

# The gcodepreview OpenSCAD library\*

Author: William F. Adams  
willadams at aol dot com

2024/04/12

## Abstract

The gcodepreview library allows using OpenSCAD to move a tool in lines and output dxf and G-code files so as to work as a CAD/CAM program for CNC.

## Contents

<b>1</b>	<b>readme.md</b>	<b>1</b>
<b>2</b>	<b>gcodepreview</b>	<b>3</b>
2.1	Position and Variables . . . . .	4
2.2	Tools and Changes . . . . .	8
2.3	File Handling . . . . .	10
2.4	Movement and Cutting . . . . .	20
<b>3</b>	<b>gcodepreview_template.scad</b>	<b>22</b>

## 1 readme.md

```
1 <rdm> # gcodepreview
2 <rdm>
3 <rdm> OpenSCAD library for moving a tool in lines and arcs so as to model how a part would be cut
4 <rdm> using G-Code, so as to allow OpenSCAD to function as a compleat CAD/CAM solution for
5 <rdm> subtractive CNC (mills and routers).
6 <rdm>
7 <rdm> ![OpenSCAD Cut Joinery Module](https://raw.githubusercontent.com/WillAdams/gcodepreview/main
8 <rdm>
9 <rdm> Updated to make use of Python in OpenSCAD:
10 <rdm>
11 <rdm> http://www.guenther-sohler.net/openscad/
12 <rdm>
13 <rdm> (previous versions had used RapCAD)
14 <rdm>
15 <rdm> A BlockSCAD file for the main modules is available at:
16 <rdm>
```

---

\*This file (gcodepreview.dtx) has version number v0.2, last revised 2024/04/12.

```

17 <rdm> https://www.blockscad3d.com/community/projects/1244473
18 <rdm>
19 <rdm> The project is discussed at:
20 <rdm>
21 <rdm> https://forum.makerforums.info/t/g-code-preview-using-openscad-rapcad/85729
22 <rdm>
23 <rdm> and
24 <rdm>
25 <rdm> https://forum.makerforums.info/t/openscad-and-python-looking-to-finally-be-resolved/85729
26 <rdm>
27 <rdm> and
28 <rdm>
29 <rdm> https://willadams.gitbook.io/design-into-3d/programming
30 <rdm>
31 <rdm> Usage is:
32 <rdm>
33 <rdm> Place the file in C:\Users\\~\Documents\OpenSCAD\libraries (C:\Users\\~\Documents\OpenSCAD\libraries)
34 <rdm> deprecated since RapCAD is not longer needed since Python is now used for writing
35 <rdm>
36 <rdm> (While it was updated for use w/ RapCAD, so as to take advantage of the write command
37 <rdm> it was possible to write that in Python)
38 <rdm>
39 <rdm>     use <gcodepreview.py>;
40 <rdm>     use <pygcodepreview.scad>;
41 <rdm>     include <gcodepreview.scad>;
42 <rdm>
43 <rdm> Note that it is necessary to use the first two files (this allows loading
44 <rdm> the Python commands and then wrapping them in OpenSCAD commands) and then
45 <rdm> include the last file (which allows using OpenSCAD variables to selectively
46 <rdm> implement the Python commands via their being wrapped in OpenSCAD modules)
47 <rdm>
48 <rdm> and define variables which match the project and then use commands such as:
49 <rdm>
50 <rdm>    .opengcodefile(Gcode_filename);
51 <rdm>    .opendxf(DXF_filename);
52 <rdm>
53 <rdm>     difference() {
54 <rdm>         setupstock(stocklength, stockwidth, stockthickness, zeroheight, stockorigin);
55 <rdm>
56 <rdm>         movetosafez();
57 <rdm>
58 <rdm>         toolchange(squaretoolno, speed * square_ratio);
59 <rdm>
60 <rdm>         begintoolpath(0,0,0.25);
61 <rdm>         beginpolyline(0,0,0.25);
62 <rdm>
63 <rdm>         cutoneaxis_setfeed("Z",-1,plunge*square_ratio);
64 <rdm>         addpolyline(stocklength/2,stockwidth/2,-stockthickness);
65 <rdm>
66 <rdm>         cutwithfeed(stocklength/2,stockwidth/2,-stockthickness,feed);
67 <rdm>
68 <rdm>         endtoolpath();
69 <rdm>         endpolyline();
70 <rdm>

```

```

71 <rdm>     }
72 <rdm>
73 <rdm>     closegcodefile();
74 <rdm>     closedxfile();
75 <rdm>
76 <rdm> Tool numbers match those of tooling sold by Carbide 3D (ob. discl., I work for them).
77 <rdm> Comments are included in the G-code to match those expected by CutViewer.
78 <rdm>
79 <rdm> A complete example file is: gcodepreview_template.scad another is
80 <rdm> openscad_gcodepreview_cutjoinery.tres.scad which is made from an
81 <rdm> OpenSCAD Graph Editor file:
82 <rdm>
83 <rdm> ![OpenSCAD Graph Editor Cut Joinery File](https://raw.githubusercontent.com/WillAdams/gcodepreview/master/gcodepreview_cutjoinery.tres.scad)
84 <rdm>
85 <rdm> Version 0.1 supports setting up stock, origin, rapid positioning, making cuts,
86 <rdm> and writing out matching G-code, and creating a DXF with polylines.
87 <rdm>
88 <rdm> Added features since initial upload:
89 <rdm>
90 <rdm> - endpolyline(); --- this command allows ending one polyline so as to allow multiple lines
91 <rdm> - separate dxf files are written out for each tool where tool is ball/square/V and small/large
92 <rdm> - re-writing as a Literate Program using the LaTeX package docmfp (begun 4/12/24)
93 <rdm>
94 <rdm> Not quite working feature:
95 <rdm>
96 <rdm> - exporting SVGs --- these are written out upside down due to coordinate differences between OpenSCAD and CutViewer
97 <rdm>
98 <rdm> Possible future improvements:
99 <rdm>
100 <rdm> - G-code: support for G2/G3 arcs
101 <rdm> - DXF support for curves and the 3rd dimension
102 <rdm> - G-code: import external tool libraries and feeds and speeds from JSON or CSV files --- not implemented yet
103 <rdm> - support for additional tooling shapes such as dovetail tools, or roundover tooling
104 <rdm> - general coding improvements --- current coding style is quite prosaic
105 <rdm> - generalized modules for cutting out various shapes/geometries --- a current one is to cut a circle
106 %

```

## 2 gcodepreview

As noted above, this library works by using Python code as a back-end so as to persistently store and access variables, and to write out files. Doing so requires a total of three files:

- A Python file: gcodepreview.py (gcpy)
- An OpenSCAD file: gcodepreview.scad (gcpscad)
- An OpenSCAD file which connects the other two files: pygcodepreview.scad (pyscad)

Each file will begin with a suitable comment indicating the file type:

```

107 <gcpy>#!/usr/bin/env python
108 <gcpy>

```

```

109 <pyscad> #!/OpenSCAD
110 <pyscad>
111 <gcpscad> #!/OpenSCAD
112 <gcpscad>
113 <gcpscad> //gcodepreview 0.1
114 <gcpscad> //
115 <gcpscad> //used via use <gcodepreview.py>;
116 <gcpscad> //          use <pygcodepreview.scad>;
117 <gcpscad> //          include <gcodepreview.scad>;
118 <gcpscad> //

```

`writeln` The original implementation in RapSCAD used a command `writeln` — fortunately, this command is easily re-created in Python:

```

119 <gcpy>def writeln(*arguments):
120 <gcpy>     line_to_write = ""
121 <gcpy>     for element in arguments:
122 <gcpy>         line_to_write += element
123 <gcpy>     f.write(line_to_write)
124 <gcpy>     f.write("\n")

```

which command will accept a series of arguments and then write them out to a file object.

## 2.1 Position and Variables

In modeling the machine motion and G-code it will be necessary to have the machine track several variables. This will be done using paired functions (which will return the matching variable) and a matching (global) variable, as well as additional functions for setting the matching variable.

The first such variables are for XYZ position:

```

mpx      • mpx
mpy      • mpy
mpz      • mpz

```

It will further be necessary to have a variable for the current tool:

```

currenttool • currenttool

```

For each command it will be necessary to implement an appropriate aspect in each file. The Python file will manage the Python variables and handle things which can only be done in Python, while there will be two OpenSCAD files as noted above, one which calls the Python code (this will be `used`), while the other will be `included` and will be able to access and use OpenSCAD variables, as well as implement Customizer options.

The first such routine will be appropriately enough, to set up the stock, and `psetupstock` perform other initializations.

```

125 <gcpy>def psetupstock(stocklength, stockwidth, stockthickness, zeroheight, stockorigin)
126 <gcpy>     global mpx
127 <gcpy>     mpx = float(0)
128 <gcpy>     global mpy

```

```

129 <gcpy>     mpy = float(0)
130 <gcpy>     global mpz
131 <gcpy>     mpz = float(0)
132 <gcpy>     global currenttool
133 <gcpy>     currenttool = 102
134 <gcpy>

osetupstock

135 <pyscad> module osetupstock(stocklength, stockwidth, stockthickness, zeroheight, stockorigin) {
136 <pyscad>     psetupstock(stocklength, stockwidth, stockthickness, zeroheight, stockorigin);
137 <pyscad> }
138 <pyscad>

setupstock

139 <gcpscad> module setupstock(stocklength, stockwidth, stockthickness, zeroheight, stockorigin) {
140 <gcpscad>     osetupstock(stocklength, stockwidth, stockthickness, zeroheight, stockorigin);
141 <gcpscad> //initialize default tool and XYZ origin
142 <gcpscad>     osettool(102);
143 <gcpscad>     oset(0,0,0);
144 <gcpscad>     if (zeroheight == "Top") {
145 <gcpscad>         if (stockorigin == "Lower-Left") {
146 <gcpscad>             translate([0, 0, (-stockthickness)]){
147 <gcpscad>                 cube([stocklength, stockwidth, stockthickness], center=false);
148 <gcpscad> if (generategcode == true) {
149 <gcpscad> // owriteone("setupstock");
150 <gcpscad> owritethree("(stockMin:0.00mm, 0.00mm, -,str(stockthickness),"mm");
151 <gcpscad> owritefive("(stockMax:",str(stocklength),"mm, ",str(stockwidth),"mm, 0.00mm");
152 <gcpscad>     owritenine("(STOCK/BLOCK, ",str(stocklength)," ",str(stockwidth)," ",str(stockthickn
153 <gcpscad> }
154 <gcpscad> }
155 <gcpscad> }
156 <gcpscad>     else if (stockorigin == "Center-Left") {
157 <gcpscad>         translate([0, (-stockwidth / 2), -stockthickness]){
158 <gcpscad>             cube([stocklength, stockwidth, stockthickness], center=false);
159 <gcpscad>         if (generategcode == true) {
160 <gcpscad> // owriteone("setupstock");
161 <gcpscad> owritefive("(stockMin:0.00mm, -,str(stockwidth/2),"mm, -,str(stockthickness),"mm");
162 <gcpscad> owritefive("(stockMax:",str(stocklength),"mm, ",str(stockwidth/2),"mm, 0.00mm");
163 <gcpscad>     owriteeleven("(STOCK/BLOCK, ",str(stocklength)," ",str(stockwidth)," ",str(stockthickn
164 <gcpscad> }
165 <gcpscad> }
166 <gcpscad> } else if (stockorigin == "Top-Left") {
167 <gcpscad>     translate([0, (-stockwidth), -stockthickness]){
168 <gcpscad>         cube([stocklength, stockwidth, stockthickness], center=false);
169 <gcpscad> if (generategcode == true) {
170 <gcpscad> // owriteone("setupstock");
171 <gcpscad> owritefive("(stockMin:0.00mm, -,str(stockwidth),"mm, -,str(stockthickness),"mm");
172 <gcpscad> owritethree("(stockMax:",str(stocklength),"mm, 0.00mm, 0.00mm");
173 <gcpscad> owriteeleven("(STOCK/BLOCK, ",str(stocklength)," ",str(stockwidth)," ",str(stockthickn
174 <gcpscad> }
175 <gcpscad> }
176 <gcpscad> }
177 <gcpscad> else if (stockorigin == "Center") {
178 <gcpscad> //owritecomment("Center");
179 <gcpscad>     translate([(stocklength / 2), (-stockwidth / 2), -stockthickness]){

```

```

180 (gcpscad)      cube([stocklength, stockwidth, stockthickness], center=false);
181 (gcpscad) if (generategcode == true) {
182 (gcpscad) // owriteone("setupstock");
183 (gcpscad) owriteseven("(stockMin: -,str(stocklength/2),", "-",str(stockwidth/2),"mm, -"
184 (gcpscad) owritefive("(stockMax:",str(stocklength/2),"mm, ",str(stockwidth/2),"mm, 0.00mm)");
185 (gcpscad) owritethirteen("(STOCK/BLOCK, ",str(stocklength)," ",str(stockwidth)," ",str(stockthickness))");
186 (gcpscad) }
187 (gcpscad) }
188 (gcpscad) }
189 (gcpscad) } else if (zeroheight == "Bottom") {
190 (gcpscad) //owritecomment("Bottom");
191 (gcpscad)      if (stockorigin == "Lower-Left") {
192 (gcpscad)      cube([stocklength, stockwidth, stockthickness], center=false);
193 (gcpscad) if (generategcode == true) {
194 (gcpscad) // owriteone("setupstock");
195 (gcpscad) owriteone("(stockMin:0.00mm, 0.00mm, 0.00mm)");
196 (gcpscad) owriteseven("(stockMax:",str(stocklength),"mm, ",str(stockwidth),"mm, ",str(stockthickness))");
197 (gcpscad) owriteseven("(STOCK/BLOCK, ",str(stocklength)," ",str(stockwidth)," ",str(stockthickness))");
198 (gcpscad) }
199 (gcpscad) } else if (stockorigin == "Center-Left") {
200 (gcpscad)      translate([0, (-stockwidth / 2), 0]){
201 (gcpscad)      cube([stocklength, stockwidth, stockthickness], center=false);
202 (gcpscad) if (generategcode == true) {
203 (gcpscad) // owriteone("setupstock");
204 (gcpscad) owritethree("(stockMin:0.00mm, -,str(stockwidth/2),"mm, 0.00mm)");
205 (gcpscad) owriteeven("(stockMax:",str(stocklength),"mm, ",str(stockwidth/2),"mm, ",str(stockthickness))");
206 (gcpscad) owritenine("(STOCK/BLOCK, ",str(stocklength)," ",str(stockwidth)," ",str(stockthickness))");
207 (gcpscad) }
208 (gcpscad) }
209 (gcpscad) } else if (stockorigin == "Top-Left") {
210 (gcpscad)      translate([0, (-stockwidth), 0]){
211 (gcpscad)      cube([stocklength, stockwidth, stockthickness], center=false);
212 (gcpscad) }
213 (gcpscad) if (generategcode == true) {
214 (gcpscad) // owriteone("setupstock");
215 (gcpscad) owritethree("(stockMin:0.00mm, -,str(stockwidth),"mm, 0.00mm)");
216 (gcpscad) owritefive("(stockMax:",str(stocklength),"mm, 0.00mm, ",str(stockthickness))");
217 (gcpscad) owritenine("(STOCK/BLOCK, ",str(stocklength)," ",str(stockwidth)," ",str(stockthickness))");
218 (gcpscad) }
219 (gcpscad) } else if (stockorigin == "Center") {
220 (gcpscad)      translate([(stocklength / 2), (stockwidth / 2), 0]){
221 (gcpscad)      cube([stocklength, stockwidth, stockthickness], center=false);
222 (gcpscad) }
223 (gcpscad) if (generategcode == true) {
224 (gcpscad) // owriteone("setupstock");
225 (gcpscad) owritefive("(stockMin:-,str(stocklength/2),", "-",str(stockwidth/2),"mm, 0.00mm)");
226 (gcpscad) owriteeven("(stockMax:",str(stocklength/2),"mm, ",str(stockwidth/2),"mm, ",str(stockthickness))");
227 (gcpscad) owriteeleven("(STOCK/BLOCK, ",str(stocklength)," ",str(stockwidth)," ",str(stockthickness))");
228 (gcpscad) }
229 (gcpscad) }
230 (gcpscad) }
231 (gcpscad) if (generategcode == true) {
232 (gcpscad) owriteone("G90");
233 (gcpscad) owriteone("G21");

```

```

234 <gcpscad> // owriteone("Move to safe Z to avoid workholding");
235 <gcpscad> // owriteone("G53G0Z-5.000");
236 <gcpscad> }
237 <gcpscad> //owritecomment("ENDSETUP");
238 <gcpscad> }
239 <gcpscad>

```

**xpos** It will be necessary to have Python functions which return the current values  
**ypos** of the machine position in Cartesian coordinates:

```

zpos
240 <gcpy>def xpos():
241 <gcpy>     global mpx
242 <gcpy>     return mpx
243 <gcpy>
244 <gcpy>def ypos():
245 <gcpy>     global mpy
246 <gcpy>     return mpy
247 <gcpy>
248 <gcpy>def zpos():
249 <gcpy>     global mpz
250 <gcpy>     return mpz
251 <gcpy>

```

**psetxpos** and in turn, functions which set the positions:

```

psetypos
psetzpos
252 <gcpy>def psetxpos(newxpos):
253 <gcpy>     global mpx
254 <gcpy>     mpx = newxpos
255 <gcpy>
256 <gcpy>def psetypos(newypos):
257 <gcpy>     global mpy
258 <gcpy>     mpy = newypos
259 <gcpy>
260 <gcpy>def psetzpos(newzpos):
261 <gcpy>     global mpz
262 <gcpy>     mpz = newzpos
263 <gcpy>

```

**getxpos** and as noted above, there will need to be matching OpenSCAD versions:

```

getypos
getzpos
setxpos
setypos
setzpos
264 <pyscad> function getxpos() = xpos();
265 <pyscad> function getypos() = ypos();
266 <pyscad> function getzpos() = zpos();
267 <pyscad>
268 <pyscad> module setxpos(newxpos) {
269 <pyscad>     psetxpos(newxpos);
270 <pyscad> }
271 <pyscad>
272 <pyscad> module setypos(newypos) {
273 <pyscad>     psetypos(newypos);
274 <pyscad> }
275 <pyscad>
276 <pyscad> module setzpos(newzpos) {
277 <pyscad>     psetzpos(newzpos);
278 <pyscad> }
279 <pyscad>

```

**oset**

```

280 (gcpcscad) module oset(ex, ey, ez) {
281 (gcpcscad) setxpos(ex);
282 (gcpcscad) setypos(ey);
283 (gcpcscad) setzpos(ez);
284 (gcpcscad) }
285 (gcpcscad)

```

## 2.2 Tools and Changes

Similarly Python functions and variables will be used to track and set and return

`psettool` the current tool:

```

pcurrenttool 286 (gcpy)def psettool(tn):
287 (gcpy)    global currenttool
288 (gcpy)    currenttool = tn
289 (gcpy)
290 (gcpy)def pcurrent_tool():
291 (gcpy)    global currenttool
292 (gcpy)    return currenttool
293 (gcpy)

```

`osettool` and matching OpenSCAD modules set and return the current tool:

```

currenttool 294 (pyscad) module osettool(tn){
295 (pyscad) psettool(tn);}
296 (pyscad)
297 (pyscad) function current_tool() = pcurrent_tool();
298 (pyscad)

```

`toolchange` and apply the appropriate commands for a `toolchange`.

```

299 (gcpcscad) module toolchange(tool_number,speed) {
300 (gcpcscad)    osettool(tool_number);
301 (gcpcscad) if (generategcode == true) {
302 (gcpcscad)    writecomment("Toolpath");
303 (gcpcscad)    owriteone("M05");
304 (gcpcscad) // writecomment("Move to safe Z to avoid workholding");
305 (gcpcscad) // owriteone("G53G0Z-5.000");
306 (gcpcscad) //    writecomment("Begin toolpath");
307 (gcpcscad)    if (tool_number == 201) {
308 (gcpcscad)    writecomment("TOOL/MILL,6.35, 0.00, 0.00, 0.00");
309 (gcpcscad)    } else if (tool_number == 202) {
310 (gcpcscad)    writecomment("TOOL/MILL,6.35, 3.17, 0.00, 0.00");
311 (gcpcscad)    } else if (tool_number == 102) {
312 (gcpcscad)    writecomment("TOOL/MILL,3.17, 0.00, 0.00, 0.00");
313 (gcpcscad)    } else if (tool_number == 101) {
314 (gcpcscad)    writecomment("TOOL/MILL,3.17, 1.58, 0.00, 0.00");
315 (gcpcscad)    } else if (tool_number == 301) {
316 (gcpcscad)    writecomment("TOOL/MILL,0.03, 0.00, 6.35, 45.00");
317 (gcpcscad)    } else if (tool_number == 302) {
318 (gcpcscad)    writecommment("TOOL/MILL,0.03, 0.00, 10.998, 30.00");
319 (gcpcscad)    } else if (tool_number == 390) {
320 (gcpcscad)    writecomment("TOOL/MILL,0.03, 0.00, 1.5875, 45.00");
321 (gcpcscad)    }
322 (gcpcscad)    select_tool(tool_number);
323 (gcpcscad)    owritetwo("M6T",str(tool_number));
324 (gcpcscad)    owritetwo("M03S",str(speed));

```



```

325 <gcpscad> }
326 <gcpscad> }
327 <gcpscad>

```

There must also be a module for selecting tools: `select_tool`:

```

selecttool
tool number
328 <gcpscad> module select_tool(tool_number) {
329 <gcpscad> //echo(tool_number);
330 <gcpscad>   if (tool_number == 201) {
331 <gcpscad>     gcp_endmill_square(6.35, 19.05);
332 <gcpscad>   } else if (tool_number == 202) {
333 <gcpscad>     gcp_endmill_ball(6.35, 19.05);
334 <gcpscad>   } else if (tool_number == 102) {
335 <gcpscad>     gcp_endmill_square(3.175, 19.05);
336 <gcpscad>   } else if (tool_number == 101) {
337 <gcpscad>     gcp_endmill_ball(3.175, 19.05);
338 <gcpscad>   } else if (tool_number == 301) {
339 <gcpscad>     gcp_endmill_v(90, 12.7);
340 <gcpscad>   } else if (tool_number == 302) {
341 <gcpscad>     gcp_endmill_v(60, 12.7);
342 <gcpscad>   } else if (tool_number == 390) {
343 <gcpscad>     gcp_endmill_v(90, 3.175);
344 <gcpscad>   }
345 <gcpscad> }
346 <gcpscad>

```

Each tool must be modeled in 3D using an OpenSCAD module:

`gcp_endmill_square`

```

347 <gcpscad> module gcp_endmill_square(es_diameter, es_flute_length) {
348 <gcpscad>   cylinder(r1=(es_diameter / 2), r2=(es_diameter / 2), h=es_flute_length, center=false);
349 <gcpscad> }
350 <gcpscad>

```

`gcp_endmill_ball`

```

351 <gcpscad> module gcp_endmill_ball(es_diameter, es_flute_length) {
352 <gcpscad>   translate([0, 0, (es_diameter / 2)]){
353 <gcpscad>     union(){
354 <gcpscad>       sphere(r=(es_diameter / 2));
355 <gcpscad>       cylinder(r1=(es_diameter / 2), r2=(es_diameter / 2), h=es_flute_length, center=false);
356 <gcpscad>     }
357 <gcpscad>   }
358 <gcpscad> }
359 <gcpscad>

```

`gcp_endmill_v`

```

360 <gcpscad> module gcp_endmill_v(es_v_angle, es_diameter) {
361 <gcpscad>   union(){
362 <gcpscad>     cylinder(r1=0, r2=(es_diameter / 2), h=((es_diameter / 2) / tan((es_v_angle / 2))),
363 <gcpscad>     translate([0, 0, ((es_diameter / 2) / tan((es_v_angle / 2)))]{
364 <gcpscad>       cylinder(r1=(es_diameter / 2), r2=(es_diameter / 2), h=((es_diameter * 8) / 2), center=false);
365 <gcpscad>     }
366 <gcpscad>   }
367 <gcpscad> }
368 <gcpscad>

```

## 2.3 File Handling

For writing to files it will be necessary to have commands for each step of working with the files.

There is a separate function for each type of file, and for DXFs, there are multiple file instances, one for each type of different type and size of tool which it is expected a project will work with.

```

popengcodefile
popendxfile
popendxlgblfile
popendxflgsqfile
popendxflgVfile
popendxfsmblfile
popendxfsmsqfile
popendxfsmVfile
popensvgfile
369 (gcpy)def popengcodefile(fn):
370 (gcpy)    global f
371 (gcpy)    f = open(fn, "w")
372 (gcpy)
373 (gcpy)def popendxfile(fn):
374 (gcpy)    global dxf
375 (gcpy)    dxf = open(fn, "w")
376 (gcpy)
377 (gcpy)def popendxlgblfile(fn):
378 (gcpy)    global dxflgbl
379 (gcpy)    dxflgbl = open(fn, "w")
380 (gcpy)
381 (gcpy)def popendxflgsqfile(fn):
382 (gcpy)    global dxfldsq
383 (gcpy)    dxflgsq = open(fn, "w")
384 (gcpy)
385 (gcpy)def popendxflgVfile(fn):
386 (gcpy)    global dxflgV
387 (gcpy)    dxflgV = open(fn, "w")
388 (gcpy)
389 (gcpy)def popendxfsmblfile(fn):
390 (gcpy)    global dxfsmb1
391 (gcpy)    dxfsmb1 = open(fn, "w")
392 (gcpy)
393 (gcpy)def popendxfsmsqfile(fn):
394 (gcpy)    global dxfsmsq
395 (gcpy)    dxfsmsq = open(fn, "w")
396 (gcpy)
397 (gcpy)def popendxfsmVfile(fn):
398 (gcpy)    global dx fsmV
399 (gcpy)    dx fsmV = open(fn, "w")
400 (gcpy)
401 (gcpy)def popensvgfile(fn):
402 (gcpy)    global svg
403 (gcpy)    svg = open(fn, "w")
404 (gcpy)

```

There will need to be matching OpenSCAD modules for the Python functions.

```

oopengcodefile
oopensvgfile
405 (pyscad) module oopengcodefile(fn) {
406 (pyscad)   popengcodefile(fn);
407 (pyscad) }
408 (pyscad)
409 (pyscad) module oopensvgfile(fn) {
410 (pyscad)   popensvgfile(fn);
411 (pyscad) }
412 (pyscad)
413 (pyscad) module oopendxfile(fn) {

```

```

414 <pyscad>     echo(fn);
415 <pyscad> popendxfile(fn);
416 <pyscad> }
417 <pyscad>
418 <pyscad> module oopendxflgblfile(fn) {
419 <pyscad>     popendxflgblfile(fn);
420 <pyscad> }
421 <pyscad>
422 <pyscad> module oopendxflgsqfile(fn) {
423 <pyscad>     popendxflgsqfile(fn);
424 <pyscad> }
425 <pyscad>
426 <pyscad> module oopendxflgVfile(fn) {
427 <pyscad>     popendxflgVfile(fn);
428 <pyscad> }
429 <pyscad>
430 <pyscad> module oopendxfsmblfile(fn) {
431 <pyscad>     popendxfsmblfile(fn);
432 <pyscad> }
433 <pyscad>
434 <pyscad> module oopendxfsmsqfile(fn) {
435 <pyscad>     echo(fn);
436 <pyscad>     popendxfsmsqfile(fn);
437 <pyscad> }
438 <pyscad>
439 <pyscad> module oopendxfsmVfile(fn) {
440 <pyscad>     popendxfsmVfile(fn);
441 <pyscad> }
442 <pyscad>

opengcodefile
  opensvgfile 443 <gcpscad> module opengcodefile(fn) {
  opendxfile 444 <gcpscad> if (generategcode == true) {
445 <gcpscad>     oopengcodefile(fn);
446 <gcpscad>     echo(fn);
447 <gcpscad>     owritecomment(fn);
448 <gcpscad> }
449 <gcpscad> }
450 <gcpscad>
451 <gcpscad> module opensvgfile(fn) {
452 <gcpscad> if (generatesvg == true) {
453 <gcpscad>     opensvgfile(fn);
454 <gcpscad>     echo(fn);
455 <gcpscad>     svgwriteone(str("<?xml version=",chr(34),"1.0",chr(34)," encoding=",chr(34),"UTF-8",
456 <gcpscad> // writesvglineend();
457 <gcpscad>     svgwriteone(str("<svg version=",chr(34),"1.1",chr(34)," xmlns=",chr(34),"http://www.w3.
458 <gcpscad> //<path d="M755.906 0 L755.906 377.953 L0 377.953 L0 0 L755.906 0 Z " stroke="black" str
459 <gcpscad>     svgwriteone(str("<path d=",chr(34),"M",stocklength*3.77953," 0 L",stocklength*3.77953,"
460 <gcpscad>     }
461 <gcpscad> }
462 <gcpscad>
463 <gcpscad> module opendxfile(fn) {
464 <gcpscad> if (generatedxf == true) {
465 <gcpscad>     oopendxfile(str(fn, ".dxf"));
466 <gcpscad> //     echo(fn);

```

```

467 (gpcscad)      dxfwriteone("0");
468 (gpcscad)      dxfwriteone("SECTION");
469 (gpcscad)      dxfwriteone("2");
470 (gpcscad)      dxfwriteone("ENTITIES");
471 (gpcscad)      dxfwriteone("0");
472 (gpcscad) if (large_ball_tool_no > 0) { oopendxflgblfile(str(fn, ".", large_ball_tool_no
473 (gpcscad)      dxfpreamble(large_ball_tool_no);
474 (gpcscad) }
475 (gpcscad) if (large_square_tool_no > 0) { oopendxflgsqfile(str(fn, ".", large_square_tool
476 (gpcscad)      dxfpreamble(large_square_tool_no);
477 (gpcscad) }
478 (gpcscad) if (large_V_tool_no > 0) { oopendxflgVfile(str(fn, ".", large_V_tool_no, ".dxf"
479 (gpcscad)      dxfpreamble(large_V_tool_no);
480 (gpcscad) }
481 (gpcscad) if (small_ball_tool_no > 0) { oopendxfsmblfile(str(fn, ".", small_ball_tool_no
482 (gpcscad)      dxfpreamble(small_ball_tool_no);
483 (gpcscad) }
484 (gpcscad) if (small_square_tool_no > 0) { oopendxfsmsqfile(str(fn, ".", small_square_tool
485 (gpcscad) //      echo(str("tool no", small_square_tool_no));
486 (gpcscad)      dxfpreamble(small_square_tool_no);
487 (gpcscad) }
488 (gpcscad) if (small_V_tool_no > 0) { oopendxfsmVfile(str(fn, ".", small_V_tool_no, ".dxf"
489 (gpcscad)      dxfpreamble(small_V_tool_no);
490 (gpcscad) }
491 (gpcscad) }
492 (gpcscad) }
493 (gpcscad)

```

writedxf            Once files have been opened they may be written to.  
writedxflgbl  
writedxflgsq

```

494 (gcpy)def writedxf(*arguments):
495 (gcpy)    line_to_write = ""
496 (gcpy)    for element in arguments:
497 (gcpy)        line_to_write += element
498 (gcpy)    dxf.write(line_to_write)
499 (gcpy)    dxf.write("\n")
500 (gcpy)
501 (gcpy)def writedxflgbl(*arguments):
502 (gcpy)    line_to_write = ""
503 (gcpy)    for element in arguments:
504 (gcpy)        line_to_write += element
505 (gcpy)    dxflgbl.write(line_to_write)
506 (gcpy)    print(line_to_write)
507 (gcpy)    dxflgbl.write("\n")
508 (gcpy)
509 (gcpy)def writedxflgsq(*arguments):
510 (gcpy)    line_to_write = ""
511 (gcpy)    for element in arguments:
512 (gcpy)        line_to_write += element
513 (gcpy)    dxflgsq.write(line_to_write)
514 (gcpy)    print(line_to_write)
515 (gcpy)    dxflgsq.write("\n")
516 (gcpy)
517 (gcpy)def writedxflgV(*arguments):
518 (gcpy)    line_to_write = ""

```

```

519 <gcpy>     for element in arguments:
520 <gcpy>         line_to_write += element
521 <gcpy>     dxflgV.write(line_to_write)
522 <gcpy>     print(line_to_write)
523 <gcpy>     dxflgV.write("\n")
524 <gcpy>
525 <gcpy>def writedxfsmb(*arguments):
526 <gcpy>     line_to_write = ""
527 <gcpy>     for element in arguments:
528 <gcpy>         line_to_write += element
529 <gcpy>     dxfsmb.write(line_to_write)
530 <gcpy>     print(line_to_write)
531 <gcpy>     dxfsmb.write("\n")
532 <gcpy>
533 <gcpy>def writedxfsmsq(*arguments):
534 <gcpy>     line_to_write = ""
535 <gcpy>     for element in arguments:
536 <gcpy>         line_to_write += element
537 <gcpy>     dxfsmsq.write(line_to_write)
538 <gcpy>     print(line_to_write)
539 <gcpy>     dxfsmsq.write("\n")
540 <gcpy>
541 <gcpy>def writedxfsmV(*arguments):
542 <gcpy>     line_to_write = ""
543 <gcpy>     for element in arguments:
544 <gcpy>         line_to_write += element
545 <gcpy>     dxfsmV.write(line_to_write)
546 <gcpy>     print(line_to_write)
547 <gcpy>     dxfsmV.write("\n")
548 <gcpy>
549 <gcpy>def writesvg(*arguments):
550 <gcpy>     line_to_write = ""
551 <gcpy>     for element in arguments:
552 <gcpy>         line_to_write += element
553 <gcpy>     svg.write(line_to_write)
554 <gcpy>     print(line_to_write)
555 <gcpy>
556 <gcpy>def pwritesvgline():
557 <gcpy>     svg.write("\n")
558 <gcpy>

```

owritecomment

```

    dxfwriteone
dxfwritelgbl
559 <pyscad> module owritecomment(comment) {
560 <pyscad>     writeln("(",comment,")");
561 <pyscad> }
562 <pyscad>
563 <pyscad> module dxfwriteone(first) {
564 <pyscad>     writedxf(first);
565 <pyscad>     // writeln(first);
566 <pyscad>     //     echo(first);
567 <pyscad> }
568 <pyscad>
569 <pyscad> module dxfwritelgbl(first) {
570 <pyscad>     writedxfzglbl(first);

```

```
571 (pyscad) }
572 (pyscad)
573 (pyscad) module dxfwritelgsq(first) {
574 (pyscad)   writedxflgsq(first);
575 (pyscad) }
576 (pyscad)
577 (pyscad) module dxfwritelgV(first) {
578 (pyscad)   writedxflgV(first);
579 (pyscad) }
580 (pyscad)
581 (pyscad) module dxfwritesmbl(first) {
582 (pyscad)   writedxfsmbL(first);
583 (pyscad) }
584 (pyscad)
585 (pyscad) module dxfwritesmsq(first) {
586 (pyscad)   writedxfsmsq(first);
587 (pyscad) }
588 (pyscad)
589 (pyscad) module dxfwritesmV(first) {
590 (pyscad)   writedxfsmV(first);
591 (pyscad) }
592 (pyscad)
593 (pyscad) module svgwriteone(first) {
594 (pyscad)   writesvg(first);
595 (pyscad) }
596 (pyscad)
597 (pyscad) module writesvglineend(first) {
598 (pyscad)   pwritesvgline();
599 (pyscad) }
600 (pyscad)
601 (pyscad) module owriteone(first) {
602 (pyscad)   writeln(first);
603 (pyscad) }
604 (pyscad)
605 (pyscad) module owritetwo(first, second) {
606 (pyscad)   writeln(first, second);
607 (pyscad) }
608 (pyscad)
609 (pyscad) module owritethree(first, second, third) {
610 (pyscad)   writeln(first, second, third);
611 (pyscad) }
612 (pyscad)
613 (pyscad) module owritefour(first, second, third, fourth) {
614 (pyscad)   writeln(first, second, third, fourth);
615 (pyscad) }
616 (pyscad)
617 (pyscad) module owritefive(first, second, third, fourth, fifth) {
618 (pyscad)   writeln(first, second, third, fourth, fifth);
619 (pyscad) }
620 (pyscad)
621 (pyscad) module owritesix(first, second, third, fourth, fifth, sixth) {
622 (pyscad)   writeln(first, second, third, fourth, fifth, sixth);
623 (pyscad) }
624 (pyscad)
```

```

625 <pyscad> module owriteseven(first, second, third, fourth, fifth, sixth, seventh) {
626 <pyscad> writeln(first, second, third, fourth, fifth, sixth, seventh);
627 <pyscad> }
628 <pyscad>
629 <pyscad> module owriteeight(first, second, third, fourth, fifth, sixth, seventh,eighth) {
630 <pyscad> writeln(first, second, third, fourth, fifth, sixth, seventh,eighth);
631 <pyscad> }
632 <pyscad>
633 <pyscad> module owritenine(first, second, third, fourth, fifth, sixth, seventh, eighth, ninth) {
634 <pyscad> writeln(first, second, third, fourth, fifth, sixth, seventh, eighth, ninth);
635 <pyscad> }
636 <pyscad>
637 <pyscad> module owriteten(first, second, third, fourth, fifth, sixth, seventh, eighth, ninth, tenth) {
638 <pyscad> writeln(first, second, third, fourth, fifth, sixth, seventh, eighth, ninth, tenth);
639 <pyscad> }
640 <pyscad>
641 <pyscad> module owriteeleven(first, second, third, fourth, fifth, sixth, seventh, eighth, ninth, tenth, eleventh) {
642 <pyscad> writeln(first, second, third, fourth, fifth, sixth, seventh, eighth, ninth, tenth, eleventh);
643 <pyscad> }
644 <pyscad>
645 <pyscad> module owritetwelve(first, second, third, fourth, fifth, sixth, seventh, eighth, ninth, tenth, eleventh, twelfth) {
646 <pyscad> writeln(first, second, third, fourth, fifth, sixth, seventh, eighth, ninth, tenth, eleventh, twelfth);
647 <pyscad> }
648 <pyscad>
649 <pyscad> module owritethirteen(first, second, third, fourth, fifth, sixth, seventh, eighth, ninth, tenth, eleventh, twelfth, thirteenth) {
650 <pyscad> writeln(first, second, third, fourth, fifth, sixth, seventh, eighth, ninth, tenth, eleventh, twelfth, thirteenth);
651 <pyscad> }
652 <pyscad>

dxfwrite
dxfpreamble
writesvgline 653 <gcpscad> module dxfwrite(tn,arg) {
654 <gcpscad> if (tn == large_ball_tool_no) {
655 <gcpscad>     dxfwritelgbl(arg);}
656 <gcpscad> if (tn == large_square_tool_no) {
657 <gcpscad>     dxfwritelgsq(arg);}
658 <gcpscad> if (tn == large_V_tool_no) {
659 <gcpscad>     dxfwritelgV(arg);}
660 <gcpscad> if (tn == small_ball_tool_no) {
661 <gcpscad>     dxfwritesmbl(arg);}
662 <gcpscad> if (tn == small_square_tool_no) {
663 <gcpscad>     dxfwritesmsq(arg);}
664 <gcpscad> if (tn == small_V_tool_no) {
665 <gcpscad>     dxfwritesmV(arg);}
666 <gcpscad> }
667 <gcpscad>
668 <gcpscad> module dxfpreamble(tn) {
669 <gcpscad> //     echo(str("dxfpreamble",small_square_tool_no));
670 <gcpscad>     dxfwrite(tn,"0");
671 <gcpscad>     dxfwrite(tn,"SECTION");
672 <gcpscad>     dxfwrite(tn,"2");
673 <gcpscad>     dxfwrite(tn,"ENTITIES");
674 <gcpscad>     dxfwrite(tn,"0");
675 <gcpscad> }
676 <gcpscad>
677 <gcpscad> module writesvgline(bx,by,ex,ey) {

```

```

678 (gcpscad) if (generatesvg == true) {
679 (gcpscad)     svgwriteone(str("<path d=", chr(34), "M", bx*3.77953, " ", by*3.77953, " L", ex*
680 (gcpscad)     })
681 (gcpscad) }
682 (gcpscad)
683 (gcpscad) module dxfbpl(tn,bx,by) {
684 (gcpscad)     dxfwrite(tn,"POLYLINE");
685 (gcpscad)     dxfwrite(tn,"8");
686 (gcpscad)     dxfwrite(tn,"default");
687 (gcpscad)     dxfwrite(tn,"66");
688 (gcpscad)     dxfwrite(tn,"1");
689 (gcpscad)     dxfwrite(tn,"70");
690 (gcpscad)     dxfwrite(tn,"0");
691 (gcpscad)     dxfwrite(tn,"0");
692 (gcpscad)     dxfwrite(tn,"VERTEX");
693 (gcpscad)     dxfwrite(tn,"8");
694 (gcpscad)     dxfwrite(tn,"default");
695 (gcpscad)     dxfwrite(tn,"70");
696 (gcpscad)     dxfwrite(tn,"32");
697 (gcpscad)     dxfwrite(tn,"10");
698 (gcpscad)     dxfwrite(tn,str(bx));
699 (gcpscad)     dxfwrite(tn,"20");
700 (gcpscad)     dxfwrite(tn,str(by));
701 (gcpscad)     dxfwrite(tn,"0");
702 (gcpscad) }
703 (gcpscad)
704 (gcpscad) module beginpolyline(bx,by,bz) {
705 (gcpscad) if (generatedxf == true) {
706 (gcpscad)     dxfwriteone("POLYLINE");
707 (gcpscad)     dxfwriteone("8");
708 (gcpscad)     dxfwriteone("default");
709 (gcpscad)     dxfwriteone("66");
710 (gcpscad)     dxfwriteone("1");
711 (gcpscad)     dxfwriteone("70");
712 (gcpscad)     dxfwriteone("0");
713 (gcpscad)     dxfwriteone("0");
714 (gcpscad)     dxfwriteone("VERTEX");
715 (gcpscad)     dxfwriteone("8");
716 (gcpscad)     dxfwriteone("default");
717 (gcpscad)     dxfwriteone("70");
718 (gcpscad)     dxfwriteone("32");
719 (gcpscad)     dxfwriteone("10");
720 (gcpscad)     dxfwriteone(str(bx));
721 (gcpscad)     dxfwriteone("20");
722 (gcpscad)     dxfwriteone(str(by));
723 (gcpscad)     dxfwriteone("0");
724 (gcpscad)     dxfbpl(current_tool(),bx,by);}
725 (gcpscad) }
726 (gcpscad)
727 (gcpscad) module dxfbpl(tn,bx,by) {
728 (gcpscad)     dxfwrite(tn,"VERTEX");
729 (gcpscad)     dxfwrite(tn,"8");
730 (gcpscad)     dxfwrite(tn,"default");
731 (gcpscad)     dxfwrite(tn,"70");

```



```

732 <gcpscad>    dxfwrite(tn,"32");
733 <gcpscad>    dxfwrite(tn,"10");
734 <gcpscad>    dxfwrite(tn,str(bx));
735 <gcpscad>    dxfwrite(tn,"20");
736 <gcpscad>    dxfwrite(tn,str(by));
737 <gcpscad>    dxfwrite(tn,"0");
738 <gcpscad> }
739 <gcpscad>
740 <gcpscad> module addpolyline(bx,by,bz) {
741 <gcpscad> if (generatedxf == true) {
742 <gcpscad>    dxfwriteone("VERTEX");
743 <gcpscad>    dxfwriteone("8");
744 <gcpscad>    dxfwriteone("default");
745 <gcpscad>    dxfwriteone("70");
746 <gcpscad>    dxfwriteone("32");
747 <gcpscad>    dxfwriteone("10");
748 <gcpscad>    dxfwriteone(str(bx));
749 <gcpscad>    dxfwriteone("20");
750 <gcpscad>    dxfwriteone(str(by));
751 <gcpscad>    dxfwriteone("0");
752 <gcpscad>    dxfcpl(current_tool(),bx,by);
753 <gcpscad>    }
754 <gcpscad> }
755 <gcpscad>
756 <gcpscad> module dxfcpl(tn) {
757 <gcpscad>    dxfwrite(tn,"SEQEND");
758 <gcpscad>    dxfwrite(tn,"0");
759 <gcpscad> }
760 <gcpscad>
761 <gcpscad> module closepolyline() {
762 <gcpscad> if (generatedxf == true) {
763 <gcpscad>    dxfwriteone("SEQEND");
764 <gcpscad>    dxfwriteone("0");
765 <gcpscad>    dxfcpl(current_tool());
766 <gcpscad>    }
767 <gcpscad> }
768 <gcpscad>
769 <gcpscad> module writecomment(comment) {
770 <gcpscad> if (generategcode == true) {
771 <gcpscad>    owritecomment(comment);
772 <gcpscad> }
773 <gcpscad> }
774 <gcpscad>

```

`pclosegcodefile` At the end of the project it will be necessary to close each file. In some  
`pclosesvgfile` instances it will be necessary to write additional information, depending on the  
`pclosedxfile` file format.

```

775 <gcpy>def pclosegcodefile():
776 <gcpy>    f.close()
777 <gcpy>
778 <gcpy>def pclosesvgfile():
779 <gcpy>    svg.close()
780 <gcpy>
781 <gcpy>def pclosedxfile():

```

```

782 (gcpy)    dxf.close()
783 (gcpy)
784 (gcpy)def pclosedxflgblfile():
785 (gcpy)    dxflgbl.close()
786 (gcpy)
787 (gcpy)def pclosedxflgsqfile():
788 (gcpy)    dxflgsq.close()
789 (gcpy)
790 (gcpy)def pclosedxflgVfile():
791 (gcpy)    dxflgV.close()
792 (gcpy)
793 (gcpy)def pclosedxfsmblfile():
794 (gcpy)    dxfsmb1.close()
795 (gcpy)
796 (gcpy)def pclosedxfsmsqfile():
797 (gcpy)    dxfsmsq.close()
798 (gcpy)
799 (gcpy)def pclosedxfsmVfile():
800 (gcpy)    dxfsmV.close()
801 (gcpy)

oclosegcodefile
    oclosedxfile
oclosedxflgblfile 802 (pyscad) module oclosegcodefile() {
803 (pyscad)    pclosegcodefile();
804 (pyscad) }
805 (pyscad)
806 (pyscad) module oclosedxfile() {
807 (pyscad)    pclosedxfile();
808 (pyscad) }
809 (pyscad)
810 (pyscad) module oclosedxflgblfile() {
811 (pyscad)    pclosedxflgblfile();
812 (pyscad) }
813 (pyscad)
814 (pyscad) module oclosedxflgsqfile() {
815 (pyscad)    pclosedxflgsqfile();
816 (pyscad) }
817 (pyscad)
818 (pyscad) module oclosedxflgVfile() {
819 (pyscad)    pclosedxflgVfile();
820 (pyscad) }
821 (pyscad)
822 (pyscad) module oclosedxfsmblfile() {
823 (pyscad)    pclosedxfsmblfile();
824 (pyscad) }
825 (pyscad)
826 (pyscad) module oclosedxfsmsqfile() {
827 (pyscad)    pclosedxfsmsqfile();
828 (pyscad) }
829 (pyscad)
830 (pyscad) module oclosedxfsmVfile() {
831 (pyscad)    pclosedxfsmVfile();
832 (pyscad) }
833 (pyscad)

```

```

834 <pyscad> module oclosesvgfile() {
835 <pyscad>   pclosesvgfile();
836 <pyscad> }
837 <pyscad>

closecodefile
  dxfpreamble 838 <gcpscad> module closecodefile() {
  closedxfile 839 <gcpscad> if (generategcode == true) {
840 <gcpscad>   owriteone("M05");
841 <gcpscad>   owriteone("M02");
842 <gcpscad>   oclosecodefile();
843 <gcpscad> }
844 <gcpscad> }
845 <gcpscad>
846 <gcpscad> module dxfpreamble(arg) {
847 <gcpscad>   dxfwrite(arg,"ENDSEC");
848 <gcpscad>   dxfwrite(arg,"0");
849 <gcpscad>   dxfwrite(arg,"EOF");
850 <gcpscad> }
851 <gcpscad>
852 <gcpscad> module closedxfile() {
853 <gcpscad> if (generatedxf == true) {
854 <gcpscad>   dxfwriteone("ENDSEC");
855 <gcpscad>   dxfwriteone("0");
856 <gcpscad>   dxfwriteone("EOF");
857 <gcpscad>   oclosedxfile();
858 <gcpscad>   echo("CLOSING");
859 <gcpscad> if (large_ball_tool_no > 0) { dxfpreamble(large_ball_tool_no);
860 <gcpscad>   oclosedxflgblfile();
861 <gcpscad> }
862 <gcpscad> if (large_square_tool_no > 0) { dxfpreamble(large_square_tool_no);
863 <gcpscad>   oclosedxflgsqfile();
864 <gcpscad> }
865 <gcpscad> if (large_V_tool_no > 0) { dxfpreamble(large_V_tool_no);
866 <gcpscad>   oclosedxflgVfile();
867 <gcpscad> }
868 <gcpscad> if (small_ball_tool_no > 0) { dxfpreamble(small_ball_tool_no);
869 <gcpscad>   oclosedxfsmbfile();
870 <gcpscad> }
871 <gcpscad> if (small_square_tool_no > 0) { dxfpreamble(small_square_tool_no);
872 <gcpscad>   oclosedxfsmsqfile();
873 <gcpscad> }
874 <gcpscad> if (small_V_tool_no > 0) { dxfpreamble(small_V_tool_no);
875 <gcpscad>   oclosedxfsmVfile();
876 <gcpscad> }
877 <gcpscad> }
878 <gcpscad> }
879 <gcpscad>
880 <gcpscad> module closesvgfile() {
881 <gcpscad> if (generatesvg == true) {
882 <gcpscad>   svgwriteone("</svg> ");
883 <gcpscad>   oclosesvgfile();
884 <gcpscad>   echo("CLOSING SVG");
885 <gcpscad> }
886 <gcpscad> }

```

```
887 (gcpscad)
```

## 2.4 Movement and Cutting

otm With all the scaffolding in place, it is possible to model tool movement and  
ocut cutting and to write out files which represent the desired machine motions.

```
orapid 888 (gcpscad) module otm(ex, ey, ez, r,g,b) {
889 (gcpscad) color([r,g,b]) hull(){
890 (gcpscad)     translate([xpos(), ypos(), zpos()]){
891 (gcpscad)         select_tool(current_tool());
892 (gcpscad)     }
893 (gcpscad)     translate([ex, ey, ez]){
894 (gcpscad)         select_tool(current_tool());
895 (gcpscad)     }
896 (gcpscad) }
897 (gcpscad) oset(ex, ey, ez);
898 (gcpscad) }
899 (gcpscad)
900 (gcpscad) module ocut(ex, ey, ez) {
901 (gcpscad) //color([0.2,1,0.2]) hull(){
902 (gcpscad) otm(ex, ey, ez, 0.2,1,0.2);
903 (gcpscad) }
904 (gcpscad)
905 (gcpscad) module orapid(ex, ey, ez) {
906 (gcpscad) //color([0.93,0,0]) hull(){
907 (gcpscad) otm(ex, ey, ez, 0.93,0,0);
908 (gcpscad) }
909 (gcpscad)
910 (gcpscad) module rapidbx(bx, by, bz, ex, ey, ez) {
911 (gcpscad) // writeln("G0 X",bx," Y", by, "Z", bz);
912 (gcpscad) if (generategcode == true) {
913 (gcpscad)     writecomment("rapid");
914 (gcpscad)     owritesix("G0 X",str(ex)," Y", str(ey), " Z", str(ez));
915 (gcpscad) }
916 (gcpscad)     orapid(ex, ey, ez);
917 (gcpscad) }
918 (gcpscad)
919 (gcpscad) module rapid(ex, ey, ez) {
920 (gcpscad) // writeln("G0 X",bx," Y", by, "Z", bz);
921 (gcpscad) if (generategcode == true) {
922 (gcpscad)     writecomment("rapid");
923 (gcpscad)     owritesix("G0 X",str(ex)," Y", str(ey), " Z", str(ez));
924 (gcpscad) }
925 (gcpscad)     orapid(ex, ey, ez);
926 (gcpscad) }
927 (gcpscad)
928 (gcpscad) module movetosafez() {
929 (gcpscad) //this should be move to retract height
930 (gcpscad) if (generategcode == true) {
931 (gcpscad)     writecomment("Move to safe Z to avoid workholding");
932 (gcpscad)     owriteone("G53G0Z-5.000");
933 (gcpscad) }
934 (gcpscad)     orapid(getxpos(), getypos(), retractheight+55);
```

```

935 <gcpscad> }
936 <gcpscad>
937 <gcpscad> module begintoolpath(bx,by,bz) {
938 <gcpscad> if (generategcode == true) {
939 <gcpscad>   writecomment("PREPOSITION FOR RAPID PLUNGE");
940 <gcpscad>     owritefour("GOX", str(bx), "Y",str(by));
941 <gcpscad>     owritetwo("Z", str(bz));
942 <gcpscad>   }
943 <gcpscad>   orapid(bx,by,bz);
944 <gcpscad> }
945 <gcpscad>
946 <gcpscad> module movetosafeheight() {
947 <gcpscad> //this should be move to machine position
948 <gcpscad> if (generategcode == true) {
949 <gcpscad> // writecomment("PREPOSITION FOR RAPID PLUNGE");Z25.650
950 <gcpscad> //G1Z24.663F381.0 ,"F",str(plunge)
951 <gcpscad> if (zeroheight == "Top") {
952 <gcpscad>   owritetwo("Z",str(retractheight));
953 <gcpscad> }
954 <gcpscad> }
955 <gcpscad>   orapid(getxpos(), getypos(), retractheight+55);
956 <gcpscad> }
957 <gcpscad>
958 <gcpscad> module cutoneaxis_setfeed(axis,depth,feed) {
959 <gcpscad> if (generategcode == true) {
960 <gcpscad> // writecomment("PREPOSITION FOR RAPID PLUNGE");Z25.650
961 <gcpscad> //G1Z24.663F381.0 ,"F",str(plunge) G1Z7.612F381.0
962 <gcpscad> if (zeroheight == "Top") {
963 <gcpscad>   owritefive("G1",axis,str(depth),"F",str(feed));
964 <gcpscad> }
965 <gcpscad> }
966 <gcpscad> if (axis == "X") {setxpos(depth);}
967 <gcpscad> if (axis == "Y") {setypos(depth);}
968 <gcpscad> if (axis == "Z") {setzpos(depth);}
969 <gcpscad> }
970 <gcpscad>
971 <gcpscad> module cut(ex, ey, ez) {
972 <gcpscad> // writeln("GO X",bx," Y", by, "Z", bz);
973 <gcpscad> if (generategcode == true) {
974 <gcpscad> // writecomment("rapid");
975 <gcpscad> owritesix("G1 X",str(ex)," Y", str(ey), " Z", str(ez));
976 <gcpscad> }
977 <gcpscad> if (generatesvg == true) {
978 <gcpscad> // owritesix("G1 X",str(ex)," Y", str(ey), " Z", str(ez));
979 <gcpscad> //   orapid(getxpos(), getypos(), retractheight+5);
980 <gcpscad>   writesvgline(getxpos(),getypos(),ex,ey);
981 <gcpscad> }
982 <gcpscad> ocut(ex, ey, ez);
983 <gcpscad> }
984 <gcpscad>
985 <gcpscad> module cutwithfeed(ex, ey, ez, feed) {
986 <gcpscad> // writeln("GO X",bx," Y", by, "Z", bz);
987 <gcpscad> if (generategcode == true) {
988 <gcpscad> // writecomment("rapid");

```

```

989 (gcpscad) owriteeight("G1 X",str(ex)," Y", str(ey), " Z", str(ez),"F",str(feed));
990 (gcpscad) }
991 (gcpscad) ocut(ex, ey, ez);
992 (gcpscad) }
993 (gcpscad)
994 (gcpscad) module endtoolpath() {
995 (gcpscad) if (generategcode == true) {
996 (gcpscad) //Z31.750
997 (gcpscad) // owriteone("G53G0Z-5.000");
998 (gcpscad)     owritetwo("Z",str(retractheight));
999 (gcpscad) }
1000 (gcpscad)     orapid(getxpos(),getypos(),retractheight);
1001 (gcpscad) }

```

### 3 gcodepreview\_template.scad

```

1002 (gcptmpl) //!OpenSCAD
1003 (gcptmpl)
1004 (gcptmpl) use <gcodepreview.py>;
1005 (gcptmpl) use <pygcodepreview.scad>;
1006 (gcptmpl) include <gcodepreview.scad>;
1007 (gcptmpl)
1008 (gcptmpl) $fa = 2;
1009 (gcptmpl) $fs = 0.125;
1010 (gcptmpl)
1011 (gcptmpl) /* [Export] */
1012 (gcptmpl) Base_filename = "export";
1013 (gcptmpl)
1014 (gcptmpl) /* [Export] */
1015 (gcptmpl) generatedxf = true;
1016 (gcptmpl)
1017 (gcptmpl) /* [Export] */
1018 (gcptmpl) generategcode = true;
1019 (gcptmpl)
1020 (gcptmpl) /* [Export] */
1021 (gcptmpl) generatesvg = false;
1022 (gcptmpl)
1023 (gcptmpl) /* [CAM] */
1024 (gcptmpl) toolradius = 1.5875;
1025 (gcptmpl) /* [CAM] */
1026 (gcptmpl) large_ball_tool_no = 0; // [0:0,111:111,101:101,202:202]
1027 (gcptmpl)
1028 (gcptmpl) /* [CAM] */
1029 (gcptmpl) large_square_tool_no = 0; // [0:0,112:112,102:102,201:201]
1030 (gcptmpl)
1031 (gcptmpl) /* [CAM] */
1032 (gcptmpl) large_V_tool_no = 0; // [0:0,301:301,690:690]
1033 (gcptmpl)
1034 (gcptmpl) /* [CAM] */
1035 (gcptmpl) small_ball_tool_no = 0; // [0:0,121:121,111:111,101:101]
1036 (gcptmpl)
1037 (gcptmpl) /* [CAM] */
1038 (gcptmpl) small_square_tool_no = 102; // [0:0,122:122,112:112,102:102]

```

```

1039 <gcptmpl>
1040 <gcptmpl> /* [CAM] */
1041 <gcptmpl> small_V_tool_no = 0; // [0:0,390:390,301:301]
1042 <gcptmpl>
1043 <gcptmpl> /* [Feeds and Speeds] */
1044 <gcptmpl> plunge = 100;
1045 <gcptmpl> /* [Feeds and Speeds] */
1046 <gcptmpl> feed = 400;
1047 <gcptmpl> /* [Feeds and Speeds] */
1048 <gcptmpl> speed = 16000;
1049 <gcptmpl> /* [Feeds and Speeds] */
1050 <gcptmpl> square_ratio = 1.0; // [0.25:2]
1051 <gcptmpl> /* [Feeds and Speeds] */
1052 <gcptmpl> small_V_ratio = 0.75; // [0.25:2]
1053 <gcptmpl> /* [Feeds and Speeds] */
1054 <gcptmpl> large_V_ratio = 0.875; // [0.25:2]
1055 <gcptmpl>
1056 <gcptmpl> /* [Stock] */
1057 <gcptmpl> stocklength = 219;
1058 <gcptmpl> /* [Stock] */
1059 <gcptmpl> stockwidth = 150;
1060 <gcptmpl> /* [Stock] */
1061 <gcptmpl> stockthickness = 8.35;
1062 <gcptmpl> /* [Stock] */
1063 <gcptmpl> zeroheight = "Top"; // [Top, Bottom]
1064 <gcptmpl> /* [Stock] */
1065 <gcptmpl> stockorigin = "Center"; // [Lower-Left, Center-Left, Top-Left, Center]
1066 <gcptmpl> /* [Stock] */
1067 <gcptmpl> retractheight = 9;
1068 <gcptmpl>
1069 <gcptmpl> filename_gcode = str(Base_filename, ".nc");
1070 <gcptmpl> filename_dxf = str(Base_filename);
1071 <gcptmpl> filename_svg = str(Base_filename, ".svg");
1072 <gcptmpl>
1073 <gcptmpl>.opengcodefile(filename_gcode);
1074 <gcptmpl> .opendxfile(filename_dxf);
1075 <gcptmpl>
1076 <gcptmpl> difference() {
1077 <gcptmpl> setupstock(stocklength, stockwidth, stockthickness, zeroheight, stockorigin);
1078 <gcptmpl>
1079 <gcptmpl> movetosafez();
1080 <gcptmpl>
1081 <gcptmpl> toolchange(small_square_tool_no,speed * square_ratio);
1082 <gcptmpl>
1083 <gcptmpl> begintoolpath(0,0,0.25);
1084 <gcptmpl> beginpolyline(0,0,0.25);
1085 <gcptmpl>
1086 <gcptmpl> cutoneaxis_setfeed("Z",-1,plunge*square_ratio);
1087 <gcptmpl>
1088 <gcptmpl> cutwithfeed(stocklength/2,stockwidth/2,-stockthickness,feed);
1089 <gcptmpl> addpolyline(stocklength/2,stockwidth/2,-stockthickness);
1090 <gcptmpl>
1091 <gcptmpl> endtoolpath();
1092 <gcptmpl> closepolyline();

```

## References

# Index

```

closedxfile= \subitem *+closedxfile+ (routine) \usage{19} \subitem *+closedxfile+ (routine) \usage{19} \subitem *+closedxfile+ (routine) \usage{19}
closedxfgblfile= \subitem *+closedxfgblfile+ (routine) \usage{19} \subitem *+closedxfgblfile+ (routine) \usage{19} \subitem *+closedxfgblfile+ (routine) \usage{19}
currenttool= \subitem *+currenttool+ (routine) \usage{19} \subitem *+currenttool+ (routine) \usage{19} \subitem *+currenttool+ (routine) \usage{19}
*+currenttool+ oopengcodefile= \subitem *+oopengcodefile+ (routine) \usage{19} \subitem *+oopengcodefile+ (routine) \usage{19} \subitem *+oopengcodefile+ (routine) \usage{19}
(variable) , 4 oopensvgfile= \subitem *+oopensvgfile+ (routine) \usage{19} \subitem *+oopensvgfile+ (routine) \usage{19} \subitem *+oopensvgfile+ (routine) \usage{19}
opendxfile= \subitem *+opendxfile+ (routine) \usage{19} \subitem *+opendxfile+ (routine) \usage{19} \subitem *+opendxfile+ (routine) \usage{19}
Documents= \subitem *+Documents+ \hdindex{33} {code} {33} \subitem *+Documents+ \hdindex{33} {code} {33} \subitem *+Documents+ \hdindex{33} {code} {33}
dxfpreamble= \subitem *+dxfpreamble+ (routine) \usage{19} \subitem *+dxfpreamble+ (routine) \usage{19} \subitem *+dxfpreamble+ (routine) \usage{19}
dxfpreamble= \subitem *+dxfpreamble+ (routine) \usage{15} \subitem *+dxfpreamble+ (routine) \usage{15} \subitem *+dxfpreamble+ (routine) \usage{15}
dxfwrite= \subitem *+dxfwrite+ (routine) \usage{15} \subitem *+dxfwrite+ (routine) \usage{15} \subitem *+dxfwrite+ (routine) \usage{15}
dxfwritelgbl= \subitem *+dxfwritelgbl+ (routine) \usage{13} \subitem *+dxfwritelgbl+ (routine) \usage{13} \subitem *+dxfwritelgbl+ (routine) \usage{13}
dxfwwriteone= \subitem *+dxfwwriteone+ (routine) \usage{13} \subitem *+dxfwwriteone+ (routine) \usage{13} \subitem *+dxfwwriteone+ (routine) \usage{13}
*+dxfwwriteone+ \subitem *+osettool+ (routine) \usage{13} \subitem *+osettool+ (routine) \usage{13} \subitem *+osettool+ (routine) \usage{13}
osettool= \subitem *+osettool+ (routine) \usage{13} \subitem *+osettool+ (routine) \usage{13} \subitem *+osettool+ (routine) \usage{13}
osetupstock= \subitem *+osetupstock+ (routine) \usage{5} \subitem *+osetupstock+ (routine) \usage{5} \subitem *+osetupstock+ (routine) \usage{5}
gcp endmill otm= \subitem *+otm+ (routine) \usage{20} \subitem *+otm+ (routine) \usage{20} \subitem *+otm+ (routine) \usage{20}
ball= \subitem *+gcp endmill ball+ (routine) \usage{9} \subitem *+gcp endmill ball+ (routine) \usage{9} \subitem *+gcp endmill ball+ (routine) \usage{9}
gcp endmill owritecomment= \subitem *+owritecomment+ (routine) \usage{9} \subitem *+owritecomment+ (routine) \usage{9} \subitem *+owritecomment+ (routine) \usage{9}
square= \subitem *+gcp endmill square+ (routine) \usage{9} \subitem *+gcp endmill square+ (routine) \usage{9} \subitem *+gcp endmill square+ (routine) \usage{9}
gcp endmill pclosexfile= \subitem *+pclosexfile+ (routine) \usage{19} \subitem *+pclosexfile+ (routine) \usage{19} \subitem *+pclosexfile+ (routine) \usage{19}
v= \subitem *+gcp endmill v+ (routine) \usage{19} \subitem *+gcp endmill v+ (routine) \usage{19} \subitem *+gcp endmill v+ (routine) \usage{19}
getxpos= \subitem *+getxpos+ (routine) \usage{7} \subitem *+getxpos+ (routine) \usage{7} \subitem *+getxpos+ (routine) \usage{7}
getypos= \subitem *+getypos+ (routine) \usage{7} \subitem *+getypos+ (routine) \usage{7} \subitem *+getypos+ (routine) \usage{7}
getzpos= \subitem *+getzpos+ (routine) \usage{7} \subitem *+getzpos+ (routine) \usage{7} \subitem *+getzpos+ (routine) \usage{7}
libraries= \subitem *+libraries+ \hdindex{33} {code} {33} \subitem *+libraries+ \hdindex{33} {code} {33} \subitem *+libraries+ \hdindex{33} {code} {33}
mpx= \subitem *+mpx+ (variable) \usage{4} \subitem *+mpx+ (variable) \usage{4} \subitem *+mpx+ (variable) \usage{4}
mpy= \subitem *+mpy+ (variable) \usage{4} \subitem *+mpy+ (variable) \usage{4} \subitem *+mpy+ (variable) \usage{4}
mpz= \subitem *+mpz+ (variable) \usage{4} \subitem *+mpz+ (variable) \usage{4} \subitem *+mpz+ (variable) \usage{4}
popengcodefile= \subitem *+popengcodefile+ (routine) \usage{19} \subitem *+popengcodefile+ (routine) \usage{19} \subitem *+popengcodefile+ (routine) \usage{19}
n= \subitem *+n+ \hdindex{33} {code} {33} \subitem *+n+ \hdindex{33} {code} {33} \subitem *+n+ \hdindex{33} {code} {33}
507, 515, 523, psettool= \subitem *+psettool+ (routine) \usage{8} \subitem *+psettool+ (routine) \usage{8} \subitem *+psettool+ (routine) \usage{8}
531, 539, 547, 557 psetupstock= \subitem *+psetupstock+ (routine) \usage{4} \subitem *+psetupstock+ (routine) \usage{4} \subitem *+psetupstock+ (routine) \usage{4}
psetxpos= \subitem *+psetxpos+ (routine) \usage{7} \subitem *+psetxpos+ (routine) \usage{7} \subitem *+psetxpos+ (routine) \usage{7}
oclosedxfile= \subitem *+oclosedxfile+ (routine) \usage{19} \subitem *+oclosedxfile+ (routine) \usage{19} \subitem *+oclosedxfile+ (routine) \usage{19}
oclosedxfgblfile= \subitem *+oclosedxfgblfile+ (routine) \usage{19} \subitem *+oclosedxfgblfile+ (routine) \usage{19} \subitem *+oclosedxfgblfile+ (routine) \usage{19}

```



```

routines:~pclosedxfile= \subitem **pclosedxfile+ \usage{7} \subitem **users+ \hdclindex{33}{code}
routines:~pclosegcodefile= \subitem **pclosegcodefile+ \usage{7}
routines:~pclosesvgfile= \subitem **pclosesvgfile+ \usage{7} \subitem **currenttool+ \usage{7}
routines:~pcurrenttool= \subitem **pcurrenttool+ \usage{8} \subitem **toolchange+ \usage{8} \subitem **xpx+ \usage{4}
routines:~popendxfile= \subitem **popendxfile+ \usage{12} \subitem **mpy+ \usage{4}
routines:~popendxflgsqfile= \subitem **popendxflgsqfile+ \usage{12} \subitem **wrote+ \usage{4}
routines:~popendxflgVfile= \subitem **popendxflgVfile+ \usage{12} \subitem **wrote+ \usage{4}
routines:~popendxfsmblfile= \subitem **popendxfsmblfile+ \usage{10} \subitem **tool number+ \usage{10}
routines:~popendxfsmsqfile= \subitem **popendxfsmsqfile+ \usage{15}
routines:~popendxfsmVfile= \subitem **popendxfsmVfile+ \usage{15}
routines:~popendxlgblfile= \subitem **popendxlgblfile+ \usage{19} \subitem **writedxflgbl+ (routine)
routines:~popengcodefile= \subitem **popengcodefile+ \usage{19} \subitem **writedxflgsq+ (routine)
routines:~popensvgfile= \subitem **popensvgfile+ \usage{10} \subitem **writelnt+ (routine) , \usage{10}
routines:~psettool= \subitem **psettool+ \usage{8} \subitem **writesvgline+ (routine) , \usage{8}
routines:~psetupstock= \subitem **psetupstock+ \usage{4} \subitem **setupstock+ (routine) , \usage{4}
routines:~psetxpos= \subitem **psetxpos+ \usage{7} \subitem **xpos+ (routine) , \usage{7}
routines:~psetypos= \subitem **psetypos+ \usage{7} \subitem **setypos+ (routine) , \usage{7}
routines:~psetzpos= \subitem **psetzpos+ \usage{7} \subitem **ypos+ (routine) , \usage{7}
routines:~selecttool= \subitem **selecttool+ \usage{9} \subitem **tool number+ (variable) , \usage{9}
routines:~setupstock= \subitem **setupstock+ \usage{5} \subitem **change+ \usage{5} \subitem **zpos+ \usage{5}

```