Contents

[hide]

Overview

- **G0** is *rapid move*. It moves the machine to the given coordinates, with the expectation that no machining takes place during the move (tool not in contact with the stock). Unless the mode is changed by other G-code commands, the coordinates are absolute: G0 X10 moves to X=10, G0 X10 again does nothing because you're already at X=10. If coordinates for two or more axes are given, the machine moves in a straight line to the specified point. Starting from X=0, Y=0, the command G0 X10 Y10 moves diagonally to X=10, Y=10. Even if the axes have different maximum speeds and accelerations, the machine still moves them in a coordinated fashion, so that the move is linear. Only the axes specified move; the others do not change position.
- **G1** is *linear move*. It moves the machine to the given coordinates, with the expectation that the tool would be cutting. It's exactly like G0, except it has an extra parameter, F, that gives the feed rate (the speed at which to move). The speed is expressed in units per minute (mm per minute or inches per minute, depending on mode -- default mm/min). So, with the machine at X=0, Y=0, the command G1 X100 Y100 F200 moves it diagonally to X=100, Y=100, a distance of about 141 mm, at 200 mm/min, so it takes about 43 seconds to get there.
- **G2** and **G3** are *arc moves* (clockwise and counter-clockwise), and they're described very well here. I would use only the I J form and not even bother with the R form.[1]

X, Y, Z, I, J and K are coordinates. They're in millimeters, and it makes no sense to preserve precision beyond one micron. Three decimals are plenty, but if you want four, so be it, as long as your lines are less than 50/70 characters long (so as to comply w/ conucon's line length limits).

G-code is relatively easy to read. It consists of commands (which start with G, and some with M) followed by parameters. G0 moves as fast as the machine can go and is called "traverse", G1 moves at the speed given by the F parameter and is called "feed", G2 and G3 draw arcs, clockwise and counterclockwise. The expectation is that milling occurs during G1, G2 and G3, but not G0, which is why there's no speed ("feed rate") control for G0. Of the parameters, X, Y and Z are the coordinates to move to (usually absolute, but there's a relative mode too), and, for arcs, I, J and K are the coordinates of the center (usually relative, and K is for modes that are rarely used). If a coordinate does not change during a move, it can be omitted (which is why you see almost no Z parameters). With some commands (G0 to G3 included), you don't need to repeat the command on a new line if it's the same as before: you can supply just the new set of parameters (coordinates, and maybe feed rate), which is why most of your lines are just X, Y, I, J and F. Linear moves would be just X and Y (and/or Z if moving vertically too, and/or F if speed has to change).

G-code is not compiled; the Arduino runs an interpreter which receives it line by line and executes it. It doesn't move right away, then read another line. In order to avoid starting and stopping for each little move, it processes several commands ahead and feeds them into a movement planner, which schedules the moves as fast as it can while making sure that (a) it doesn't exceed the pre-programmed speed, acceleration (and/or jerk) limits of the machine, (b) it doesn't exceed the given feed rates, (c) it keeps the movements coordinated, that is, if several axes move simultaneously, they accelerate and decelerate together so that the resultant movement is a straight line (or an arc), and (d) it can stop at the end of the last command processed so far, while still obeying the same limits. It's fairly tricky to do this, and even trickier to do it on the Arduino Uno, which has only limited processing power. The Uno also generates pulses for every single microstep each motor has to move, each perfectly timed, while also doing what I described above. Be suitably impressed!

You can tell most CAM programs how many decimals to generate. Sometimes that's part of a post-processing step for the G-code. Some G-code senders can do that for you too. Failing that, just read the file and round any number after a letter X, Y, Z, I, J or K to 3-5 decimals (3 should be plenty for mm, 4 should be good for inch coordinates too).

The "AutoCAD to G-code" is what "CAM" is about. You design the part (essentially a shape) in CAD, and then design the manufacturing process for it in CAM. It's not as easy as converting, say, JPG to PNG, which any dumb computer can do. While there are programs that attempt to guess, and maybe even do a good job for simple parts, in most cases you have to tell the CAM program what to do and how to do it for each feature of your part (type of operation, choice of tool, other parameters). The CAM program helps a lot: once you've told it how you'd like it to mill a certain feature, it'll calculate what could be a very complex toolpath for you (the G-code), taking into account the geometry of the part, the geometry of the tool, and the practical limits of your manufacturing process (e.g. you can't mill more than this much material with every pass, you can't plunge more than this much at a time, you have to do a roughing pass stopping slightly short of the final shape followed by a finishing pass with a different tool, and so on). The CAM program will help you visualize the planned movement of the tool and the resulting object. It'll keep track of features and help you optimize the order of operations, and do all sorts of other things to make the process easier. But, ultimately, it's still a blend of design and engineering, with a human involved.

G-code supported

In G-code, each "action" (modal group) should be on a separate line.

conucon accepts coordinates in various forms ("0", "0." and "0.0000" are all ok), but some CNC machines require a decimal point to follow the number, which is why you'll see things like "0." in G-code. Please note that conucon is limited in how long of a line it will accept. If your job is previewing correctly, but not running properly, check to see that line lengths are w/in its limits (50 for older versions, 70 for 0.8c dev or later) as discussed here.

Motion (G)

G-code	Meaning	Notes	Support
G0/G00	Rapid positioning	Switch to rapid linear motion mode (seek). Used to get the tool somewhere quickly without cutting --- moves the machine as quickly as possible along each axis --- an axis which needs less movement will finish before the others, so one cannot count on the movement being a straight line.	
G1/G01	Linear interpolation	Switch to linear motion at the current feed rate. Used to cut a straight line --- the interpreter will determine the acceleration needed along each axis to ensure direct movement from the original to the destination point at no more than the current Feed rate (F see below).	
G2/G02	Circular interpolation, clockwise	Switch to clockwise arc mode. The interpreter will cut an arc or circle from the current position to the destination using the specified radius (R) or center (IJK location) at the current Feed rate (F see below).	
G3/G03	Circular interpolation, counterclockwise	Switch to anti-clockwise arc mode. Corollary to G02 above.	
G4/G04	Dwell (pause)	This should probably be calculated to be only one or two spindle rotations	

		for best efficiency.
G10	Set Work Coordinate Origin (and resultant Offsets)	Coordinate system origin setting. This setting is persistent and expects the user to follow good practices and not manually move the machine, instead only using jogging commands via the interface or a pendant which works through the control system, or to have and use homing switches. Example given is G10 L20 P1 X0 Y0 Z0 (which is the equivalent of G92 X0 Y0 Z0)[2] for work co-ordinate system #1 which is accessed using the command G54 (#2--#6 are G55--G59 respectively) --- see Re: Move after Homing to a position for a discussion of this.
G17	Select the XY plane (for arcs)	
G18	Select the XZ plane (for arcs)	
G19	Select the YZ plane (for arcs)	
G20	After this, units will be in inches	Best practice: do this at the start of a program and nowhere else. The usual minimum increment in G20 is one ten-thousandth of an inch (0.0001").
G21	After this, units will be in mm	Best practice: do this at the start of a program and nowhere else. The usual minimum increment in G21 (one thousandth of a millimeter, .001 mm, that is, one micrometre).
G28	Go to Pre-Defined Position	Takes an argument of an X Y Z coordinate for the intermediate point that the tool tip will pass through on its way home to machine zero.
G28.1	Set Pre-Defined Position	Takes X Y Z addresses which define the intermediate point that the tool tip will pass through on its way home to part zero, not machine zero.
G30	Go to Pre-Defined Position (Return to secondary home position)	Takes a P address specifying which machine zero point is desired, if the machine has several secondary points (P1 to P4). Takes X Y Z addresses which define the intermediate point that the tool tip will pass through on its way home to machine zero.
G30.1	Set Pre-Defined Position	
G53	Absolute mode override	Move in Machine Coordinates
G54--G59	Work Coordinate Systems	Fixture offset 1--6. CF G10 and G92.
G80	Motion mode cancel	Canned cycle
G90	Switch to absolute distance mode	Coordinates are now relative to the origin of the currently active coordinate system, as opposed to the current position. G0 X-10 Y5 will move to the position 10 units to the left and 5 above the origin X0,Y0. cf. G91 below.
G91	Switch to incremental distance mode	Coordinates are now relative to the current position, with no consideration for machine origin. G0 X-10 Y5 will move to the position 10 units to the left and 5 above the current position. cf. G90 above.
G92	Coordinate offset. Change the current coordinates without moving	e.g. "G92 x0 y0 z0" makes the current position a temporary home position. The intent is to allow one to re-use code (say for a part when doing multiples) which has a given origin. The temporary origin can then be cleared and the actual machine origin, set via G10 restored by the command below.
G92.1	Clear (temporary) Coordinate System Offsets	Previously set by G92 (see above).
G93	Set inverse time feed rate mode	An F word is interpreted to mean that the move should be completed in (one divided by the F number) minutes. For example, if F is 2, the move

		should be completed in half a minute	
G94	Set units per minute feed rate mode	An F Word is interpreted to mean the controlled point should move at a certain number of units (or degrees) per minute	

Feed Rate (F)

F-code	Meaning	Notes	Support
F	Defines feed rate	Unit used is that set by G20 or G21 (see above). The Feed setting in the G-Code determines the maximum rate at which the motor can rotate when moving a given distance so as to get up to (but not exceed) the Feed rate and the maximum acceleration limit which is set in conucon.	

Machine Options (M)

M-code	Meaning	Notes	Support
M0	Program Pause and End	Stop --- stops the machine so you can change the tool	
M1	Sleep		
M2	Program Pause and End	End	
M3	Spindle direction	Clockwise --- (re)starts the spindle	
M4	Spindle direction	Counterclockwise	
M5	Spindle Control	Stop spindle rotation --- if the system is wired up to start/stop the spindle.	
M7	Coolant Control	Mist	
M8	Coolant Control	Flood coolant on	
M9	Coolant Control	All coolant off	
M10	Vacuum	On	
M11	Vacuum	Off	
M30	Program Pause and End	End and rewind	

Tooling (T)

Tooling commands are not supported . Most don't make sense unless one has an automatic tool changer (ATC).

G-code *Not* supported

Please note that unsupported G-code may cause conucon to behave oddly, for example drifting into a corner.

G-code	Meaning	Notes
G40	Tool radius comp off	
G49	Tool offset comp cancel	
G83	Deep hole drilling canned cycle	A Perl program for post-processing a file which contains such commands is available on github.
G91.1	incremental distance mode for arcs	

conucon Specific Commands[11]

Command	Meaning	Notes
?	Current status	Displays conucon's active state and the real-time current position, both in machine coordinates and work coordinates.
$#	View gcode parameters	Display the stored offsets from machine coordinates for each co-ordinate system G54--G59, G28 and 30 and G92.
$G	View gcode parser state	Displays all of the active gcode modes that the parser will use when interpreting any incoming command.
$H	Run homing cycle	
~	Cycle start	
!	Feed hold	Brings the active cycle to a stop via a controlled deceleration, so as not to lose position. Once finished or paused, conucon will wait until a cycle start command is issued to resume to program.
Ctrl-x	Reset conucon	Resets conucon, but in a controlled way, retaining your machine position.

Using the Work Coordinate Systems

If you have limit switches on your ShapeOko it opens up the world of work coordinate systems.

There are six user-definable work coordinate systems (WCS) available that are selected with the G54-G59 commands. These are modal, meaning that once you select a WCS all following commands will reference that coordinate system until you select a different one. They are also persistent, meaning that they will be stored between resets and power cycles.

You can also reference the absolute machine coordinate system (home position) using the G53 command, but this command must be issued on every line that you want to reference machine zero on. (eg: G0 G53 X0 Y0 will send the tool to machine zero at rapid speed. The next command will reference the previously selected WCS.)

What are work coordinate systems good for? The easiest way to think about it is that it allows you to set multiple 'zero' locations on your machine which are stored in semi-permanent memory and can be recalled after homing the machine.

For instance I have a small vise that is semi-permanently mounted to the end of my work area and the G55 WCS is set so the face of the non-moving jaw of the vice is X zero and a stop I have bolted to the table is Y zero. This means that when I want to use the vise all I have to do is home the machine, select the G55 WCS, pickup Z zero and hit send on my G code file.

I also have several different jigs that I use that are fastened to the table and located by dowel pins. This allows me to swap in a different jig, select the appropriate WCS and get right to work without having to indicate in the zero location.

The first step to using work coordinate systems is to enable homing. On conucon that requires setting $16 and $17 to 1.

You probably also want to set the step idle delay ($7) to 255 so the motors are locked all the time and the machine will not drift or get bumped.

After you run the homing cycle ($H) your machine absolute zero will be set. The standard location is in the upper right corner, so that all the coordinates will be in negative space, although it doesn't really matter where you zero the machine.

The next bit is important.

From now on you should only move the machine using the jog feature or by entering G code commands. If you move the machine by hand it will not know where it is and you will lose your zero. You can't use a hand wheel to set the Z depth either.

Once you have jogged the machine to your zero location, you can set a WCS by issuing the command: G10 P[1-9] L20 X [offset] Y [offset] Z [offset]

Let me explain that a little bit.

The P[1-9] is used to select the work coordinate system to change. P1 = G54, P2 = G55, etc.

The X, Y and Z are all optional (but you need at least one otherwise there is nothing to set.)

The [offset] is optional (in that it can be '0') but is useful to compensate for the radius of the tool you are picking up your zero with.

For example, my procedure when setting a WCS is as follows:

1. Chuck up a piece of 1/4" drill rod. I have pre-measured this rod and it is .250" in diameter +/- .0005".
2. Home the machine.
3. Jog over past the edge of the work piece or jig that I want to indicate as 'zero' in the X axis.
4. Lower the Z and jog back to the edge. I sneak up on it until a piece of paper just gets captured between the rod and the edge.
5. Set the X zero for the selected WCS. The value to enter is 1/2 the diameter of the rod + the thickness of the paper (I usually use .004"). So I would enter 'G10 P1 L20 X-0.129' The value you enter is the distance from zero to where the center of the tool actually is. It can be positive or negative.
6. Repeat the procedure for the Y axis.

I do it a little differently for the Z axis. You will have to pick up the Z zero every time you change tools (unless you built a tool changer.) The procedure I use for the Z axis is as follows:

1. Chuck up the first tool I'm going to use.
2. Move over to a flat area on the work piece that is designated Z zero (although it doesn't have to be Z zero, you can input any offset you like to compensate for material thickness.)
3. Using the same 1/4" diameter drill rod, jog the Z down until the rod no longer passes under the tool.
4. Set Z zero in the work coordinates: 'G10 P1 L20 Z0.25'

Note that there are nine settings for the work coordinate systems (P1-9) but only 6 WCS selection commands in the G54-G59 range. How do you select the other three? The last three work coordinate systems are accessed using G59.1, G59.2 and G59.3. This is not currently supported in conucon and is unlikely to be added. So in conucon there are only six usable WCS.

What about G92?

G92 changes the current coordinate system to the current tool position (plus any offset you enter.) This offset remains in effect until you use G92 to change it again, or use G92.1 to remove all offsets.

G92 is useful to set a quick zero position for running parts, but I don't use it as a general rule. The main reason is that G92 is not persistent across resets and power cycles so if you have to E-Stop, or your run is interrupted for some reason, you have to re-pickup your zero.

What about G28 and G30?

G28 and G30 are not WCS settings.

G28 and G30 are persistent, stored positions that you can send the machine to with a single command. G28.1 and G30.1 are used to store the current machine position in absolute machine coordinates.

You set the G28 position by moving the machine to the position you wish to set and issuing G28.1.

Once they are set, you can issue the G28 or G30 command and the machine will move all three axis, **at rapid speed**, to the predefined position.

Note: These commands will not raise the Z axis before moving the X and Y so use caution.

You can specify an optional, intermediate position by adding X, Y or Z values to the command.

Important Note: These coordinates are in the current WCS, not absolute machine values.

So if you wanted to raise the Z first, you could say G28 Z1.5 and it would rapid the Z to 1.5 - in the current WCS - and then rapid move X, Y and Z to the saved position.

```
G00 ( G00:Rapid positioning )
G01 ( G01:Linear interpolation )
G02 ( G02:CW circular/helical interpolation )
G03 ( G03:CCW circular/helical interpolation )
G04 ( G04:Dwell )
G05 ( G05:Spline definition )
G06 ( G06:Spline interpolation )
G07 ( G07:Imaginary axis designation )
G08 ( G08:Radius mode )
G09 ( G09:Exact stop check )
G10 ( G10:Program parameter input )
G11 ( G11:Program parameter input cancel )
G12 ( G12:Circle Cutting CW )
G13 ( G13:Circle Cutting CCW )
G14 ( G13:Polar coordinate programming, absolute )
G15 ( G15:Polar coordinate programming, relative )
G16 ( G16:Definition of pole point in polar system )
G17 ( G17:X-Y plane selection )
G18 ( G18:X-Z plane selection )
G19 ( G19:Y-Z plane selection )
G20 ( G20:Inch system selection )
G21 ( G21:Milimeter system selection )
G28 ( G28:Return to home )
G30 ( G30:Return to secondard home )
G31 ( G31:Skip function  )
G32 ( G32:Thread cutting )
G33 ( G33:Constant pitch threading )
G34 ( G34:Variable pitch threading )
G40 ( G40:Tool radius comp off )
G41 ( G41:Tool radius compensation left )
G42 ( G42:Tool radius compensation right )
G43 ( G43:Tool offset compensation positive )
G44 ( G44:Tool offset compensation negative )
G45 ( G45:Tool offset compensation negative )
G46 ( G46:Axis offset single decrease )
G47 ( G47:Axis offset double increase )
G48 ( G48:Axis offset double decrease )
G49 ( G49:Tool offset comp cancel )
G50 ( G50:Scaling OFF )
G61 ( G61:Exact stop mode )
G63 ( G63:Tapping mode )
G64 ( G64:Constant velocity mode )
G65 ( G65:Custom macro simple call )
G66 ( G66:Custom macro modal call )
G67 ( G67:Custom macro modal call cancel )
G68 ( G68:Coordinate system rotation ON )
G69 ( G69:Coordinate system rotation OFF )
G70 ( G70:Enter inch mode )
G73 ( G73:High speed drilling canned cycle )
G74 ( G74:Left hand tapping canned cycle )
G76 ( G76:Fine boring canned cycle )
```

```
G80 ( G80:Cancel canned cycle )
G81 ( G81:Drilling to final depth canned cycle )
G82 ( G82:Drilling to final depth canned cycle )
G83 ( G83:Deep hole drilling canned cycle )
G84 ( G84:Tapping or thread cutting with balanced chuck canned cycle )
G85 ( G85:Reaming canned cycle )
G86 ( G86:boring canned cycle )
G87 ( G87:Reaming with measuring stop canned cycle )
G88 ( G88:Boring with spindle stop canned cycle )
G89 ( G89:Boring with intermediate stop canned cycle )
G90 ( G90:Absolute prog )
G91 ( G91:Incremental programming )
G92 ( G92:Reposition current point - can be used to zero machine )
G94 ( G94:Inch per minute )
G95 ( G95:Per revolution feed )
G98 ( G98:Set Initial Plane default )

M01 ( M01:Optional Stop )
M02 ( M02:End of program..no rewind )
M03 ( M03:Spindle On )
M04 ( M04:Spindle CCW )
M05 ( M05:Spindle Stop )
M06 ( M06:Tool change )
M07 ( M07: Coolant On )
M08 ( M08:Flood coolant on )
M09 ( M09:Mist Coolant Device Off )
M10 ( M10:(Mach) Digital Pin Off )
M11 ( M10:(Mach) Digital Pin On )
M20 ( M20:(RepRap) List SD Card )
M21 ( M21:(RepRap) Init SD Card )
M22 ( M20:(RepRap) Release SD Card )
M23 ( M22:(RepRap) Select SD File )
M24 ( M24:(RepRap) Start/Resume SD Print )
M25 ( M25:(RepRap) Pause SD Print )
M26 ( M26:enable automatic b-axis clamping  )
M27 ( M27:disable automatic b-axis clamping )
M27 ( M27:disable automatic b-axis clamping )
M28 ( M28:(RepRap) Start SD Write )
M29 ( M29:(RepRap) Stop SD Write )
M30 ( M30:End program...rewind stop )
M42 ( M42:(RepRap) Set output pin )
M47 ( M47:Repeat program from first line )
M48 ( M48:enable speed and feed overrides )
M49 ( M49:Disable speed and feed overrides )
M50 ( M50:(EMC2) Feed Control Override )
M51 ( M51:(EMC2) Spindle Speed Override Control )
M52 ( M52:(EMC2) Adaptive Feed Control )
M53 ( M53:(EMC2) Feed stop control )
M60 ( M60:pallet shuttle and program stop )
M61 ( M61:(EMC2) Set current tool number )
M62 ( M62:(EMC2) turn on digital output synched with motion )
M63 ( M63:(EMC2) Turn off digital output synched with motion )
M64 ( M64:(EMC2) Turn on digital output immediately )
M65 ( M65:(EMC2) Turn off digital output immediately )
M66 ( M66:(EMC2) Input control )
M80 ( M80:(RepRap) Turn on P/S )
M81 ( M81:(RepRap) Turn off P/S )
M82 ( M82:(RepRap) Set E codes Absolute (default) )
M83 ( M83:(RepRap) Set E codes relative while in Absolute Coordinates (G90) mode )
M84 ( M84:(RepRap) Disable steppers until next move )
M85 ( M85:(RepRap) Set inactivity shutdown timer )
M95 ( M95:?? )
M98 ( M98:Call subroutine )
M99 ( M99:Return from subroutine )
M104 ( M104: (RepRap) Set Extruder Target Temp )
M105 ( M105:(RepRap) Read current Temp )
M106 ( M106:(RepRap) Fan On )
M107 ( M107:(RepRap) Fan off )
M109 ( M109:(RepRap) Wait for extruder current temp to reach target temp. )
M114 ( M114:(RepRap) Display current position )
M115 ( M115:(RepRap) Capabilities string )
```

```
M140 ( M140:(RepRap) Set bed target temp )
M190 ( M190:(RepRap) Wait for bed current temp to reach target temp )
M201 ( M201:(RepRap)  Set max acceleration in units/s^2 for print moves (M201 X1000 Y1000) )
M202 ( M202:(RepRap)  Set max acceleration in units/s^2 for travel moves (M202 X1000 Y1000) )
F ( F:Feedrate: )
H ( H:Tool length offset index: )
I ( I:X axis offset for arcs: )
J ( J:Y axis offset for arcs: )
K ( K:Z axis offset for arcs: )
O ( O:Subroutine label number: )
P ( P:Line Number: )
Q ( Q:Repititions of subroutine call: )
R ( R:Arc radius: )
S ( S:Spindle Speed: )
T ( T:Tool Number: )
# ( #: Variable Assignment:# )
% ( %: Start or end of program )
```