

G-Code, GCode, gcode

CNC machine codes, software tools, and music(?!?)

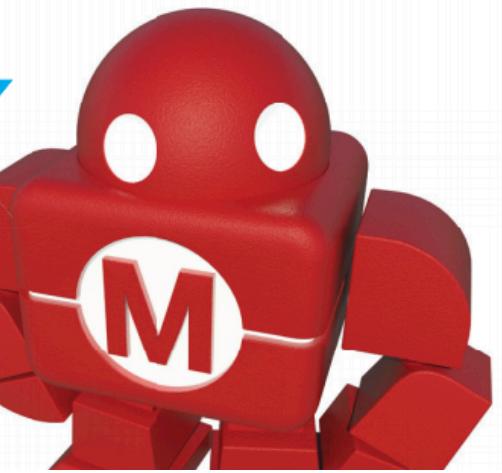
Announcements

- **Bay Area Maker Faire** (May 17-19)
- **Thursday:**
 - Recap of Assignment #1 submissions
 - Lecture: Computer-Aided Manufacturing (CAM)
- **Next Tuesday, April 30: Short project pitches**
 - Project Proposals due **May 9**
- **Assignment #3: T-Bot**
 - Optional: limit switches
 - Optional: design a holder for a pen, etc.
 - Bonus: Improve design
 - e.g., reduce "binding" forces



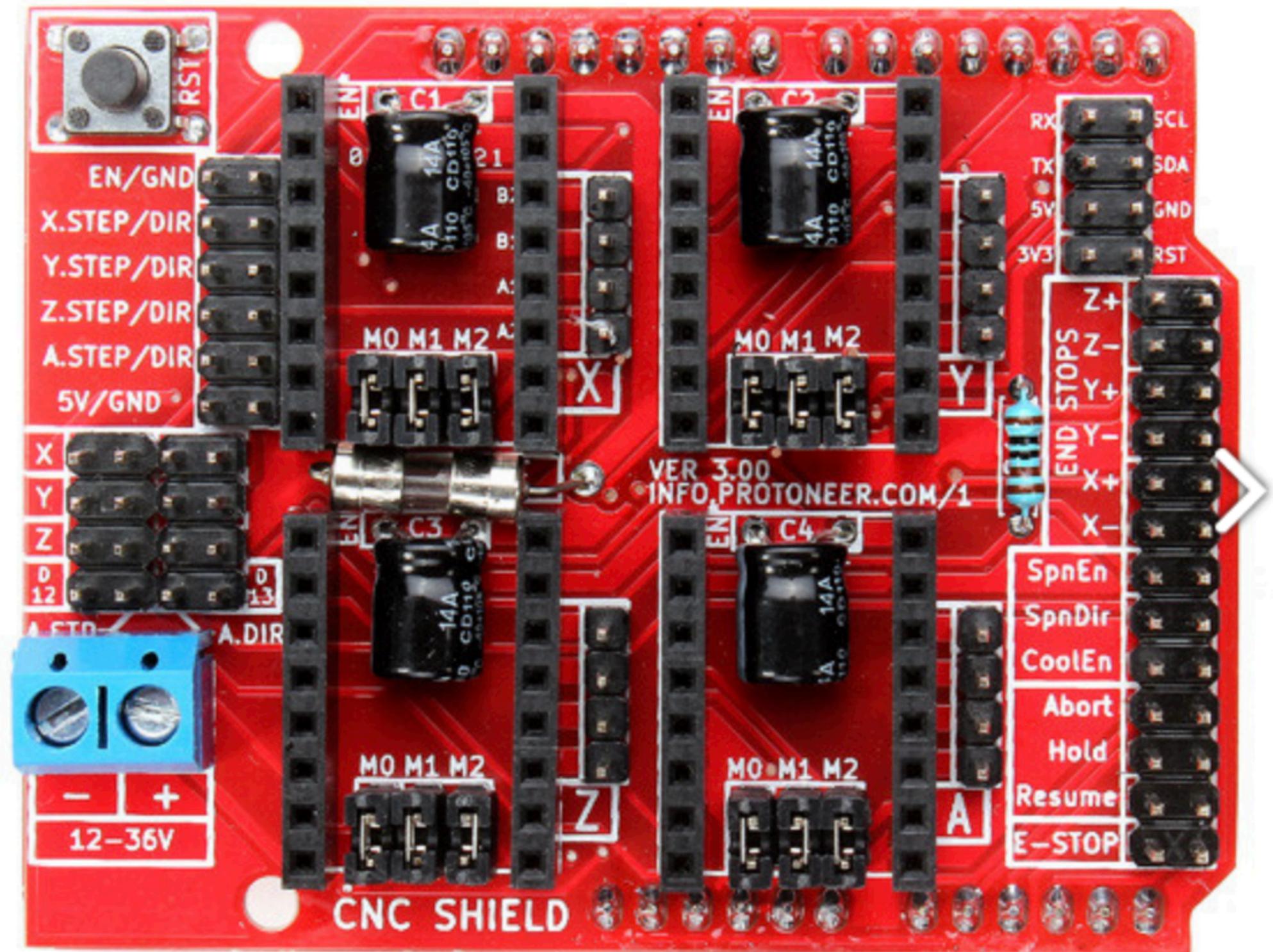
THE
FUTURE
WE MAKE

Maker Faire® MAY 17–19
TICKETS GET YOURS TODAY! makerfaire.com
FRI 1–5PM, SAT 10AM–7PM, SUN 10AM–6PM
BAY AREA SAN MATEO EVENT CENTER

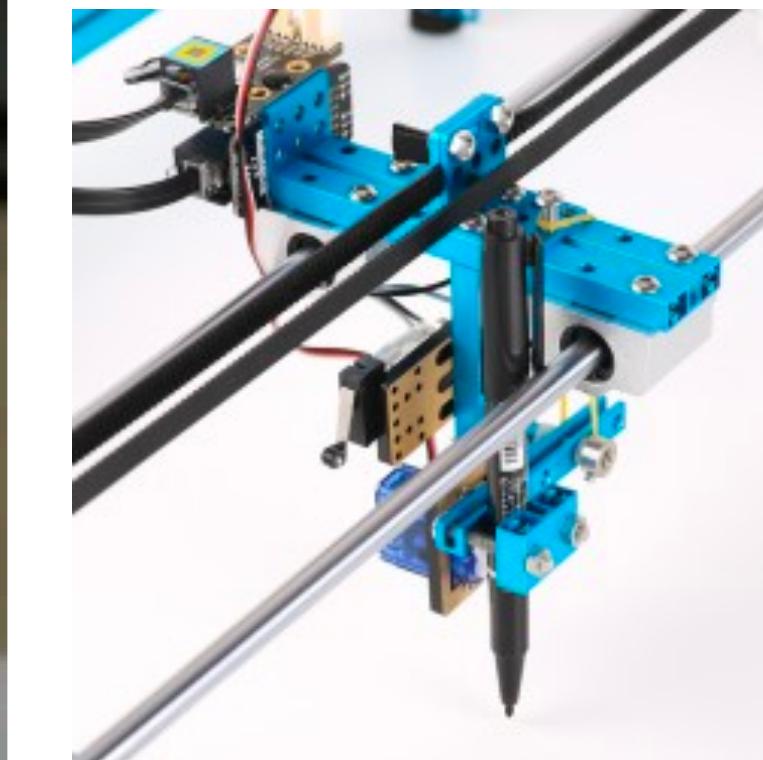
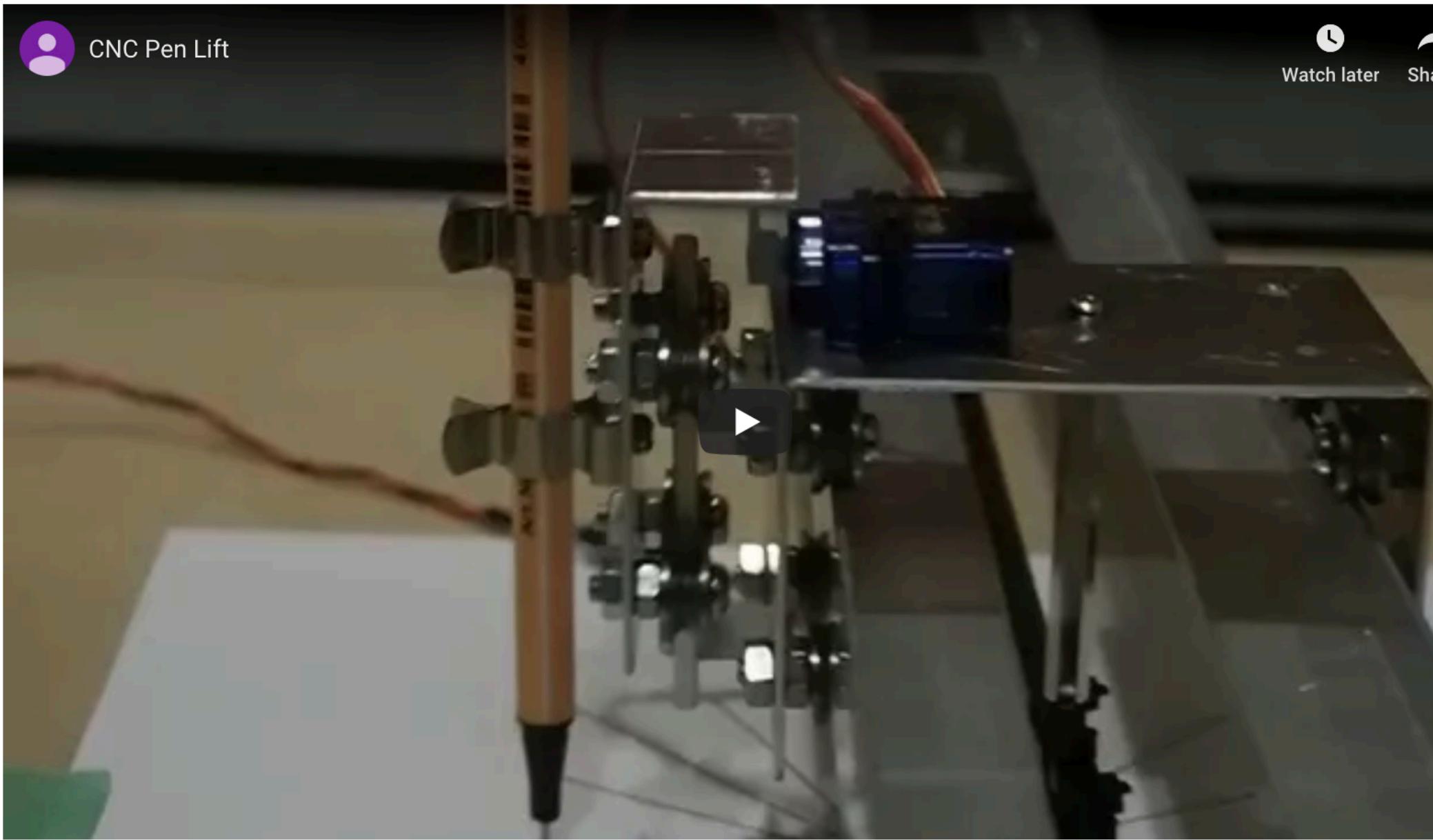


Re: Limit Switches

- Wire them up to your CNC SHIELD
- Take a look at grbl settings:
 - \$21 - Hard limits, boolean
 - \$5 - Limit pins invert, boolean
- General info on wiring limit switches:
 - <https://github.com/gnea/grbl/wiki/Wiring-Limit-Switches>



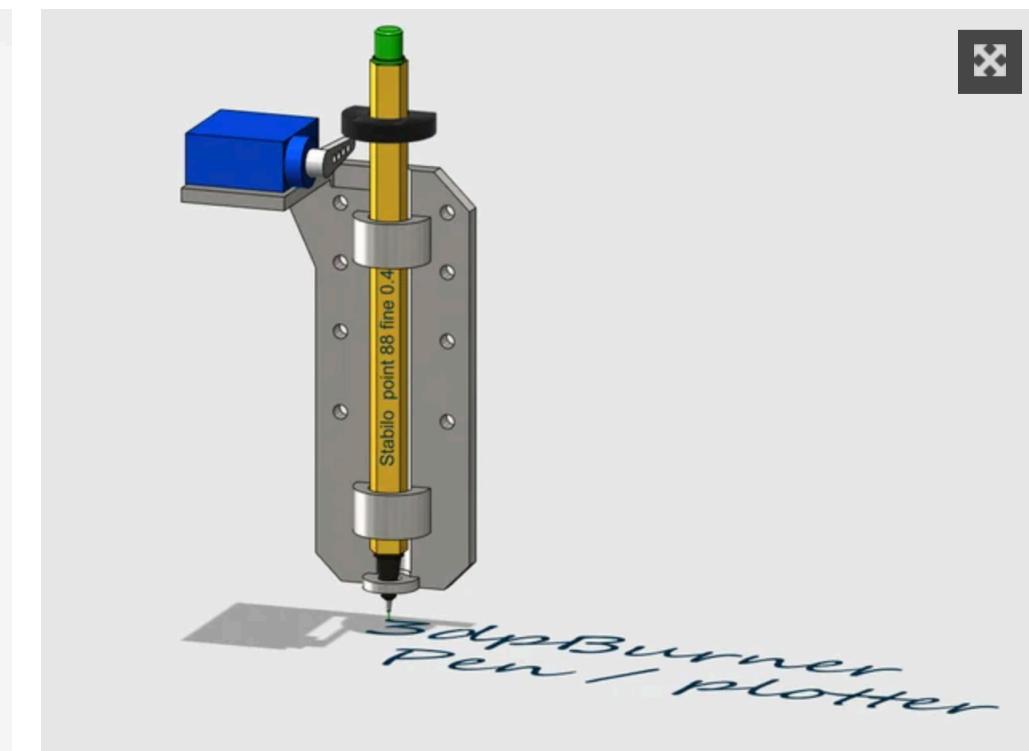
Some Pen-Holder Designs



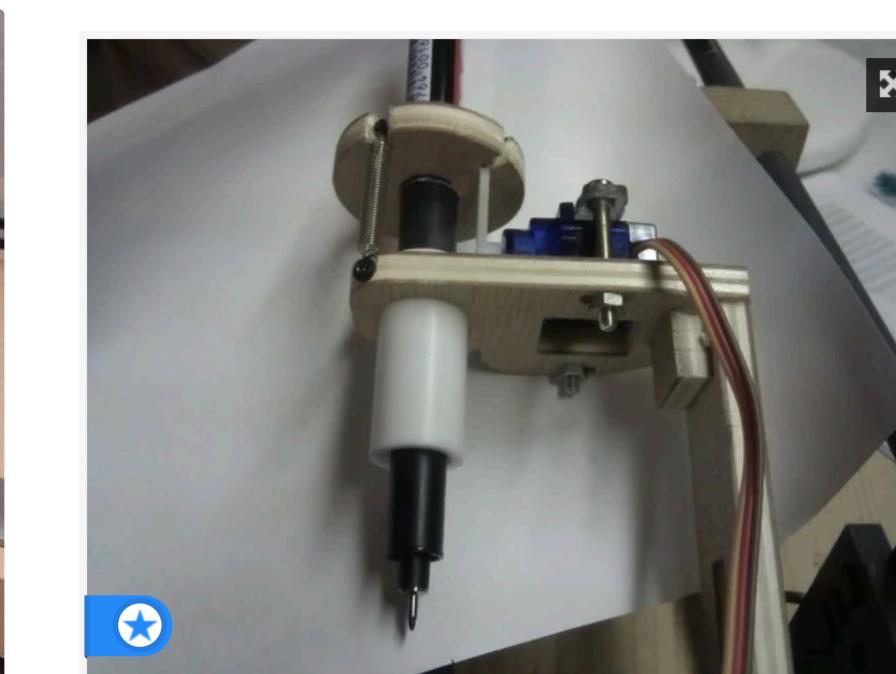
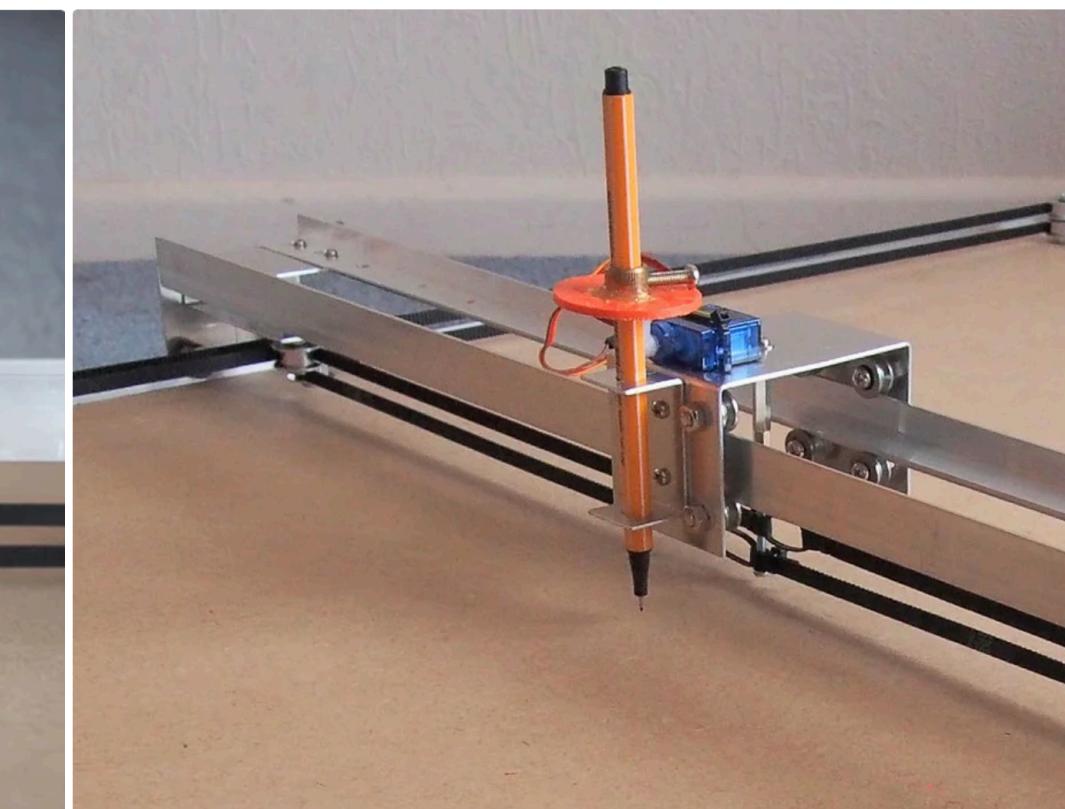
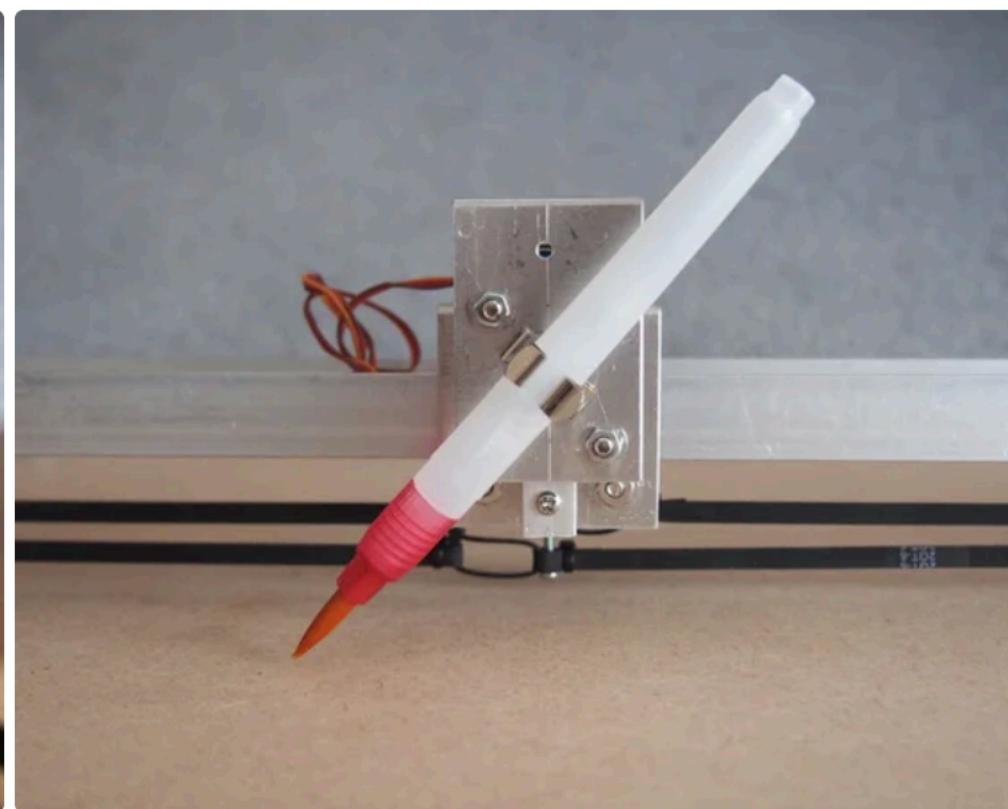
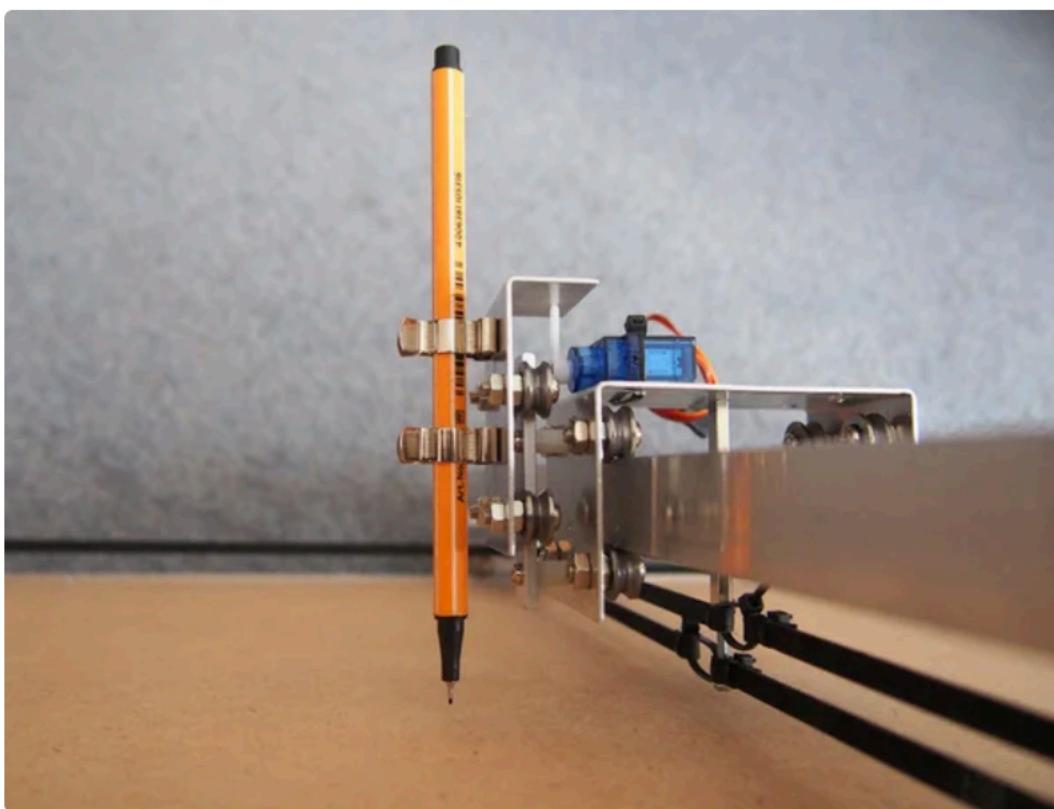
[https://www.makeblock.com/
project/xy-plotter-robot-kit](https://www.makeblock.com/project/xy-plotter-robot-kit)



<https://www.axidraw.com>



<https://www.thingiverse.com/thing:1451492>



<https://www.thingiverse.com/thing:13407>



<https://create.arduino.cc/projecthub/Yogeshmodi/sketch-it-cnc-plotter-95019d>

<https://www.instructables.com/id/CNC-Pen-Lift>

G-Code, GCode, gcode

CNC machine codes, software tools, and music(?!?)

Why use G-Code to make things?

We could write CNC code like this...

```
// The following is a simple stepper motor control procedures,  
  
# define EN 8 // stepper motor enable , active low  
# define X_DIR 5 // X -axis stepper motor direction control  
# define Y_DIR 6 // y -axis stepper motor direction control  
# define Z_DIR 7 // z axis stepper motor direction control  
# define X_STP 2 // x -axis stepper control  
# define Y_STP 3 // y -axis stepper control  
# define Z_STP 4 // z -axis stepper control  
/*  
// Function : step . function: to control the direction of the stepper motor , the number of steps .  
// Parameters : dir direction control , dirPin corresponding stepper motor DIR pin , stepperPin corresponding stepper motor " step " pin , Step number of step of no return value.  
*/  
void step (boolean dir, byte dirPin, byte stepperPin, int steps)  
{  
    digitalWrite (dirPin, dir);  
    delay (50);  
    for (int i = 0; i < steps; i++) {  
        digitalWrite (stepperPin, HIGH);  
        delayMicroseconds (800);  
        digitalWrite (stepperPin, LOW);  
        delayMicroseconds (800);  
    }  
}  
void setup () { // The stepper motor used in the IO pin is set to output  
    pinMode (X_DIR, OUTPUT); pinMode (X_STP, OUTPUT);  
    pinMode (Y_DIR, OUTPUT); pinMode (Y_STP, OUTPUT);  
    pinMode (Z_DIR, OUTPUT); pinMode (Z_STP, OUTPUT);  
    pinMode (EN, OUTPUT);  
    digitalWrite (EN, LOW);  
}  
void loop () {  
    step (false, X_DIR, X_STP, 200); // X axis motor reverse 1 ring, the 200 step is a circle.  
    step (false, Y_DIR, Y_STP, 200); // y axis motor reverse 1 ring, the 200 step is a circle.  
    step (false, Z_DIR, Z_STP, 200); // z axis motor reverse 1 ring, the 200 step is a circle.  
    delay (1000);  
    step (true, X_DIR, X_STP, 200); // X axis motor forward 1 laps, the 200 step is a circle.  
    step (true, Y_DIR, Y_STP, 200); // y axis motor forward 1 laps, the 200 step is a circle.  
    step (true, Z_DIR, Z_STP, 200); // z axis motor forward 1 laps, the 200 step is a circle.  
    delay (1000);  
}
```

Why is this not great for makers?

G-Code

- Most common language for CNC machines
- Developed in '50s at MIT Servomechanisms Lab
- Simple commands for simple machines
- Many common codes that are universal to most machines / parsers
- Many machine-specific code interpretations, (ISO) standards, and G-Code dialects
- <https://en.wikipedia.org/wiki/G-code>



"This simple aluminum ashtray represents a revolution in the machine tool industry. It was produced in 1959 as part of a demonstration of a milling machine controlled by a computer punch tape instead of a human operator. The development of this machine was more than a decade in the making and the result of a complex story about competing visions for this technology. After World War II, the U.S. Air Force gave several contracts to the Parsons Corporation to develop further the numerically control machining innovations made by its founder John Parsons. Interested in experiments being conducted at the MIT Servomechanisms Laboratory, Parsons proposed in 1949 that MIT become a project subcontractor to provide expertise on automatic control. Over the next 10 years, MIT gained control over the entire project as the Servomechanisms Laboratory vision of "three-axis continuous path control" supplanted the original Parsons conception of "plunge-cutting positioning." Conflict always shapes technology but this particular story, chronicled by historian David Noble, has become a significant object lesson in the history of technology."

G-Code Overview

- An introduction from linuxcnc:
 - **Overview:** <http://linuxcnc.org/docs/html/gcode/overview.html>
 - **G Codes:** <http://linuxcnc.org/docs/html/gcode/g-code.html>
 - **M Codes:** <http://linuxcnc.org/docs/html/gcode/m-code.html>
- Some key items:
 - Format of a line
 - Commands and machine modes
 - Modal groups, e.g., for motion commands
 - "Order of Execution" & "Best Practices"
 - Parameters/variables, operators, functions, ...

22. G Code Order of Execution

The order of execution of items on a line is defined not by the position of each item on the line, but by the following list:

- O-word commands (optionally followed by a comment but no other words allowed on the same line)
- Comment (including message)
- Set feed rate mode (G93, G94).
- Set feed rate (F).
- Set spindle speed (S).
- Select tool (T).
- HAL pin I/O (M62-M68).
- Change tool (M6) and Set Tool Number (M61).
- Spindle on or off (M3, M4, M5).
- Save State (M70, M73), Restore State (M72), Invalidate State (M71).
- Coolant on or off (M7, M8, M9).
- Enable or disable overrides (M48, M49,M50,M51,M52,M53).
- User-defined Commands (M100-M199).
- Dwell (G4).
- Set active plane (G17, G18, G19).
- Set length units (G20, G21).
- Cutter radius compensation on or off (G40, G41, G42)
- Cutter length compensation on or off (G43, G49)
- Coordinate system selection (G54, G55, G56, G57, G58, G59, G59.1, G59.2, G59.3).
- Set path control mode (G61, G61.1, G64)
- Set distance mode (G90, G91).
- Set retract mode (G98, G99).
- Go to reference location (G28, G30) or change coordinate system data (G10) or set axis offsets (G92, G92.1, G92.2, G94).
- Perform motion (Go to G3, G33, G38.n, G73, G76, G80 to G89), as modified (possibly) by G53.
- Stop (M0, M1, M2, M30, M60).

LinuxCNC "G-Code" Quick Reference

<http://linuxcnc.org/docs/html/gcode.html>

LinuxCNC "G-Code" Quick Reference

Code	Parameters	Description
Motion (X Y Z A B C U V W apply to all motions)		
G0		Rapid Move
G1		Linear Move
G2,G3	I J K or R, P	Arc Move
G4	P	Dwell
G5	I J P Q	Cubic Spline
G5.1	I J	Quadratic Spline
G5.2	P L	NURBS
G38.2 - G38.5		Straight Probe
G33	K	Spindle Synchronized Motion
G33.1	K	Rigid Tapping
G80		Cancel Canned Cycle
Canned cycles (X Y Z or U V W apply to canned cycles, depending on active plane)		
G81	R L (P)	Drilling Cycle
G82	R L (P)	Drilling Cycle, Dwell
G83	R L Q	Drilling Cycle, Peck
G73	R L Q	Drilling Cycle, Chip Breaking
G85	R L (P)	Boring Cycle, Feed Out
G89	R L (P)	Boring Cycle, Dwell, Feed Out
G76	P Z I J R K Q H L E	Threading Cycle
Distance Mode		
G90,G91		Distance Mode
G90.1,G91.1		Arc Distance Mode
G7		Lathe Diameter Mode
G8		Lathe Radius Mode
Feed Rate Mode		
G93,G94,G95		Feed Rate Mode
Spindle Control		
M3,M4,M5	S	Spindle Control
M19		Orient Spindle
G96,G97	S D	Spindle Control Mode
Coolant		
M7,M8,M9		Coolant Control
Tool Length Offset		
G43	H	Tool Length Offset

Tool Length Offset		
G43	H	Tool Length Offset
G43.1		Dynamic Tool Length Offset
G43.2	H	Apply additional Tool Length Offset
G49		Cancel Tool Length Compensation
Stopping		
M0,M1		Program Pause
M2,M30		Program End
M60		Pallet Change Pause
Units		
G20,G21		Units (inch, mm)
Plane Selection (affects G2, G3, G81...G89, G40...G42)		
G17 - G19.1		Plane Select
Cutter Radius Compensation		
G40		Compensation Off
G41,G42	D	Cutter Compensation
G41.1,G42.1	D L	Dynamic Cutter Compensation
Path Control Mode		
G61 G61.1		Exact Path Mode
G64	P Q	Path Blending
Return Mode in Canned Cycles		
G98,G99		Canned Cycle Return Level
Other Modal Codes		
F		Set Feed Rate
S		Set Spindle Speed
T		Select Tool)
M48,M49		Speed and Feed Override Control
M50	P0 (off) or P1 (on)	Feed Override Control
M51	P0 (off) or P1 (on)	Spindle Speed Override Control
M52	P0 (off) or P1 (on)	Adaptive Feed Control
M53	P0 (off) or P1 (on)	Feed Stop Control
G54-G59.3		Select Coordinate System

Flow-control Codes		
o sub		Subroutines, sub/endsub call
o while		Looping, while/endwhile do/while
o if		Conditional, if/else/endif
o repeat		Repeat a loop of code
[]		Indirection
o call		Call named file
M70		Save modal state
M71		Invalidate stored state
M72		Restore modal state
M73		Save and Auto-restore modal state
Input/Output Codes		
M62 - M65	P	Digital Output Control
M66	P E L Q	Wait on Input
M67	T	Analog Output,Synchronized
M68	T	Analog Output, Immediate
Non-modal Codes		
M6	T	Tool Change
M61	Q	Set Current Tool
G10 L1	P Q R	Set Tool Table
G10 L10	P	Set Tool Table
G10 L11	P	Set Tool Table
G10 L2	P R	Set Coordinate System
G10 L20	P	Set Coordinate System
G28,G28.1		Go/Set Predefined Position
G30,G30.1		Go/Set Predefined Position
G53		Move in Machine Coordinates
G92		Coordinate System Offset
G92.1,G92.2		Reset G92 Offsets
G92.3		Restore G92 Offsets
M101 - M199	P Q	User Defined Commands
Comments & Messages		
<a href;"="">;(...)		Comments
<a href;"="">(MSG,...)		Messages
<a href;"="">(DEBUG,...)		Debug Messages
<a href;"="">(PRINT,...)		Print Messages

Note: Not all available in grbl.



About Grbl

Grbl is a free, open source, high performance software for controlling the motion of machines that move, that make things, or that make things move, and will run on a straight Arduino. If the maker movement was an industry, Grbl would be the industry standard.

Most open source 3D printers have Grbl in their hearts. It has been adapted for use in hundreds of projects including laser cutters, automatic hand writers, hole drillers, graffiti painters and oddball drawing machines. Due to its performance, simplicity and frugal hardware requirements Grbl has grown into a little open source phenomenon.

<https://github.com/gnea/grbl/wiki>



Who should use Grbl

Makers who do milling or laser cutting and need a nice, simple controller for their system that will run on the ubiquitous Arduino Uno. People who loathe to clutter their space with legacy PC-towers just for the parallel-port. Tinkerers who need a controller written in tidy, modular C as a basis for their project.

Nice features

Grbl is great for light duty production. We use it for all our milling, running it from our laptops or Raspberry Pis using superb GUIs written for Grbl to stream G-code jobs. Grbl is written in highly optimized C utilizing all the clever features of the Arduino's Atmega328p chips to achieve precise timing and asynchronous operation. It is able to maintain more than **30kHz** step rate and delivers a clean, jitter free stream of control pulses.

Grbl is for three axis machines. No rotation axes (yet) – just X, Y, and Z.

The G-code interpreter implements a subset of the LinuxCNC standard and is supported by most CAM-tools with no issues. For descriptions of these G-codes, see LinuxCNC's superb documentation for their G-code descriptions, ([G-code Quick Reference](#)), and the [Shapeoko wiki](#) which attempts to list all codes supported by Grbl with appropriate commentary. Note that there are only a handful of deviations from the written G-code standard listed below. If you notice any other discrepancies, please let us know!



Supported G-Codes in v1.1

- **G0, G1:** *Linear Motions*
- **G2, G3:** *Arc and Helical Motions*
- **G4:** *Dwell*
- **G10 L2, G10 L20:** *Set Work Coordinate Offsets*
- **G17, G18, G19:** *Plane Selection*
- **G20, G21:** *Units*
- **G28, G30:** *Go to Pre-Defined Position*
- **G28.1, G30.1:** *Set Pre-Defined Position*

- **G38.2:** *Probing*
- **G38.3, G38.4, G38.5:** *Probing*
- **G40:** *Cutter Radius Compensation Modes OFF (Only)*
- **G43.1, G49:** *Dynamic Tool Length Offsets*
- **G53:** *Move in Absolute Coordinates*
- **G54, G55, G56, G57, G58, G59:** *Work Coordinate Systems*
- **G61:** *Path Control Modes*
- **G80:** *Motion Mode Cancel*
- **G90, G91:** *Distance Modes*
- **G91.1:** *Arc IJK Distance Modes*
- **G92:** *Coordinate Offset*
- **G92.1:** *Clear Coordinate System Offsets*
- **G93, G94:** *Feedrate Modes*
- **M0, M2, M30:** *Program Pause and End*
- **M3, M4, M5:** *Spindle Control*
- **M7*, M8, M9:** *Coolant Control*
- **M56* :** *Parking Motion Override Control*

LinuxCNC "G-Code" Quick Reference

<http://linuxcnc.org/docs/html/gcode.html>

LinuxCNC "G-Code" Quick Reference

Code	Parameters	Description
Motion (X Y Z A B C U V W apply to all motions)		
G0		Rapid Move
G1		Linear Move
G2, G3	I J K or R, P	Arc Move
G4	P	Dwell
G5	I J P Q	Cubic Spline
G5.1	I J	Quadratic Spline
G5.2	P L	NURBS
G38.2 - G38.5		Straight Probe
G33	K	Spindle Synchronized Motion
G33.1	K	Rigid Tapping
G80		Cancel Canned Cycle
Canned cycles (X Y Z or U V W apply to canned cycles, depending on active plane)		
G81	R L (P)	Drilling Cycle
G82	R L (P)	Drilling Cycle, Dwell
G83	R L Q	Drilling Cycle, Peck
G73	R L Q	Drilling Cycle, Chip Breaking
G85	R L (P)	Boring Cycle, Feed Out
G89	R L (P)	Boring Cycle, Dwell, Feed Out
G76	P Z I J R K Q H L E	Threading Cycle
Distance Mode		
G90, G91		Distance Mode
G90.1, G91.1		Arc Distance Mode
G7		Lathe Diameter Mode
G8		Lathe Radius Mode
Feed Rate Mode		
G93, G94, G95		Feed Rate Mode
Spindle Control		
M3, M4, M5	S	Spindle Control
M19		Orient Spindle
G96, G97	S D	Spindle Control Mode
Coolant		
M7, M8, M9		Coolant Control
Tool Length Offset		
G43	H	Tool Length Offset

Tool Length Offset		
G43	H	Tool Length Offset
G43.1		Dynamic Tool Length Offset
G43.2	H	Apply additional Tool Length Offset
G49		Cancel Tool Length Compensation
Stopping		
M0, M1		Program Pause
M2, M30		Program End
M60		Pallet Change Pause
Units		
G20, G21		Units (inch, mm)
Plane Selection (affects G2, G3, G81...G89, G40...G42)		
G17 - G19.1		Plane Select
Cutter Radius Compensation		
G40		Compensation Off
G41, G42	D	Cutter Compensation
G41.1, G42.1	D L	Dynamic Cutter Compensation
Path Control Mode		
G61 G61.1		Exact Path Mode
G64	P Q	Path Blending
Return Mode in Canned Cycles		
G98, G99		Canned Cycle Return Level
Other Modal Codes		
F		Set Feed Rate
S		Set Spindle Speed
T		Select Tool
M48, M49		Speed and Feed Override Control
M50	P0 (off) or P1 (on)	Feed Override Control
M51	P0 (off) or P1 (on)	Spindle Speed Override Control
M52	P0 (off) or P1 (on)	Adaptive Feed Control
M53	P0 (off) or P1 (on)	Feed Stop Control
G54-G59.3		Select Coordinate System

Flow-control Codes		
o sub		Subroutines, sub/endsub call
o while		Looping, while/endwhile do/while
o if		Conditional, if/else/endif
o repeat		Repeat a loop of code
[]		Indirection
o call		Call named file
M70		Save modal state
M71		Invalidate stored state
M72		Restore modal state
M73		Save and Auto-restore modal state
Input/Output Codes		
M62 - M65	P	Digital Output Control
M66	P E L Q	Wait on Input
M67	T	Analog Output,Synchronized
M68	T	Analog Output, Immediate
Non-modal Codes		
M6	T	Tool Change
M61	Q	Set Current Tool
G10 L1	P Q R	Set Tool Table
G10 L10	P	Set Tool Table
G10 L11	P	Set Tool Table
G10 L2	P R	Set Coordinate System
G10 L20	P	Set Coordinate System
G28, G28.1		Go/Set Predefined Position
G30, G30.1		Go/Set Predefined Position
G53		Move in Machine Coordinates
G92		Coordinate System Offset
G92.1, G92.2		Reset G92 Offsets
G92.3		Restore G92 Offsets
M101 - M199	P Q	User Defined Commands
Comments & Messages		
:(...)		Comments
(MSG,...)		Messages
(DEBUG,...)		Debug Messages
(PRINT,...)		Print Messages

Note: Not all available in grbl.

grbl "G-Code" Quick Reference

<http://linuxcnc.org/docs/html/gcode.html>

LinuxCNC "G-Code" Quick Reference

Code	Parameters	Description
Motion	(X Y Z A B C U V W apply to all motions)	
G0		Rapid Move
G1		Linear Move
G2, G3	I J K or R, P	Arc Move
G4	P	Dwell
G5	I J P Q	Cubic Spline
G5.1	I J	Quadratic Spline
G5.2	P L	NURBS
G38.2 - G38.5		Straight Probe
G33	K	Spindle Synchronized Motion
G33.1	K	Rigid Tapping
G80		Cancel Canned Cycle
Canned cycles	(X Y Z or U V W apply to canned cycles, depending on active plane)	
G81	R L (P)	Drilling Cycle
G82	R L (P)	Drilling Cycle, Dwell
G83	R L Q	Drilling Cycle, Peck
G73	R L Q	Drilling Cycle, Chip Breaking
G85	R L (P)	Boring Cycle, Feed Out
G89	R L (P)	Boring Cycle, Dwell, Feed Out
G76	P Z I J R K Q H L E	Threading Cycle
Distance Mode		
G90, G91		Distance Mode
G90.1, G91.1		Arc Distance Mode
G7		Lathe Diameter Mode
G8		Lathe Radius Mode
Feed Rate Mode		
G93, G94, G95		Feed Rate Mode
Spindle Control		
M3, M4, M5	S	Spindle Control
M19		Orient Spindle
G96, G97	S D	Spindle Control Mode
Coolant		
M7, M8, M9		Coolant Control

Tool Length Offset		
G43	H	Tool Length Offset
G43.1		Dynamic Tool Length Offset
G43.2	H	Apply additional Tool Length Offset
G49		Cancel Tool Length Compensation
Stopping		
M0, M1		Program Pause
M2, M30		Program End
M60		Pallet Change Pause
Units		
G20, G21		Units (inch, mm)
Plane Selection (affects G2, G3, G81...G89, G40...G42)		
G17 - G19.1		Plane Select
Cutter Radius Compensation		
G40		Compensation Off
G41, G42	D	Cutter Compensation
G41.1, G42.1	D L	Dynamic Cutter Compensation
Path Control Mode		
G61 G61.1		Exact Path Mode
G64	P O	Path Blending
Return Mode in Canned Cycles		
G98, G99		Canned Cycle Return Level
Other Modal Codes		
F		Set Feed Rate
S		Set Spindle Speed
T		Select Tool
M48, M49		Speed and Feed Override Control
M50	P0 (off) or P1 (on)	Feed Override Control
M51	P0 (off) or P1 (on)	Spindle Speed Override Control
M52	P0 (off) or P1 (on)	Adaptive Feed Control
M53	P0 (off) or P1 (on)	Feed Stop Control
G54-G59.3		Select Coordinate System

Flow-control Codes		
o sub		Subroutines, sub/endsub call
o while		Looping, while/endwhile do/while
o if		Conditional, if/else/endif
o repeat		Repeat a loop of code
[]		Indirection
o call		Call named file
M70		Save modal state
M71		Invalidate stored state
M72		Restore modal state
M73		Save and Auto-restore modal state
Input/Output Codes		
M62 - M65	P	Digital Output Control
M66	P E L Q	Wait on Input
M67	T	Analog Output, Synchronized
M68	T	Analog Output, Immediate
Non-modal Codes		
M6	T	Tool Change
M61	Q	Set Current Tool
G10 L1	P Q R	Set Tool Table
G10 L10	P	Set Tool Table
G10 L11	P	Set Tool Table
G10 L2	P R	Set Coordinate System
G10 L20	P	Set Coordinate System
G28, G28.1		Go/Set Predefined Position
G30, G30.1		Go/Set Predefined Position
G53		Move in Machine Coordinates
G92		Coordinate System Offset
G92.1, G92.2		Reset G92 Offsets
G92.3		Restore G92 Offsets
M101 - M199	P Q	User Defined Commands
Comments & Messages		
<a href;"="">:(...)		Comments
<a href;"="">(MSG,...)		Messages
<a href;"="">(DEBUG,...)		Debug Messages
<a href;"="">(PRINT,...)		Print Messages

Note: Not all available in grbl.

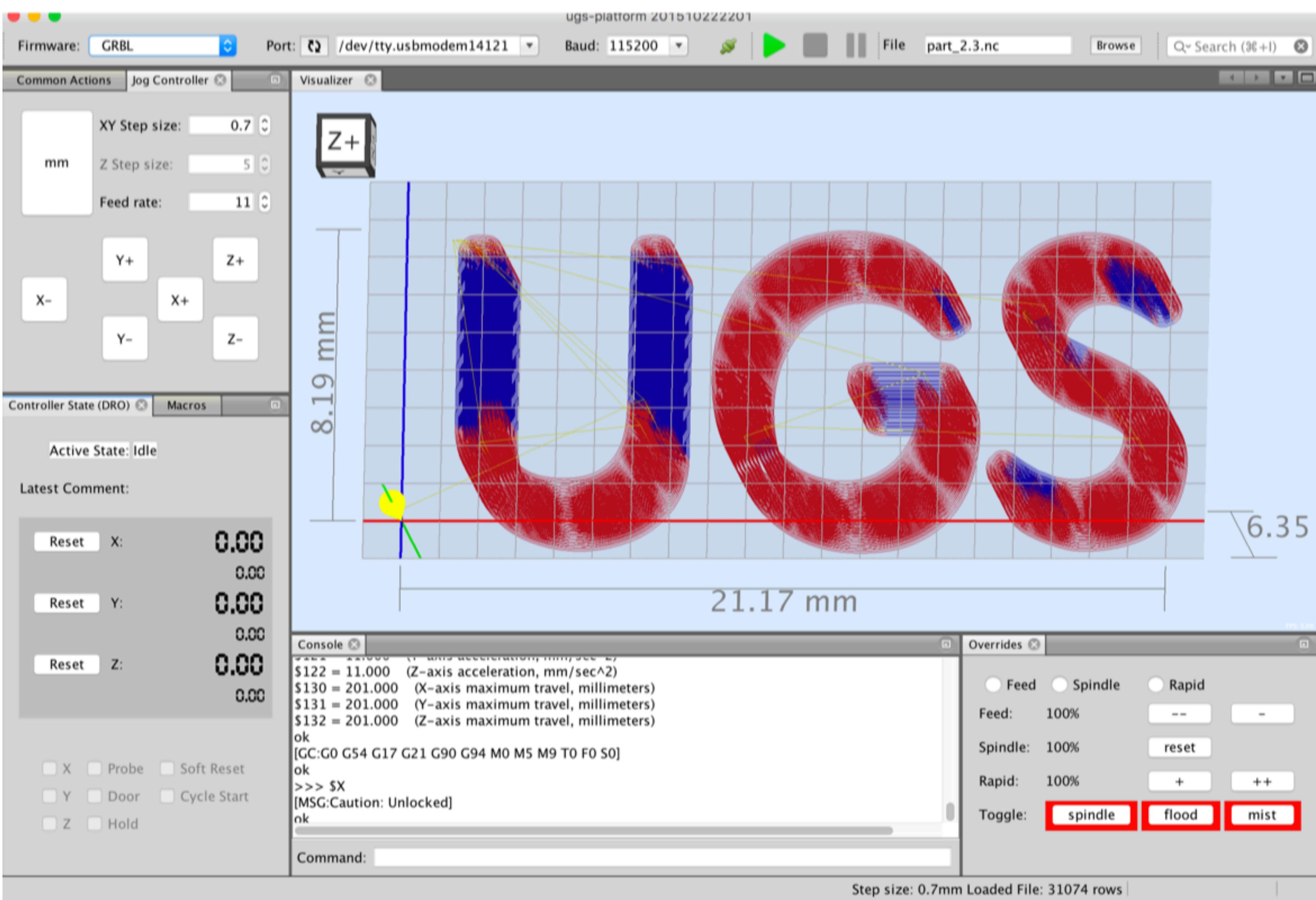


- "\$" overrides grbl gcode interpreter
- "\$\$" returns current machine parameter settings stored in EEPROM
- Can set values, e.g.,
 - $\$100 = 80.0$

```
**** Connected to /dev/tty.usbmodem14401 @ 115200 baud ****
?
Grbl 1.1g ['$' for help]
>>> $$

$0 = 10  (Step pulse time, microseconds)
$1 = 25  (Step idle delay, milliseconds)
$2 = 0   (Step pulse invert, mask)
$3 = 0   (Step direction invert, mask)
$4 = 0   (Invert step enable pin, boolean)
$5 = 0   (Invert limit pins, boolean)
$6 = 0   (Invert probe pin, boolean)
$10 = 1  (Status report options, mask)
$11 = 0.010 (Junction deviation, millimeters)
$12 = 0.002 (Arc tolerance, millimeters)
$13 = 0   (Report in inches, boolean)
$20 = 0   (Soft limits enable, boolean)
$21 = 0   (Hard limits enable, boolean)
$22 = 0   (Homing cycle enable, boolean)
$23 = 0   (Homing direction invert, mask)
$24 = 25.000 (Homing locate feed rate, mm/min)
$25 = 500.000 (Homing search seek rate, mm/min)
$26 = 250  (Homing switch debounce delay, milliseconds)
$27 = 1.000 (Homing switch pull-off distance, millimeters)
$30 = 1000 (Maximum spindle speed, RPM)
$31 = 0   (Minimum spindle speed, RPM)
$32 = 0   (Laser-mode enable, boolean)
$100 = 80.000 (X-axis travel resolution, step/mm)
$101 = 80.000 (Y-axis travel resolution, step/mm)
$102 = 250.000 (Z-axis travel resolution, step/mm)
$110 = 2500.000 (X-axis maximum rate, mm/min)
$111 = 2500.000 (Y-axis maximum rate, mm/min)
$112 = 500.000 (Z-axis maximum rate, mm/min)
$120 = 200.000 (X-axis acceleration, mm/sec^2)
$121 = 200.000 (Y-axis acceleration, mm/sec^2)
$122 = 10.000 (Z-axis acceleration, mm/sec^2)
$130 = 200.000 (X-axis maximum travel, millimeters)
$131 = 200.000 (Y-axis maximum travel, millimeters)
$132 = 200.000 (Z-axis maximum travel, millimeters)
ok
>>> $G
[GC:G0 G54 G17 G21 G90 G94 M5 M9 T0 F0 S0]
ok
```

Software Tools



universal gcode sender

webgcode

G-Code Q'n'dirty toolpath simulator

Paste your g-code in the left-hand window and see the preview of your tool path on the right.
The right-hand pane are interactive, drag them to change the point of view.

```

1 G0 Y10 Z-5
2 G1 Z-10
3 G1 Y20
4 G02 X10 Y30 R10
5 G1 X30
6 G2 X40 Y20 R10
7 G1 Y10
8 G2 X30 Y0 R10
9 G1 X10
10 G2 X0 Y10 Z-15 R10 (yeah spiral !)
11 G3 X-10 Y20 R-10 (yeah, long arc !)
12 G3 X0 Y10 I10 (center)
13 G91 G1 X10 Z10
14 G3 Y10 R5 Z3 (circle in incremental)
15 Y10 R5 Z3 (again, testing modal state)
16 G20 G0 X1 (one inch to the right)
17 G3 X-1 R1 (radius in inches)
18 G3 X1 Z0.3 I0.5 J0.5 (I,J in inches)
19 G21 (back to mm)
20 G80 X10 (do nothing)
21 G90
22 G0 X30 Y30 Z30

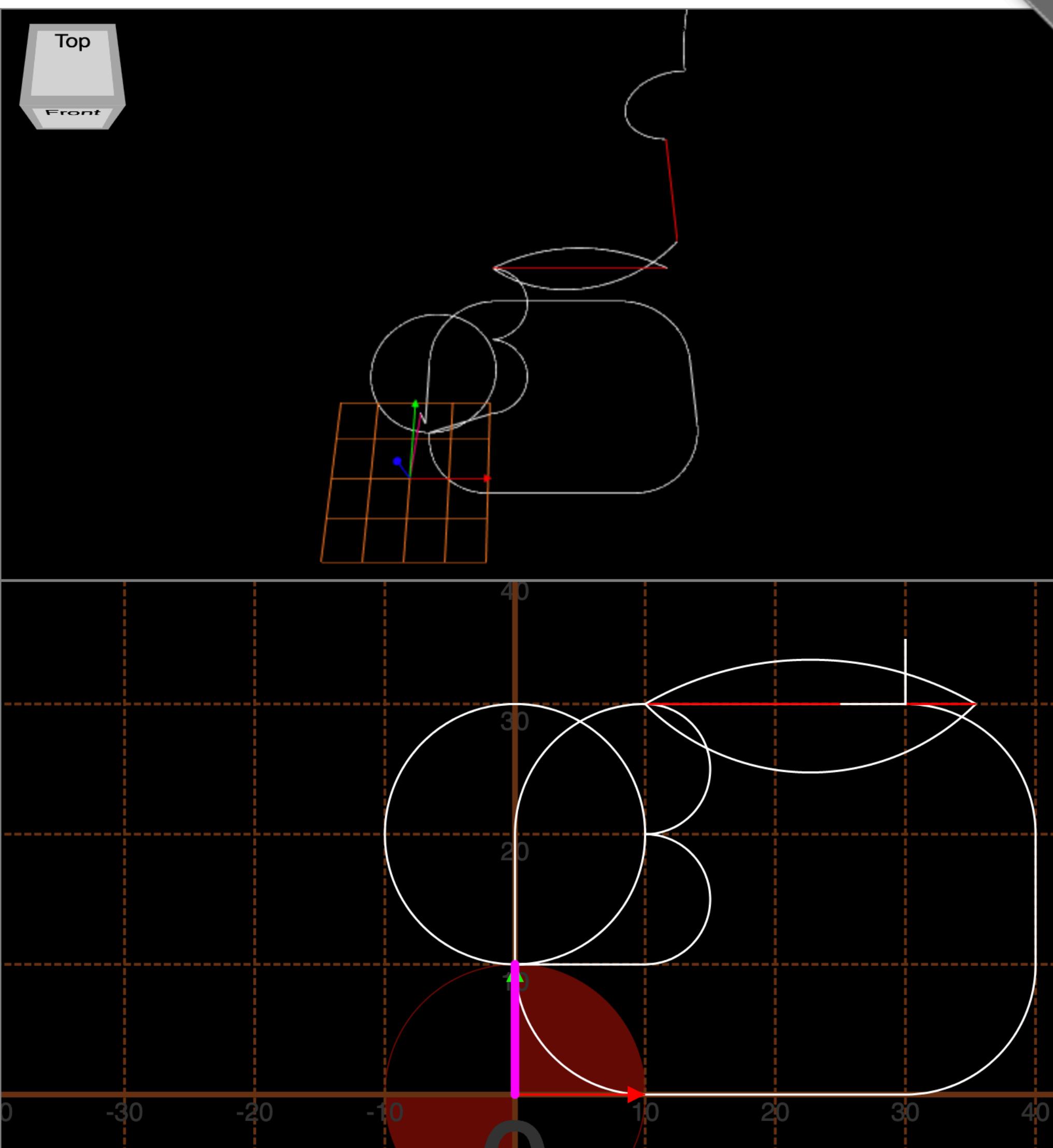
```

Total Duration:
1m39s

Bounds (@tool center):

	min	max
X	-10	40
Y	0	34.9901
Z	-15	50

- <https://nraynaud.github.io/webgcode>
- <https://github.com/nraynaud/webgcode>



gcmc: G-code Meta Compiler

```
/* Trace a path at given offset */
function trace(path, offset)
{
    goto(path[-1] + offset);
    foreach(path; v) {
        move(v + offset);
    }

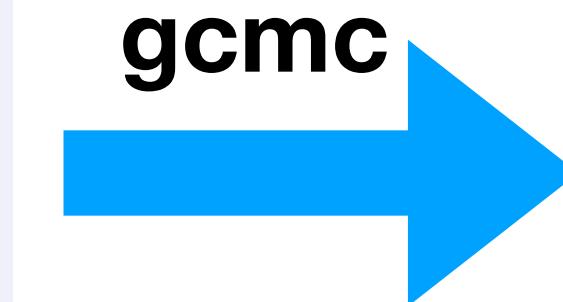
    /* Make a hole at center point with given radius */
    function hole(point, radius)
    {
        goto(point - [radius]);
        circle_cw_r([radius, 0]);
    }

    /* ----- Main Program ----- */

    HD = 6.0mm;      // Gear center-hole diameter
    N = 9;           // Number of teeth
    PA = 20.0deg;    // Pressure angle
    D = 100.0mm;     // Pitch diameter
    P = N/D;         // Diametral pitch

    // First gear
    hole([-D/2.0, 0.0mm], HD/2.0);
    trace(gear_P(N, PA, P), [D/2.0, 0.0mm]);

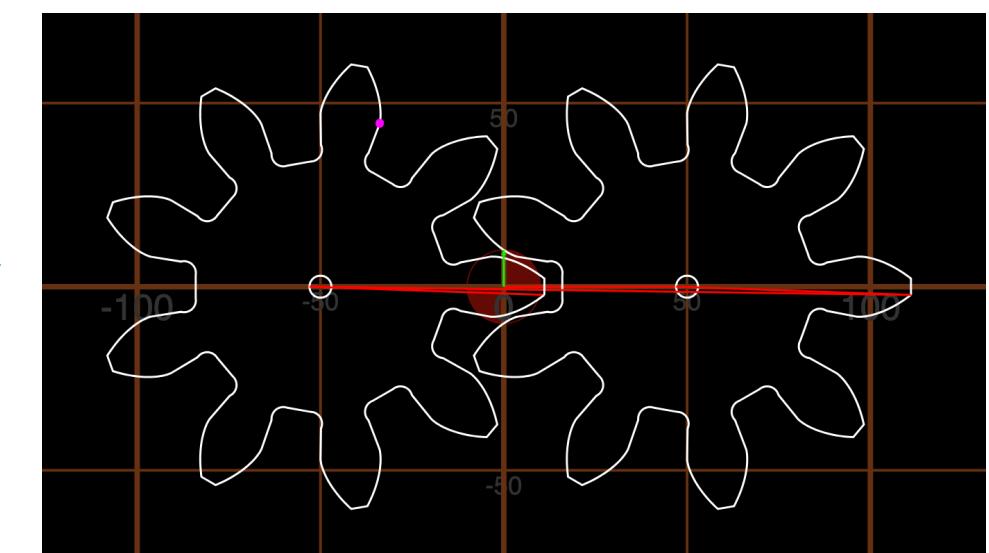
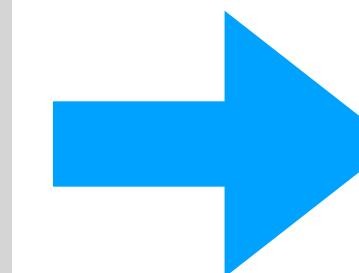
    // Second gear
    hole([-D/2.0, 0.0mm], HD/2.0);
    trace(gear_P(N, PA, P), [-D/2, 0.0mm]);
}
```



involute-gear.gcmc

```
(gcmc compiled code, do not change)
(2019-04-22 22:33:48)
(-- prologue begin --)
G17 ( Use XY plane )
G21 ( Use mm )
(G40) ( Cancel cutter radius compensation )
G49 ( Cancel tool length compensation )
G54 ( Default coordinate system )
G80 ( Cancel canned cycle )
G90 ( Use absolute distance mode )
G94 ( Units Per Minute feed rate mode )
G64 ( Enable path blending for best speed )
(-- prologue end --)
F600.00000000
G0 X47.00000000 Y0.00000000
G2 X47.00000000 Y0.00000000 I3.00000000 J0.00000000
G0 X111.07016798 Y-2.23662350
G1 X111.07016798 Y-0.00000000
G1 X111.07016798 Y2.23662350
G1 X110.18361185 Y2.90060651
G1 X109.12860272 Y3.63953515
G1 X108.09645894 Y4.31001572
...
```

involute-gear.gcode



CNC result

- <http://www.vagrearg.org/content/gcmc>
- <http://www.vagrearg.org/content/gcmc-intro>

pygcode

- <https://github.com/fragmuffin/pygcode>
- <https://github.com/fragmuffin/pygcode/wiki>

- Useful commands:



Very
Useful!

- **pygcode-norm:** Normalizes gcode from different software; intended as a preprocess for more reliable interpreting, e.g., by your CNC machine
- **pygcode-crop:** Cropping script
- Other useful features:
 - writing
 - parsing
 - interpretation/simulation via virtual machine

"This library came from my own needs to interpret and convert erroneous arcs to linear segments, and to expand canned drilling cycles, but also as a means to *learn GCode*."

<https://github.com/linuxcnc/simple-gcode-generators>

📄 README.md

Simple G-Code Generators

This repository contains a collection of Python scripts that generate simple G-Code for LinuxCNC. For me to fire up a high dollar CAD program and the use the POST processor to generate simple routines is a waste of time. So I'm writing a series of Python programs to do this. If you did an LinuxCNC install, or have Mac OS X, you already have all you need.

You can either clone this repository using Git or download the whole repository as [a zip file](#).

All of these scripts, written by various authors, are licenced under the [GNU General Public License](#).

The Scripts

- [Airfoil Generator](#) - 3-4 Axis XY-XYUV Foam EDM Style airfoil generator
- [Arc Generator](#) - generate an arc from the diameter, the start and end angle
- [Bezel Engraving](#) - engraves a bezel like you would see on the front panel of a stereo around the volume control knobs
- [Bolt Circle Array](#) - generates a circular array for canned drill cycles
- [Counterbore](#) - generates the G-code for counterbores for socket head cap screws
- [Drilling Speeds-n-Feeds](#) - helps you to calculate the speeds and feeds for drilling
- [Facing Software](#) - super simple facing Generator
- [Grid](#) - generate various shapes of grid to test the speed and the accuracy of a milling machine
- [Grill](#) - drills a circular array of holes typically used as a speaker grill or as ventilation holes in a chassis panel
- [Pocketing](#) - Rectangular-Circular Pocketing Generator
- [Text Engraving](#) - This software engraves a text string
- [Multi-line Text Engraving](#) - Engrave up to 10 lines of text
- [Ruler Engraving](#) - Engrave generic ruler in metric or standard with text

gcode?



Stepper motor music (floppy drives, etc.)

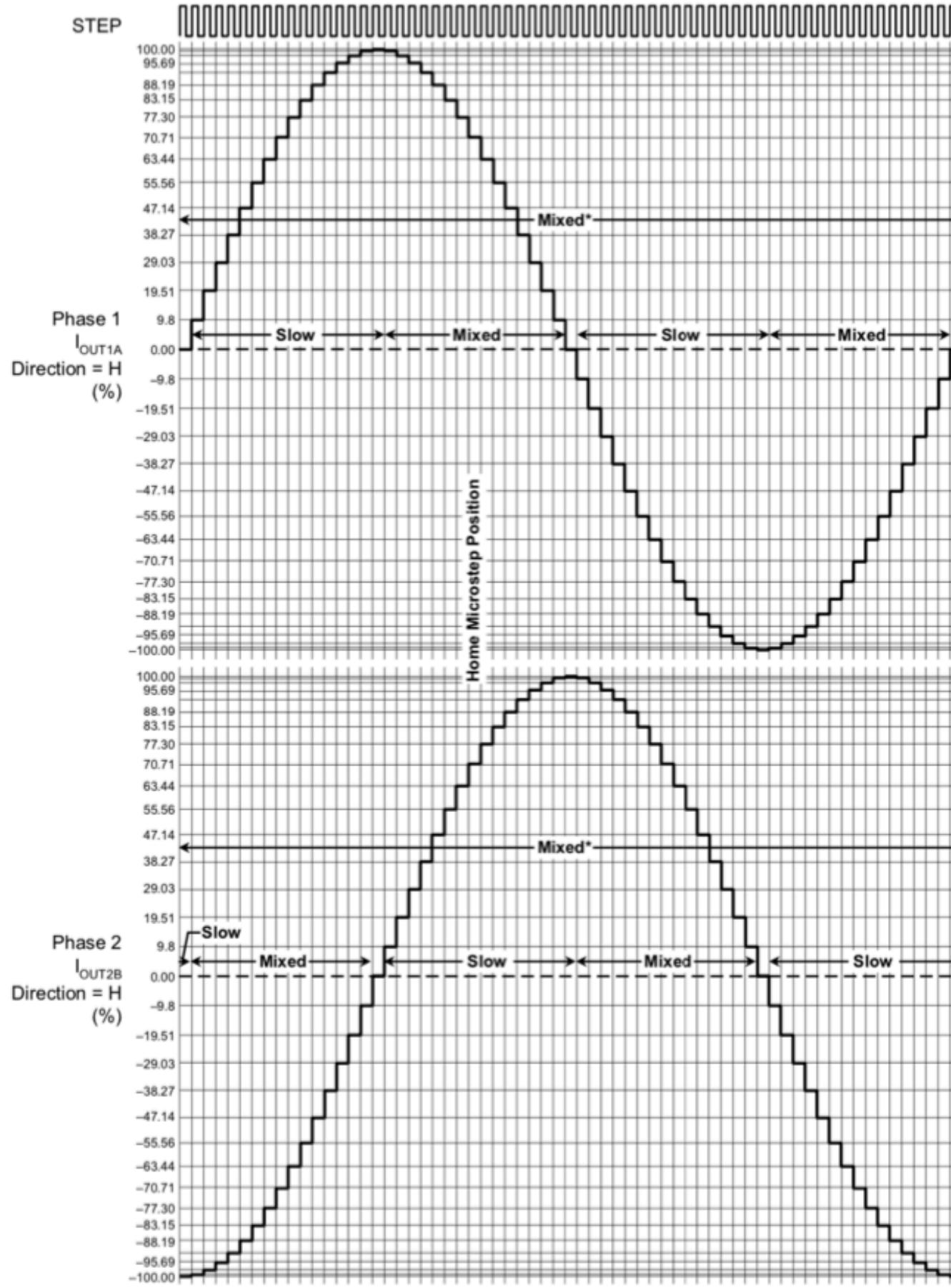


Star Wars - Imperial March on Eight Floppy Drives

1,636,213 views

https://www.youtube.com/watch?v=cM_sAxrAu7Q

Recall micro-stepping



16 Step

GRBL Settings

Specification

\$\$

Calculation of grbl settings for GT2 belts

\$100=80 // x steps per mm = 16 * 200 / 40

\$101=80 // y steps per mm

Stepping frequency

- **Steps per mm, S**
- **Feed rate, F** (in mm/min)
- **Stepping frequency** (steps/sec):

$$\begin{aligned} f &= S \text{ [steps/mm]} * F \text{ [mm/min]} * 1/60 \text{ [min/sec]} \\ &= S * F / 60 \quad [\text{in Hz}] \end{aligned}$$

- **In our case:** S=80 steps/mm, so we have

$$f = 4/3 F \text{ [in Hz]} \quad \longleftrightarrow \quad F = 3/4 f \text{ [in mm/min]}$$

- **Example:** middle C is 261.6 Hz $\Rightarrow F = 196 \text{ mm/min}$
- **Map notes to feed rates** using lookup table, $F(f)$

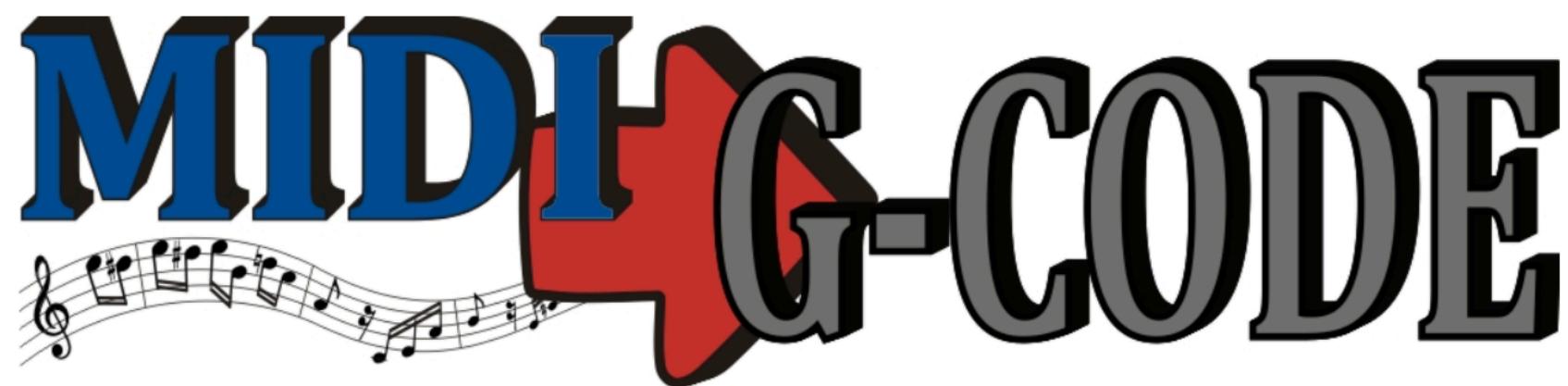
Turning music into motion

- Assume **monophonic** track for now
- To play note at frequency f for time dt , move distance F^*dt at feedrate F .
- Simulate rests using very low feedrates.



<https://www.ultimatesolver.com/en/midi2gcode>

Midi → G-Code Converter



Version	2.7.3
Description	This converter can convert music files (MIDI files) into G-code (for CNC milling machines). So with each CNC milling machine that is driven by stepping motors, it is possible to create music.
Statistics	The converter was used about 84.800 times.
MIDI file selection (*.mid)	<input type="file"/> Choose File No file chosen (max. 800 kB)
Processing	<input type="button" value="Analyze file"/>

Manually mixing gcode tracks (the painful way)

- **Problem:** Midi tracks don't always stick to stepper motors, even for monophonic tracks :/
- **Solution:** Merge mono gcode files into a polyphonic gcode file (*Yes, this is an exercise in gcode*)
- Given 2 monophonic gcode trajectories, $X(t)$ and $Y(t)$ sampled using G01 cmd
 - G01 X## F##
 - G01 Y## F##
- Determine shortest note time, $dt = \text{MIN} (dt_x = dx/F_x , dt_y = dy/F_y)$
- Play both notes as an XY trajectory with X motion of $F_x * dt$, and Y motion of $F_y * dt$
 - In incremental distance mode (G91) this would look like
 - G01 X dt_x Y dt_y Ffeedrate
 - Where the magic feedrate is $\text{Sqrt}(F_x^*F_x + F_y^*F_y)$.
 - Note: Outputs are actually all absolute distance mode (G90)

G-Code Music Demo

