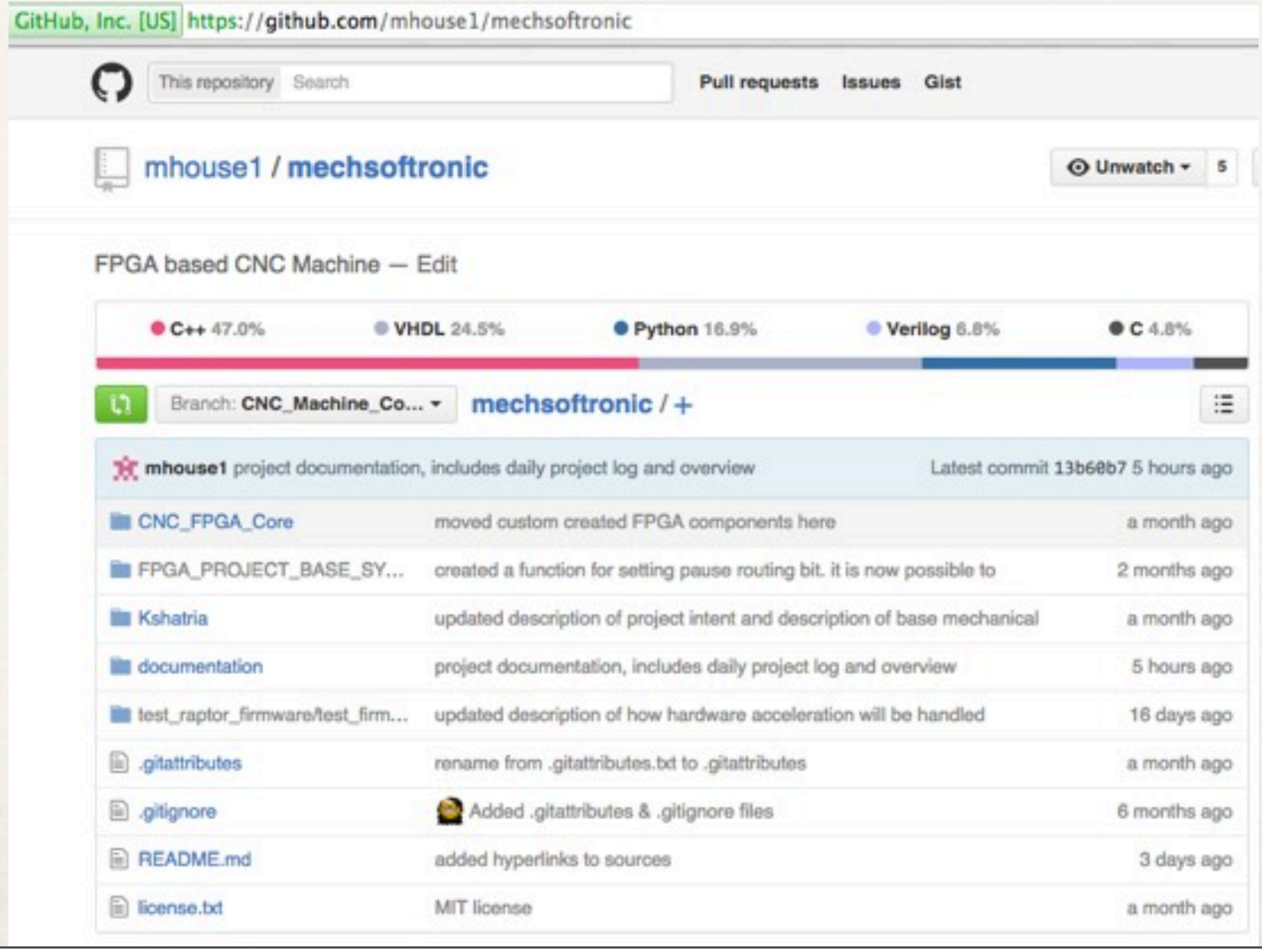
- ✓ Able to go from CAD to numbers to firmware to finished product
  - ✓ Z-axis acceleration working
  - ✓ XY-axis must move at slower speeds until synchronized acceleration implemented
- ★ As of 4/23/2014 project been OpenSourced on GitHub
  - ★ 5 users watch project
  - ★ starred 17 times
  - ★ forked once

Future work

- implement synchronized XY axes acceleration motion
  - estimated completion date 07/01/2016
- implement firmware GCode look ahead algorithm
  - estimated completion date 03/01/2016
- move controller to MKIII mechanical platform
  - TBD

Everything documented

- See GitHub mechsoftronic documentation for more details
  - all things relating to the project design and work has been documented



Everything documented

- From May, 2013 to today

CONTENTS

Tools

- This CNC will support drawings from AutoCAD
  - draw object in autoCAD, export as *.DXF file
  - hint: draw different categories on its own layer
- Convert DXF to GCode using DXF2GCODE (OpenSource software)
- Another useful GCode viewing tool is OpenSCAM
  - it does not allow you to view points like parse_gcode.py does but it does allow you to view a simulated routing