Guide to editing a Vectric Post Processor

Vectric
Passionate About CNC

Version 1.1.1

November 17th 2010

Disclaimer

All CNC machines (routing, engraving, and milling) are potentially dangerous and because Vectric Ltd has no control over how the software described in this document might be used, Vectric Ltd or any associated Resellers cannot accept responsibility for any loss or damage to the work piece, machine or any individual, howsoever caused by misusing the software. Extreme care should always be taken and the output from the software thoroughly checked before sending it to a CNC machine.

The information in this manual may be subject to change without any prior notice. The software described in this manual is supplied under the terms and conditions of the software license agreement and may only be used in accordance with the terms of this agreement.

Vectric Ltd
Unit 3 Dunstall Court
Astwood Lane
Feckenham
B96 6QH
UK

www.vectric.com

E-mail info@vectric.com
Phone +44 (0) 1527 460 459
Fax +44 (0) 1527 460 459

Table of Contents

DOCUMENT REVISION 4

INTRODUCTION 5

WHAT DOES THE POST PROCESSOR DO? 5
POST PROCESSOR SECTIONS 6
File Comments 6
Global File Statements 6
Tape Splitting Support 8
Line Terminating Characters 8
Block Numbering 8
VARIABLES 9
Format of Variables 11
A Typical Variable 12
Formatting the Output Value 12
Optional Format Flags 13
Default Formatting for Variables 14
MULTIPLIER VALUE 19
POST PROCESSOR BLOCKS 20
HEADER 20
TOOLCHANGE 21
NEW_SEGMENT 21
INITIAL_RAPID_MOVE 21
RAPID_MOVE 22
FIRST_FEED_MOVE 22
FEED_MOVE 22
FIRST_CW_ARC_MOVE 22
CW_ARC_MOVE 23
FIRST_CCW_ARC_MOVE 23
CCW_ARC_MOVE 23
FOOTER 24
OTHER LESS FREQUENTLY USED SECTIONS. 25
FEED_RATE_CHANGE 25
FIRST_PLUNGE_MOVE 25
PLUNGE_MOVE 25
RETRACT_MOVE 26
Special Characters 27
EDITING A VECTRIC POST PROCESSOR – AN EXAMPLE: ADDING TOOL-CHANGE COMMANDS. 28
EDITING A VECTRIC POST PROCESSOR – AN EXAMPLE: CHANGING THE FILE EXTENSION. 33
POST PROCESSOR DIFFERENCES BETWEEN DIFFERENT VECTRIC PRODUCTS. 35
Cut3D (Release 1.025) – Unsupported Commands and Variables 35
PhotoVCarve (Release 1.102) – Unsupported Commands and Variables 35
Cut2D (Release 1.100) – Unsupported Commands and Variables 35
TIPS AND TRICKS. 36
POST PROCESSOR NAMING CONVENTIONS. 37

Document revision

Revision Number	Revision Date	Reason
1.0.0	July 12th 2009	Written.
1.1.0	October 1st 2009	Documented new post processor features for Aspire V2.5
1.1.1	November 17th 2010	Amended file location for PostP folder in new versions.

Introduction

What does the Post Processor Do?

The post processor is the section of the program that converts the XYZ coordinates for the tool moves into a format that is suitable for a particular router or machine tool.
This document details how to create and edit the configuration files that customize the output from the program to suit a particular machine control.

Below are sections of a typical program that has been post processed into both G-Code and HPGL.

G-Code example

```
T1 M6
G17
G0 Z4.5000
G0 X0.0000 Y0.0000 S12000 M3
G0 X2.4567 Y7.8342 Z0.2500
G1 Z-0.0500 F5.0
G3 X3.3784 Y8.7559 I0.0000 J0.9218 F66.0
G3 X2.4567 Y9.6777 I-0.9218 J0.0000
G3 X1.5349 Y8.7559 I0.0000 J-0.9218
```

…

HPGL-Code example

```
IN;PA;
PU2496,7960;
PD2496,7960;
AA2496,8896,90.000
AA2496,8896,90.000
AA2496,8896,90.000
AA2496,8896,90.000
PU2496,7960;
PU2496,6096;
```

…

Machine controller manufacturers will often customize the file format required for programs to run on a particular machine in order to optimise the control to suit the individual characteristics of that machine.
The Vectric post processor uses simple text based configuration files, to enable the user to tailor a configuration file, should they wish to do so.

Post Processor Sections

Vectric post processors are broken down into sections to aid clarity, try to write your post processors in a similar style to aid debugging.

File Comments

A section where you can describe the post processor and record any changes to the post processor, each line is a comment and starts with a '+' character or a '|' character.

```
+ History
+ Who     When      What
+ ======== ========== ==========================
+ Tony     14/07/2006 Written
+ Mark     26/08/2008 Combined ATC commands, stop spindle on TC
+================================================
```

Global File Statements.

Statements are items that are either used only once, or have static values throughout the file. Write statement names in upper case letters for clarity.

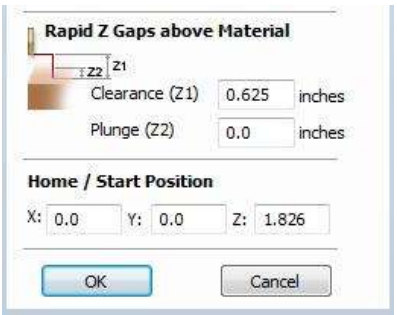
POST_NAME = "Text Output Arcs (mm) (*.txt)" *The name that will appear in the post processor list.*

FILE_EXTENSION = "txt" *The file extension that the file will be given.*

UNITS = "MM" *The units that the file outputs (INCHES or MM).*

PRINT_DIRECT = "YES" *The machine tool manufacturer has supplied a driver (usually a printer driver), that can directly accept the NC file output.*
For an example, see the file: Generic HPGL_Arcs.pp

RAPID_PLUNGE_TO_STARTZ = "YES" *Indicates that plunge moves to Plunge (Z2) height (that is set on the material setup form), are rapid moves.*



DIRECT_OUTPUT = "Display Name | Manufacturers.Document" *The control software uses a document interface that can directly accept the NC file output.*

Example: DIRECT_OUTPUT = "Mach|Mach4.Document"

For an example, see the file: Mach2_3_ATC_Arcs_inch.pp

ROTARY_WRAP_Y = A *The moves in the Y axis are to be wrapped around a cylinder of the specified diameter. The "Y" values will be output as "A"*

For an example, see the file: Mach2_3-WY_ATC_Arcs_mm.pp

ROTARY_WRAP_X = **B** *The moves in the X axis are to be wrapped around a cylinder of the specified diameter. The "X" values will be output as "B"*

For an example, see the file: Mach2_3-WX_ATC_Arcs_mm.pp

SPINDLE_SPEED_RANGE = 1 15 4500 15000 *Spindle speed for this machine is output as a range of integer numbers between 1 and 15 representing the actual speed in RPM of the spindle, (between 4500 and 15000 RPM in the quoted example).*

For an example, see the file: Roland_MDX-40_mm.pp

SUBSTITUTE = "*O1 S1 O2 S2 On Sn* " *This command allows you to substitute a character output within the variables (such as [TOOL_NAME]) and substitute that character, with another. This feature can be useful for those cases where particular characters cause errors on an NC control.*

The characters are entered in pairs, Original – Substituted.

For example MACH 3 control software uses parentheses as comment delimiters, and does not allow nested comments. Most tools within the Vectric Tool Database have parentheses within the "Name" section; if these names are output, this would cause an error within Mach3. The command SUBSTITUTE = "({)} " *would convert the* () *characters to* {} *characters, and avoid this error.*

For an example, see the file: Mach2_3_ATC_Arcs_inch.pp

Tape Splitting Support

TAPE_SPLITTING = " "

Format is : TAPE_SPLITTING = MAX_NUM_LINES LINE_TOL "FILENAME_FORMAT" START_INDEX INDEX_ON_FIRST_FILE

For example a command of TAPE_SPLITTING = 1000 100 "%s_%d.tap" 1 "YES"
would lead to ...

Output will be split into multiple files of a maximum of 1000 lines (+ however many lines in there are within the footer section of the post processor), if a retract move exists after line 900 (1000 – 100), the file will be split at that move. If the file was called "toolpath" the split files would be named toolpath_1.tap, toolpath_2.tap etc. The first toolpath output will be "toolpath_ 1.tap" there will be no file named "toolpath" without an index number, (as INDEX_ON_FIRST_FILE= YES is used), unless the file was less than 1000 lines long, in which case the file would not be split.

For an example, see the file: Busellato_Gen_Arcs_TS_mm.pp

Note: Some controllers that require NC files to be split, also have limitations on the number of characters within a filename. For example they may require the file to be named with the MSDOS style 8.3 filename format. This should be considered when naming the output file.

Line Terminating Characters

LINE_ENDING = "[13][10]"

Decimal values of the characters appended to each separate line of the post processed file. (Will usually be [13][10], (Carriage return, Line feed) for any controller that can read a windows or MSDOS format text file.

Block Numbering

If you wish to add line numbers to the output file, the current line number is added with the variable [N]

LINE_NUMBER_START = 0	*Value at which the line numbering should start.*
LINE_NUMBER_INCREMENT = 10	*Incremental value between line numbers*
LINE_NUMBER_MAXIMUM = 999999	*Maximum line number to output, before cycling to the LINE_NUMBER_START value again.*
	Important - some controllers have a limit to the number of lines that can be displayed on the control, EG. 65,536

Variables

Below is a table of the variables that are available for use within the configuration files.

Variable Name	Output using	Value	Example in file:
FEED_RATE	[F]	Current Feed Rate.	Mach2_3_ATC_Arcs_inch.pp
CUT_RATE	[FC]	Current Cut Feed Rate.	CNCShark-USB_Arcs_inch.pp
PLUNGE_RATE	[FP]	Current Plunge Feed Rate.	CNCShark-USB_Arcs_inch.pp
SPINDLE_SPEED	[S]	Current Spindle Speed in R.P.M.	GCode_arc_inch.pp
TOOL_NUMBER	[T]	Current Tool Number.	Mach2_3_ATC_Arcs_inch.pp
PREVIOUS_TOOL_NUMBER	[TP]	Previous Tool Number.	NC-Easy.pp
LINE_NUMBER	[N]	Current Line Number in file.	Mach2_3_ATC_Arcs_inch.pp
TOOL_NAME	[TOOLNAME]	Name of Current Tool.	MaxNC_inch.pp
TOOLPATH_NAME	[TOOLPATH_NAME]	Name of Current Toolpath.	Viccam_ATC_Arcs_inch.pp
TOOLPATH_FILENAME	[TP_FILENAME]	Filename (Produced by "Save Toolpath(s)").	ez-Router_inch.pp
TOOLPATH_DIR	[TP_DIR]	Folder Toolpath File was saved to.	Woodp_arc_mm.pp
TOOLPATH_EXTENSION	[TP_EXT]	Toolpath File Extension.	TekcelE_Arc_ATC_3D.pp
TOOLPATH_PATHNAME	[PATHNAME]	Toolpath Folder Pathname.	WinPC-NC_ATC_Arcs_mm.pp
X_POSITION	[X]	Current coordinate of tool position in X axis.	GCode_arc_inch.pp
Y_POSITION	[Y]	Current coordinate of tool position in Y axis.	GCode_arc_inch.pp
Z_POSITION	[Z]	Current coordinate of tool position in Z axis.	GCode_arc_inch.pp
ARC_CENTRE_I_INC_POSITION	[I]	Arc centre in X Axis (relative to last X,Y position).	Mach2_3_ATC_Arcs_inch.pp
ARC_CENTRE_J_INC_POSITION	[J]	Arc centre in Y Axis (relative to last X,Y position).	Mach2_3_ATC_Arcs_inch.pp
ARC_CENTRE_I_ABS_POSITION	[IA]	Arc centre in X Axis (absolute coordinates).	Isel_arc_mm.pp
ARC_CENTRE_J_ABS_POSITION	[JA]	Arc centre in Y Axis (absolute coordinates).	Isel_arc_mm.pp
ARC_START_X_POSITION	[ArcStartX]	Start position of an arc in X axis.	TextOutput_Arcs_mm.pp
ARC_START_Y_POSITION	[ArcStartY]	Start position of an arc in Y axis.	TextOutput_Arcs_mm.pp
ARC_MID_X_POSITION	[ArcMidX]	Mid-point of arc in X (absolute coordinates).	TextOutput_Arcs_mm.pp
ARC_MID_Y_POSITION	[ArcMidY]	Mid-point of arc in Y (absolute coordinates).	TextOutput_Arcs_mm.pp
ARC_MID_X_INC_POSITION	[ArcMidXI]	Mid-point of arc in X (incremental coordinates).	TextOutput_Arcs_mm.pp
ARC_MID_Y_INC_POSITION	[ArcMidYI]	Mid-point of arc in Y (incremental coordinates).	TextOutput_Arcs_mm.pp
ARC_RADIUS	[Radius]	The radius of an arc.	Bosch_ATC_Arcs_mm.pp
ARC_ANGLE	[Angle]		Generic HPGL_Arcs.pp
X_HOME_POSITION	[XH]	Home tool position for X axis.	CAMTech_CMC3_mm.pp
Y_HOME_POSITION	[YH]	Home tool position for Y axis.	CAMTech_CMC3_mm.pp
Z_HOME_POSITION	[ZH]	Home tool position for Z axis.	CAMTech_CMC3_mm.pp
SAFE_Z_HEIGHT	[SAFEZ]	Safe Z Height / Rapid Clearance Gap.	EMC2 Arcs(inch)(*.ngc)
WRAP_DIAMETER	[WRAP_DIA]	Diameter of cylinder that axis is wrapped around.	*Mach2_3-WY_ATC_Arcs_mm.pp*
X_LENGTH	[XLENGTH]	Length of material in X.	Mach2_3_ATC_Arcs_inch.pp
Y_LENGTH	[YLENGTH]	Length of material in Y.	Mach2_3_ATC_Arcs_inch.pp
Z_LENGTH	[ZLENGTH]	Length of material in Z.	Mach2_3_ATC_Arcs_inch.pp
X_MIN	[XMIN]	Minimum value of material in X.	MaxNC_inch.pp
Y_MIN	[YMIN]	Minimum value of material in Y.	MaxNC_inch.pp
Z_MIN	[ZMIN]	Minimum value of material in Z.	MaxNC_inch.pp
X_MAX	[XMAX]	Maximum value of material in X.	MaxNC_inch.pp
Y_MAX	[YMAX]	Maximum value of material in Y.	MaxNC_inch.pp
Z_MAX	[ZMAX]	Maximum value of material in Z.	MaxNC_inch.pp

Table of variables – continued.

Variable Name	Output using	Value	Example in file:
X_ORIGIN_POS	[X_ORIGIN_POS]	Origin Position in X.	TextOutput_Arcs_mm.pp
Y_ORIGIN_POS	[Y_ORIGIN_POS]	Origin Position in Y.	TextOutput_Arcs_mm.pp
Z_ORIGIN	[Z_ORIGIN]	Z Zero Position, Table or Material Surface.	TextOutput_Arcs_mm.pp
XY_ORIGIN	[XY_ORIGIN]	X, Y Origin.	TextOutput_Arcs_mm.pp
TOOLS_USED	[TOOLS_USED]	List of tools used (In order of use).	Mach2_3_ATC_Arcs_inch.pp
TOOLPATHS_OUTPUT	[TOOLPATHS_OUTPUT]	List of toolpaths used in file (in order of use).	Mach2_3_ATC_Arcs_inch.pp
TOOLPATH_NOTES	[TOOLPATH_NOTES]	Toolpath Notes (Toolpath Control form).	Mach2_3_ATC_Arcs_inch.pp
FILE_NOTES	[FILE_NOTES]	File Notes (Edit > Notes).	Mach2_3_ATC_Arcs_inch.pp
TIME	[TIME]	File creation time.	Mach2_3_ATC_Arcs_inch.pp
DATE	[DATE]	File creation date.	Mach2_3_ATC_Arcs_inch.pp

Format of Variables

Values for tool position, feed rates, spindle speeds etc. are inserted into the file using variables.

Variables are used throughout the file; the variables are replaced with the current value for that item when the file is post processed.

For example, the current X, Y and Z tool positions at any time, are inserted into the file by using the variable output, [X], [Y] and [Z] respectively.

Write variable names in upper case letters for clarity.

A variable is formatted as follows:

VAR VARIABLE = [VO|WO|CS|VF|MX]

Key:

VO = Variable Output for example, X, XH, F.

WO= When output, A=Always | C= Only when Changed.

CS= Character String output before value.

VF= Value Format, determines the format that the value is output with.

MX= Multiplier Value – See Page 19 for details.

A Typical Variable

1	2	3	4	5	6	7	8	9	10	11	12	13	14	14
VAR	Z_HOME_POSITION	=	[	ZH	|	A	|	Z	|	F	1.0	|	-1	]

1 VAR - This line is a Variable.
2 Variable Name.
3 Equals Sign.
4 Open Square bracket - (start of variable formatting parameters).
5 Variable label - I.E. label that is substituted with the variable value.
6 Vertical Bar - Parameter separator.
7 A= Always output value, C= Only output value when it changes.
8 Vertical Bar - Parameter separator.
9 Character string to print before variable value
10 Vertical Bar - Parameter separator.
11 Optional Format Flag(s) – See Page 13 for details.
12 Value Format - units and number of decimal places to output
13 Vertical Bar - Parameter separator.
14 Output multiplier – See Page 19 for details.
15 Close Square Bracket – End of formatting parameters.

Variables consist of a Text String enclosed within square brackets e.g. [T]

Formatting the Output Value

The values of the variable are formatted as follows:

{Format Flags} {Field Width} {Decimal Separator} {Decimal Places}

The format flags are optional and only needed by a small number of controllers they will be described shortly.

Field Width

The Field width represents the minimum number of characters that are output. The field width is usually set to "1" a value greater than 1 is typically only required if a controller expects to see a fixed number of characters for the value. If this is the case, a number greater than 1 can be entered. The number entered will ensure that that number of characters is output. The number that represents the field width includes the full floating-point number for the output value, (including the decimal separator character).

Decimal Separator

The decimal separator character is almost always just a period character, but there are some controllers that expect to see a comma character. (For an example of a post processor that does not use a period character, see the file: Heidenhain_inch.pp)

Decimal Places

The number of decimal places output following the decimal separator. The values are often set at 3 for controllers operating in Metric, or 4 for controllers operating in Inches.

Optional Format Flags

The output values can be further modified by using the optional format flags.

Format Flags

Flag	Function	Default (without flag)
-	Left justify the output value.	Values are right justified.
+	Prefix the output value with a '+' or '-'	Only negative values are prefixed.
0	If value has fewer characters than the set minimum, the value is prefixed with zeros.	Value is prefixed with blank spaces.
#	Value is always output with a separator character (In practice this would only change the output value if the value is set to output integer values only).	When output is set to integer only, separator character is not appended to value.

Default Formatting for Variables

Most variables have a default format; (shown below) to set a different format for a variable, enter the line below in your post processor and alter the parameters to suit your controller.

VAR LINE_NUMBER = [N|A||1.0]

Example: VAR LINE_NUMBER = [N|A|N|1.0]

The line number will Always be output. An 'N' character will be inserted before the line number and it will be output as an integer number

VAR SPINDLE_SPEED = [S|A||1.0]

Example: VAR SPINDLE_SPEED = [S|A|S|1.0]

The speed will Always be output. An 'S' character will be inserted before the value and it will be output as an integer number

VAR FEED_RATE = [F|A||1.0]

Example: VAR FEED _RATE= [F|C|F|1.1|0.01666]

Note: In this format string there is an optional extra parameter, this is a value multiplier. See the note on page 19 for more details.

The feed rate will be output with an 'F' character before the value, and will only be output when it Changes. The value will be output to 1 decimal place.

VAR PLUNGE_RATE = [FP|A||1.0]

Example: VAR PLUNGE _RATE= [FP|C|F|1.1|0.01666]

Note: In this format string there is an optional extra parameter, this is a value multiplier. See the note on page 19 for more details.

The feed rate will be output with an 'F' character before the value, and will only be output when it Changes. The value will be output to 1 decimal place.

VAR CUT_RATE = [FC|C ||1.0]

Example: VAR CUT_RATE= [FC|C|F|1.1|0.01666]

Note: In this format string there is an optional extra parameter, this is a value multiplier. See the note on page 19 for more details.

The feed rate will be output with an 'F' character before the value, and will only be output when it Changes. The value will be output to 1 decimal place.

+ Tool position in x,y and z - Defaults

VAR X_POSITION = [X|A| |1.3]
VAR Y_POSITION = [Y|A| |1.3]
VAR Z_POSITION = [Z|A| |1.3]

Example: VAR X_POSITION = [X|A|X|1.3]

The position value will be output with an 'X' character before the value, the position will Always be output, and will be output to 3 decimal places, this would typically be suitable for a control that requires metric output. If you wished to output the values to 4 decimal places as would be more typical for a controller operating in inches. You would format the line as follows.
VAR X_POSITION = [X|A|X|1.4]

+ Home tool positions - Defaults

VAR X_HOME_POSITION = [XH|A||1.3]
VAR Y_HOME_POSITION = [YH|A||1.3]
VAR Z_HOME_POSITION = [ZH|A||1.3]

Example: VAR Y_HOME_POSITION = [YH|A|Y|1.3]

The position value will be output with a 'Y' character before the value, the position will Always be output, and will be output to 3 decimal places, this would typically be suitable for a control that requires metric output. If you wished to output the values to 4 decimal places as would be more typical for a controller operating in inches. You would format the line as follows.
VAR Y_HOME_POSITION = [YH|A|Y|1.4]

+ Safe Z (Rapid clearance) height - Defaults

VAR SAFE_Z_HEIGHT = [SAFEZ|A||1.3]

Example: VAR SAFE_Z_HEIGHT = [SAFEZ |A|Z|1.3|-1]

The value will be output with a 'Z' character before it, the value will Always be output, and will be output to 3 decimal places, this would typically be suitable for a control that requires metric output. If you wished to output the values to 4 decimal places as would be more typical for a controller operating in inches. You would format the line as follows.

VAR SAFE_Z_HEIGHT = [SAFEZ |A|Z|1.4|- 1]

Note: In this format string there is an optional extra parameter, this is a value multiplier. See the note on page 19 for more details.

+ Arc Output Variables - Defaults

The following methods of defining arcs are available, the appropriate definition should be chosen to match that used by your controller.

Check your control documentation for details.

VAR ARC_START_X_POSITION = [ArcStartX|A| |1.3]

VAR ARC_START_Y_POSITION = [ArcStartY|A| |1.3]

Example: VAR ARC_START_Y_POSITION = [ArcStartY|A|Y|1.3]

The value will be output with a 'Y' character before it, the value will Always be output, and will be output to 3 decimal places, this would typically be suitable for a control that requires metric output. If you wished to output the values to 4 decimal places as would be more typical for a controller operating in inches. You would format the line as follows.

VAR ARC_START_Y_POSITION = [ArcStartY|A|Y|1.4]

VAR ARC_CENTRE_I_INC_POSITION = [I|A| |1.3]
VAR ARC_CENTRE_J_INC_POSITION = [J|A| |1.3]

Example: VAR ARC_CENTRE_J_INC_POSITION = [J|A|J|1.3]

The value will be output with a 'J' character before it, the value will Always be output, and will be output to 3 decimal places, this would typically be suitable for a control that requires metric output. If you wished to output the values to 4 decimal places as would be more typical for a controller operating in inches. You would format the line as follows:

VAR ARC_CENTRE_J_INC_POSITION = [J|A|J|1.4]

VAR ARC_CENTRE_I_ABS_POSITION = [IA|A| |1.3]
VAR ARC_CENTRE_J_ABS_POSITION = [JA|A| |1.3]

Example: VAR ARC_CENTRE_I_ABS_POSITION = [IA|A|I|1.3|-1]

Note: In this format string there is an optional extra parameter, this is a value multiplier. See the note on page 19 for more details.

The value will be output with an 'I' character before it, the value will Always be output, and will be output to 3 decimal places, this would typically be suitable for a control that requires metric output. If you wished to output the values to 4 decimal places as would be more typical for a controller operating in inches. You would format the line as follows:

VAR ARC_CENTRE_I_ABS_POSITION = [IA|A|I|1.4|- 1]

VAR ARC_MID_X_POSITION = [ArcMidX|A| |1.3]
VAR ARC_MID_Y_POSITION = [ArcMidY|A| |1.3]

Example: VAR ARC_ MID_X_POSITION = [ArcMidX|A|X|1.3|-1]

The value will be output with an 'X' character before it, the value will Always be output, and will be output to 3 decimal places, this would typically be suitable for a control that requires metric output. If you wished to output the values to 4 decimal places as would be more typical for a controller operating in inches. You would format the line as follows:

VAR ARC_ MID_X_POSITION = [ArcMidX|A|X|1.4|- 1]

VAR ARC_MID_X_INC_POSITION = [ArcMidXI|A| |1.3]
VAR ARC_MID_Y_INC_POSITION = [ArcMidYI|A| |1.3]

Example: VAR ARC_MID_Y_INC_POSITION = [ArcMidYI|A|Y|1.3|-1]

Note: In this format string there is an optional extra parameter, this is a value multiplier. See the note on page 19 for more details.

The value will be output with a 'Y' character before it, the value will Always be output, and will be output to 3 decimal places, this would typically be suitable for a control that requires metric output. If you wished to output the values to 4 decimal places as would be more typical for a controller operating in inches. You would format the line as follows:

VAR ARC_MID_Y_INC_POSITION = [ArcMidYI|A|Y|1.4|- 1]

VAR ARC_RADIUS = [Radius|A| |1.3]

Example: VAR ARC_RADIUS = [Radius|A|R|1.3]

The value will be output with an 'R' character before it, the value will Always be output, and will be output to 3 decimal places, this would typically be suitable for a control that requires metric output. If you wished to output the values to 4 decimal places as would be more typical for a controller operating in inches. You would format the line as follows:

VAR ARC_RADIUS = [Radius|A|R|1.4]

VAR ARC_ANGLE = [Angle|A| |1.3]

Example: VAR ARC_ANGLE = [Angle|A||1.3|40]

Note: In this format string there is an optional extra parameter, this is a value multiplier. See the note on page 19 for more details.

The value will be output with no characters before it, the value will Always be output, and will be output to 3 decimal places.

Material Block information – Defaults

VAR X_LENGTH = [XLENGTH|A| |1.3]
VAR Y_LENGTH = [YLENGTH|A| |1.3]
VAR Z_LENGTH = [ZLENGTH|A| |1.3]

VAR X_MIN = [XMIN|A| |1.3]
VAR Y_MIN = [YMIN|A| |1.3]
VAR Z_MIN = [ZMIN|A| |1.3]

VAR X_MAX = [XMAX|A| |1.3]
VAR Y_MAX = [YMAX|A| |1.3]
VAR Z_MAX = [ZMAX|A| |1.3]

VAR X_MIN = [XMIN|A| |1.3]

Example: VAR X_MIN = [XMIN|A|X|1.3]

The value will be output with an X character before it, the value will Always be output, and will be output to 3 decimal places.

Multiplier Value

The multiplier value is used to multiply the value to output a different value.

Common reasons for wishing to do this are:

To convert the default output of an Inch post processor, from inches per minute to inches per second, (Multiply by 0.01666).

To convert the default output of a Metric post processor, from mm per minute to mm per second, (Multiply by 0.0166).

To make positive values negative (and vice versa), (Multiply by -1).

To convert the output of an arc angle from radians to degrees, (Multiply by 57.2957795).

To multiply or divide by a fixed factor (I.E. produce 1:4 scale model, Multiply by 0.25)

Post Processor Blocks

HEADER

+---
+ Commands output at the start of the file
+---
begin HEADER
"Commands"

The header is the location for the instructions that are output once,
at the start of the file, these generally setup modal commands for the controller.

For example, the Header might contain a command to display the filename on the controller and a series of "G-Codes" to set the machine up, for instance G20 to tell the control that the moves are in inches, or G21 to tell the control that the moves are in millimetres.

Variables that you might wish to be within the header section,
could include:

Information about the Material Block
Minimum extent in X = [XMIN] Minimum extent in Y = [YMIN] Minimum extent in Z = [ZMIN]
Maximum extent in X = [XMAX] Maximum extent in Y = [YMAX] Maximum extent in Z = [ZMAX]
Length of material in X = [XLENGTH]"
Length of material in Y = [YLENGTH]"
Depth of material in Z = [ZLENGTH]"

Home Position Information

Home X = [XH] Home Y = [YH] Home Z = [ZH]

Rapid clearance gap or Safe Z = [SAFEZ]

Details of the first tool to be used.

Tool Number = [T]"
Tool name = [TOOLNAME]

Initial cutting speeds
Feed Rate used for cutting and plunging into the material = [F]
Feed Rate whilst tool is Cutting the material = [FC]
Feed Rate whilst tool is plunging into the material = [FP]
Actual values depend on the UNITS set (see Global File Settings)
Defaults are either MM/Minute or Inches/Minute, but the output can be changed to suit by setting the appropriate "VAR FEED_RATE" formatting.

Spindle Speed
Spindle Speed = [S] *R.P.M.*

TOOLCHANGE

+--
+ Commands output at toolchange
+--
begin TOOLCHANGE
"Commands"
Commands that are output when a change of tool is required.
Variables and commands that might be used include:
Previous Tool Number = [TP]
Tool Number = [T]
Tool name = [TOOLNAME]
Toolpath Name = [TOOLPATH_NAME]
Toolpath Pathname = [PATHNAME]
Toolpath File Name = [TP_FILENAME]
Toolpath File Directory = [TP_DIR]
Toolpath Extension = [TP_EXT]
Spindle Speed = [S] *R.P.M.*
M3 *M Code often used to turn spindle on (Clockwise rotation).*
M5 *M Code often used to turn spindle off.*

NEW_SEGMENT

+--
+ Commands output for a new segment (new toolpath with current toolnumber)
+--
begin NEW_SEGMENT
"Commands"
For an example of a NEW_SEGEMENT section, see the file: Mach2_3_ATC_Arcs_inch.pp

Commands that are output when a new toolpath uses the currently selected tool, but perhaps a different spindle speed is required or the machine requires additional instructions. Any commands that are used in the NEW_SEGMENT section should not need to be included within the TOOLCHANGE section as a tool-change will also automatically call the instructions in the NEW_SEGMENT section.
Variables that are commonly used include.
Spindle Speed = [S] *R.P.M.*
M3 *M Code often used to turn spindle on (Clockwise rotation).*
M5 *M Code often used to turn spindle off.*

INITIAL_RAPID_MOVE

+ Commands output for Initial rapid move

begin INITIAL_RAPID_MOVE
"Commands"
For an example of a INITIAL_RAPID_MOVE section, see the file: Saom_OSAI_Arc_inch.pp

Commands that are output when the very first rapid move is made. A Section not used for most posts, but useful if the very first rapid move, needs to output different information to subsequent rapid moves.
This section is sometimes required for HPGL variants.

RAPID_MOVE

+--
+ Commands output for rapid moves
+--
begin RAPID_MOVE
“Commands”

Commands that are output when rapid moves are required.

FIRST_FEED_MOVE

+--
+ Commands output for first feed rate moves.
+--
begin FIRST_FEED_MOVE
“Commands”
This section is commonly used where controllers require that the Feed Rate is set at the first feed move, this rate would then be used for subsequent cut moves.

For an example of a FIRST_FEED_MOVE section, see the file: Axyz_Arcs_ATC_inch.pp

FEED_MOVE

+--
+ Commands output for feed rate moves
+--
begin FEED_MOVE
“Commands”
Used to output information required at every move, or all feed moves except for the First Feed Move, if a FIRST_FEED_MOVE section is present within the post processor.
.

FIRST_CW_ARC_MOVE

+--
+ Commands output for the first clockwise arc move
+--
begin FIRST_CW_ARC_MOVE
“Commands”
Similar to the FIRST_FEED_MOVE section, but for clockwise arc segments.
This section is commonly used where controllers require that the Feed Rate is set for the first arc segment, this rate would then be used for subsequent arc moves in the same direction.

For an example of a FIRST_CW_ARC_MOVE section, see the file: Centroid_Arcs_inch.pp

CW_ARC_MOVE

+----------
+ Commands output for clockwise arc moves.
+----------
begin CW_ARC_MOVE
"Commands"
Similar to the FEED_MOVE section, but for clockwise arc segments.

For an example of a CW_ARC_MOVE section, see the file: Centroid_Arcs_inch.pp

FIRST_CCW_ARC_MOVE

+----------
+ Commands output for the first counter-clockwise arc move
+----------
begin FIRST_CCW_ARC_MOVE
"Commands"
Similar to the FIRST_FEED_MOVE section, but for counter-clockwise arc segments.
This section is commonly used where controllers require that the Feed Rate is set for the first arc segment, this rate would then be used for subsequent arc moves in the same direction.

For an example of a FIRST_CCW_ARC_MOVE section, see the file: Centroid_Arcs_inch.pp

CCW_ARC_MOVE

+----------
+ Commands output for counter-clockwise arc moves.
+----------
begin CCW_ARC_MOVE
"Commands"
Similar to the FEED_MOVE section, but for counter-clockwise arc segments.

For an example of a CCW_ARC_MOVE section, see the file: Centroid_Arcs_inch.pp

FOOTER

The footer is the section of the post processor for instructions that are sent to the controller at the end of a file. These might be instructions to return the tool to the home position, switch the spindle off or switch the power off to the drives.

```
+-------------------------------------------------
+   Commands output at the end of the file
+-------------------------------------------------
begin FOOTER
"Commands"
Variables that are commonly used include.
G00  [XH]  [YH]  [ZH]      Rapid to X,Y,Z Home position.
M05      M Code often used to turn spindle off.
M30      M Code often used to signify the end of the file.
```

Other less frequently used sections.

FEED_RATE_CHANGE

+--
+ Commands output when feed rate changes
+--
begin FEED_RATE_CHANGE
"Commands"
For an example of a FEED_RATE_CHANGE section, see the file: Gravograph_IS200.pp

Commands that are output when the feed rate is changed.
This section is not often used as many controllers will accept feed rate changes appended to other instructions, but sometimes used with HPGL variants.

FIRST_PLUNGE_MOVE

+--
+ Commands output for the First Plunge Move
+--
begin FIRST_PLUNGE_MOVE
"Commands"
For an example of a FIRST_PLUNGE_MOVE section, see the file: Holz-Her_7123_ATC_Arcs_mm.pp

This section is often used on machines that do not fully support simultaneous 3D movement, for example the Z Axis cannot travel as fast as the X & Y Axis.
Another use of this section might be to include commands that you wish to output whenever the first plunge move occurs.
For example, commands to switch on a plasma torch.
Multiple plunges would normally only be output within a ramping move, so this command would be useful for controls that automatically rapid between cuts and where instructions such as revised speeds and feed need to be specified on the first plunge move and these instructions are not required for subsequent plunge moves within the ramping operation.

PLUNGE_MOVE

+--
+ Commands output for Plunge Moves
+--
begin PLUNGE_MOVE
"Commands"
For an example of a PLUNGE_MOVE section, see the file: Burny_arc_inch.pp

This section is often used on machines that do not fully support simultaneous 3D movement, for example the Z Axis cannot travel as fast as the X & Y Axis.

Another use of this section might be to include commands that you wish to output whenever a plunge move occurs.
For example, commands to switch on a plasma torch.

RETRACT_MOVE

+----------
+ Commands output for Retract Moves
+----------
begin RETRACT_MOVE
“Commands”
For an example of a RETRACT _MOVE section, see the file: Burny_arc_inch.pp

A use of this section might be to include commands to switch off a plasma torch.

Special Characters

Most characters can be output within the confines of the post processor output statements; however, certain characters have special meaning within the post processor configuration files and cannot be output directly.
These are, the Square brackets [ ], and the double quote character "
It may be the case that you need to output one of these characters within your output file.

If you wish to output one of these characters, you can do so by enclosing the decimal equivalent of the ASCII value of the special character that you wish to output, within square brackets, as shown below. This method can also be used to insert any ASCII value, even non-printable characters.

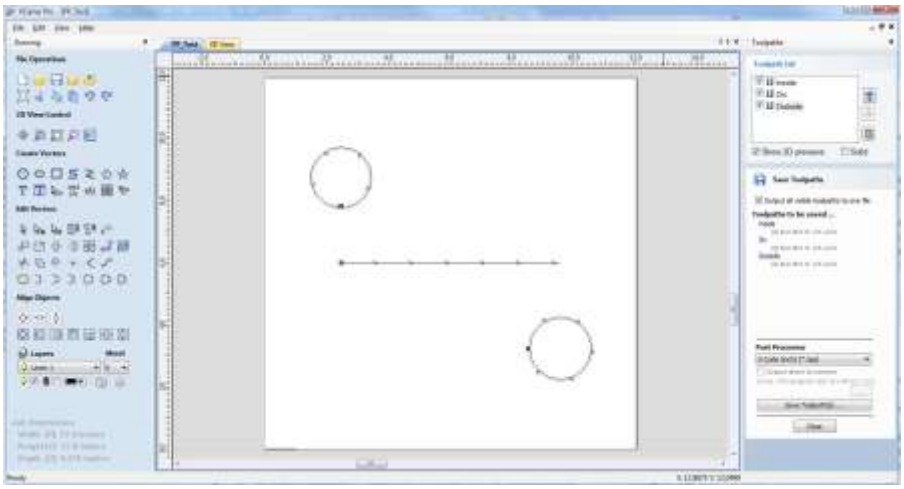
[91] *Outputs a left hand square bracket.*
[93] *Outputs a right hand square bracket.*
[34] *Outputs a double quote character.*

[13] *Outputs a carriage return.*
[10] *Outputs a line feed.*

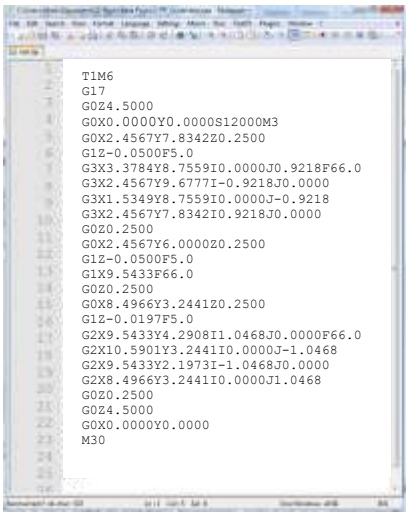
For an example of a file that uses special characters, please see: Biesse_Rover_Arcs_mm.pp

Editing a Vectric Post Processor – An example: Adding tool-change commands.

For the majority of cases, the quickest and easiest way of producing a customised post processor to suit your controller, will be to edit an existing post processor.
To do this, first create a simple test file that you can use to test the output of your post processor. A simple file might consist of a line, and two circles. Produce a shallow cutting profile toolpaths for each of the shapes, machining “On” the line, “Inside” one of the circles and “Outside” the other circle.

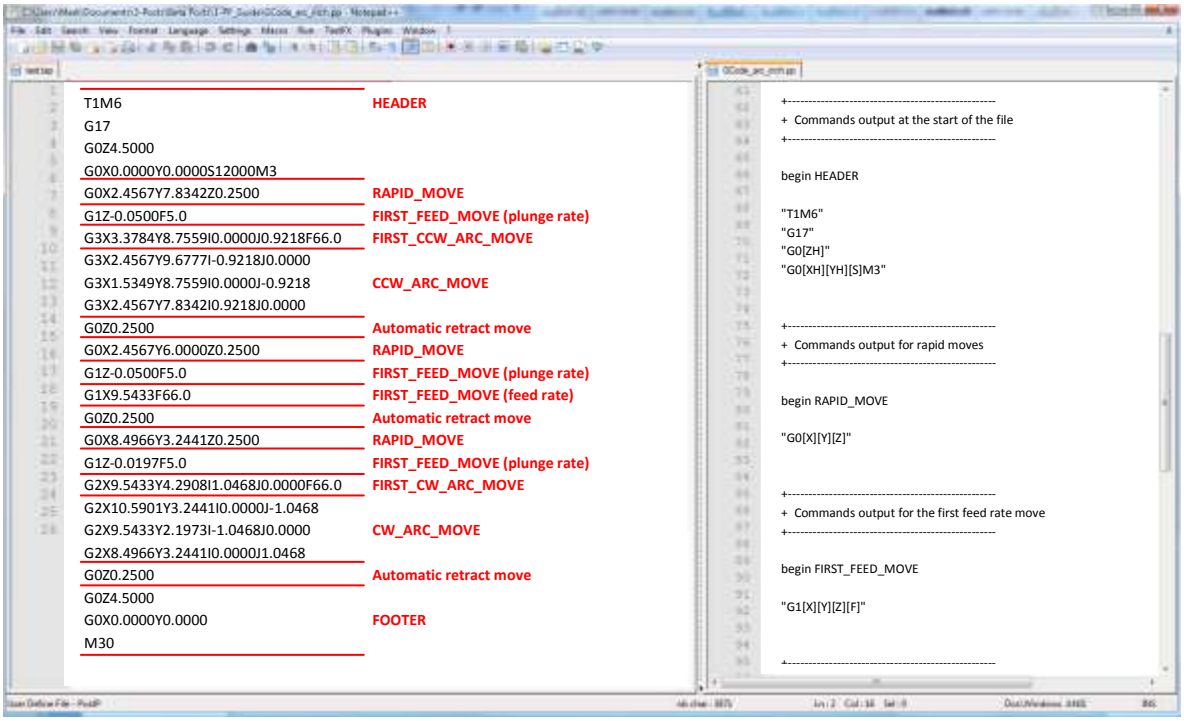


Save a toolpath using your base post processor and take a look at it using your favourite text editor. Below is an example of the test file posted using the "G-Code Arcs (inch) (*.tap)" post processor The example below is displayed using the popular Notepad ++ editor.



```
T1M6
G17
G0Z4.5000
G0X0.0000Y0.0000S12000M3
G0X2.4567Y7.8342Z0.2500
G1Z-0.0500F5.0
G3X3.3784Y8.7559I0.0000J0.9218F66.0
G3X2.4567Y9.6777I-0.9218J0.0000
G3X1.5349Y8.7559I0.0000J-0.9218
G3X2.4567Y7.8342I0.9218J0.0000
G0Z0.2500
G0X2.4567Y6.0000Z0.2500
G1Z-0.0500F5.0
G1X9.5433F66.0
G0Z0.2500
G0X8.4966Y3.2441Z0.2500
G1Z-0.0197F5.0
G2X9.5433Y4.2908I1.0468J0.0000F66.0
G2X10.5901Y3.2441I0.0000J-1.0468
G2X9.5433Y2.1973I-1.0468J0.0000
G2X8.4966Y3.2441I0.0000J1.0468
G0Z0.2500
G0Z4.5000
G0X0.0000Y0.0000
M30
```

For our example, we will add a tool-change section to this post processor.
First, make a safe copy of the post processor that you are customising.
If you open the post processor that you are editing in a text editor, you will be able to see the lines of text within the post processor that formatted the output of your test file.



The post processor configuration files are located in the “PostP” folder for the application. The post processor configuration files have a “.PP” file extension.
The PostP folder can be reached from within the application, by clicking
“File > Open Application Data Folder” from the main menu of the application.

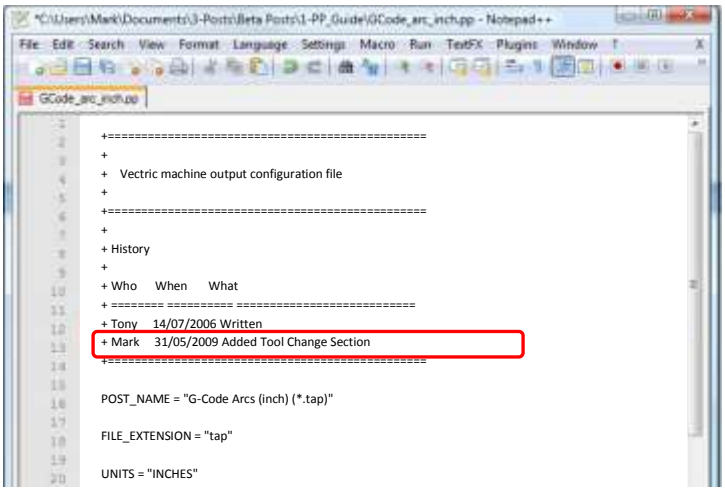
To add a Tool Change section to the post processor, you will need to consult the documentation for the control of the machine tool (or control software).
For this example, we will assume that the instructions that you need to add to perform a tool change for your particular machine tool are as follows:

M05	*Instruction to turn off the spindle prior to tool change.*
M0	*Instruction to return existing tool to tool holder.*
M06T*Tool_Number n*	*Instruction to select new tool Tool_Number n*
G43H*Tool_Number n*	*Instruction for control to use Tool length offset for tool n*
S*xxx* M03	*Set spindle speed to xxx; Turn on spindle (clockwise rotation).*

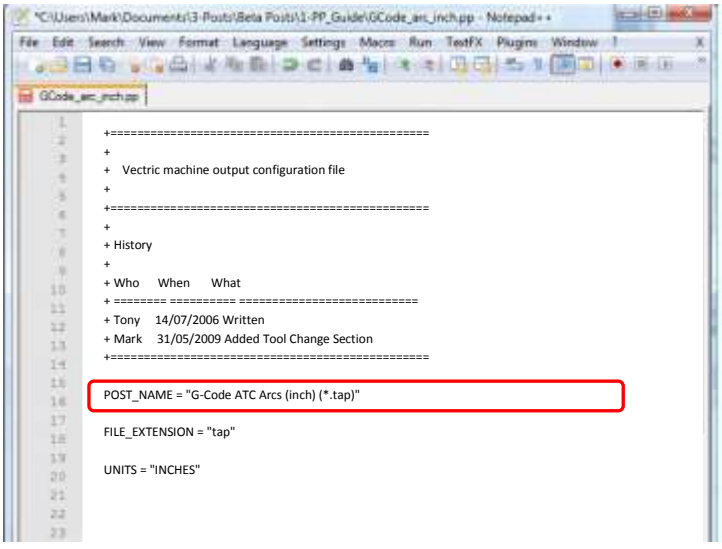
Edit the post processor using your favourite text editor.

If the operating system on your computer is Microsoft Vista and User Access Control is enabled, copy or move the Post Processor that you are editing from the PostP folder to a folder below your user area.

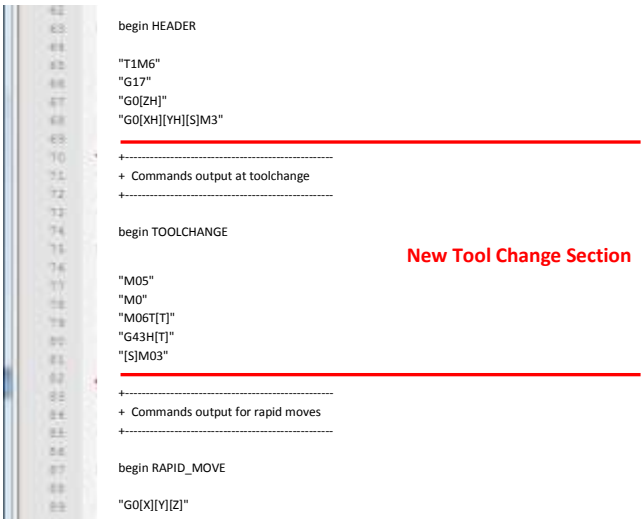
The first thing that you should edit within the file is the History Comment section; So that you have a record of the changes.



Next edit the POST_NAME to reflect that this post processor outputs automatic tool change (ATC) commands, the new post will be displayed as “G-Code ATC Arcs (inch)(*.tap)” in the list of post processors.



Next add a Tool Change Section that will include the instructions.
The location of the new section within the file is not important, but a good place to insert it is between the Header and Rapid Move sections.



Add some comment lines at the top of the new section, (beginning with the + character) to describe the section and make the file as a whole, easier to read.
Next enter the line “begin TOOLCHANGE” to instruct the post processor that the following instructions are to be output for every tool-change, (except the initial tool selection, the commands for these are contained within the header section).

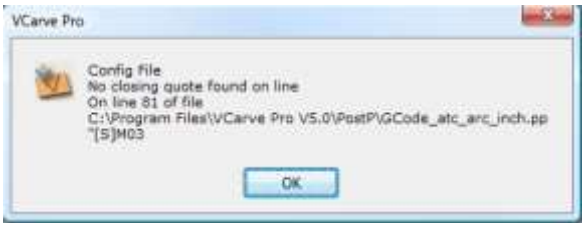
The next step is to enter in the instructions that you require, enclosed within double quote marks. The “[T]” on the third and fourth instruction lines of our example, will be substituted with the tool number when the file is post processed; The “[S]” on the fifth line will be substituted with the spindle speed for the tool.

Finally you will need to save the changes to the file, as you have changed the POST_NAME, save the file using a new name, for example “GCODE_ATC_Arcs_inch.pp”

If the operating system on your computer is Microsoft Windows 7 or Microsoft Vista and User Access Control is enabled, copy the file that you have edited back to the “PostP” folder.

To test the new post processor, If the software is running, restart the software.

If there are any syntax errors with your post processor, an error similar to the picture below will be displayed as the software starts, the post processor that you have edited will not appear in the drop down list of post processor configuration files.
You will need to rectify any errors and restart the software.



If there are no errors displayed when the software is started, open your test file
and save one or more of your test toolpaths.

Select the post processor from the drop down list of post processor configuration and press the "Save Toolpath(s)" button.

Take a look at the file that you have just saved in a text editor.

If the content of the file looks good, try the file on your machine.

Please take all necessary precautions when running the output from a modified post processor for the first time.

Editing a Vectric Post Processor – An example: Changing the File Extension.

The File extension that is automatically produced by the post processor can be changed within the “Save As” dialog box, when you click on the “Save Toolpath(s)” button.

However, rather than change the file extension every time. It is more convenient to permanently change the file extension produced by the post processor.

To do this:

Make a safe copy of the post processor that you wish to edit.

The post processor configuration files are located in the “PostP” folder of the product installation folder and have a “.PP” file extension.
For Aspire version 2, the default location of the PostP folder is:
“C:\Program Files\Aspire V2.0\PostP” the location of the PostP folder will be different depending on the product installed and whether the software was installed to a custom location when the software was installed.

Edit the post processor using your favourite text editor.

If the operating system on your computer is Microsoft Windows 7 or Microsoft Vista and User Access Control is enabled,
copy or move the Post Processor that you are editing from the PostP folder to a folder below your user area.

Look for the following two lines within the post processor configuration file that begin with:

POST_NAME =

FILE_EXTENSION =

and alter these accordingly.

For example, if you wished to change the file extension produced by the “G Code ATC (inch)(*.tap)” post processor from “.tap” to “.nc”

Edit the lines:

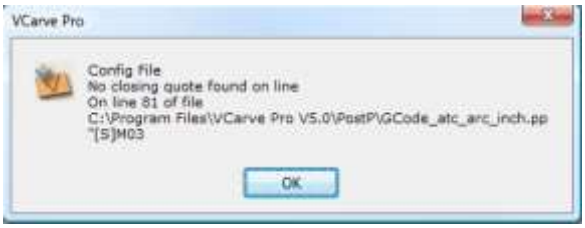
From	To
POST_NAME = "G Code ATC (inch) (*.tap)" FILE_EXTENSION = "tap"	POST_NAME = "G Code ATC (inch) (*.nc)" FILE_EXTENSION = "nc"

Save the changes to your file.

If the operating system on your computer is Microsoft Windows 7 or Microsoft Vista and User Access Control is enabled, copy the file that you have edited back to the "PostP" folder.

To test the new post processor, If the software is running, restart the software.
If there are any syntax errors with your post processor, an error similar to the picture below will be

displayed as the software starts, the post processor that you have edited will not appear in the drop down list of post processor configuration files.
You will need to rectify any errors and restart the software.



If there are no errors displayed when the software is started, open your test file
and save one or more of your test toolpaths.

Select the post processor from the drop down list of post processor configuration and press the "Save Toolpath(s)" button.

Take a look at the file that you have just saved in a text editor.

If the content of the file looks good, try the file on your machine.

Please take all necessary precautions when running the output from a modified post processor for the first time.

Post Processor differences between different Vectric Products.

Not all of the Post Processor sections and variables are supported in all Vectric products.
For the release versions of Vectric products as of October 1st 2009, the following differences apply.

Cut3D (Release 1.025) – Unsupported Commands and Variables

DIRECT_OUTPUT
SUBSTITUTE
TAPE_SPLITTING
RAPID_PLUNGE_TO_STARTZ
ROTARY_WRAP_Y
ROTARY_WRAP_X
begin FIRST_PLUNGE_MOVE [Section]

Variables not supported.
[WRAP_DIA] [X_ORIGIN_POS] [Y_ORIGIN_POS] [Z_ORIGIN] [XY_ORIGIN] [TOOLS_USED]
[TOOLPATH_NOTES] [FILE_NOTES] [TIME] [DATE] [TOOLPATHS_OUTPUT]

PhotoVCarve (Release 1.102) – Unsupported Commands and Variables

All **ARC** Support commands and variables
DIRECT_OUTPUT
PRINT_DIRECT
SPINDLE_SPEED_RANGE
SUBSTITUTE
TAPE_SPLITTING
RAPID_PLUNGE_TO_STARTZ
ROTARY_WRAP_Y
ROTARY_WRAP_X
begin FIRST_PLUNGE_MOVE [Section]
begin NEW_SEGMENT [Section]

Variables not supported.
[WRAP_DIA] [X_ORIGIN_POS] [Y_ORIGIN_POS] [Z_ORIGIN] [XY_ORIGIN] [TOOLS_USED]
[TOOLPATH_NOTES] [FILE_NOTES] [TIME] [DATE] [TOOLPATHS_OUTPUT]

Cut2D (Release 1.100) – Unsupported Commands and Variables

Tool change commands are accepted but not used by the program.
SUBSTITUTE
TAPE_SPLITTING
RAPID_PLUNGE_TO_STARTZ
ROTARY_WRAP_Y
ROTARY_WRAP_X
begin FIRST_PLUNGE_MOVE [Section]

Variables not supported.
[WRAP_DIA] [X_ORIGIN_POS] [Y_ORIGIN_POS] [Z_ORIGIN] [XY_ORIGIN] [TOOLS_USED]
[TOOLPATH_NOTES] [FILE_NOTES] [TIME] [DATE] [TOOLPATHS_OUTPUT]

Tips and Tricks.

1. Always make a safe copy of the post processor that you are editing, in case you need to start again from scratch.

2. If using a word processor program, such as Microsoft Word, to edit a post processor, make sure that the file is saved as plain text. The file should not contain formatting information.

3. If editing post processors on a computer that runs Microsoft Windows 7 or Microsoft Vista, do not edit the files directly within the "Program Files*Product folder*\PostP" folder. Always edit the file within your user area and copy the edited file to "Program Files*Product folder*\PostP".

4. Use comments when you make changes, a comment is a text that follows a + or a | character. Comments will not be acted upon by the program but can help in documenting changes that you have made and make those changes understandable in the future.

5. All Instruction lines must be contained within quote marks.

6. If possible, use a text editor that makes use of line numbers; This will make it easier to debug the post processor if there are any errors in the file. The program will check the post processors in the PostP folder when the program starts.
   If syntax errors are present in the file, an error message will be displayed, showing the line number of the first error encountered.

7. Once you have successfully edited a post processor, make a safe copy of it.
   If you install a later version of the Vectric product that you are using, remember to copy your modified post processor to the PostP folder of the new version of the software.
   And select your modified post processor, the first time that you save a toolpath,
   (The software will remember your selection for subsequent actions).

8. If you install another version of the software or you upgrade the version of the software, remember to copy your safe copies of your edited post processors to the PostP folder of the new version. Ensure that you select the correct post processor the first time that you post process a file using the new version of the software.

Post Processor Naming Conventions.

Within the post processors that ship with the products,
you will see the following terms or abbreviations used within the
POST_NAME

Arc or Arcs	=	The post processor will output true arcs within the design as Arc moves, (G2,G3,Radius of Arc etc.).
Inc	=	Arcs are output with centre coordinates incremental to the last move.
Abs	=	Arcs are output with absolute coordinate values.
ATC	=	The post processor will output Automatic Tool Change (ATC) commands.
WY	=	Wrapped out output post processor – Y values wrapped around X axis
WX	=	Wrapped out output post processor – X values wrapped around Y axis
TS*number*	=	Tape Splitting post processor. An optional *number* is the maximum number of lines that will be output within a single file. Files that exceed this number of lines will be split into multiple files. (*See* Tape Splitting Support *for more details).*
(mm)	=	The post processor units are Metric.
(inch)	=	The post processor units are English (Imperial) inches.
(*.xyz)	=	The file extension that the output file will be created with.