RosettaCNC G-code language

All rights reserved on this manual. No part of this document can be copied or reproduced in any form without prior written authorisation. RosettaCNC Motion® does not insure or guarantee its contents and explicitly declines all liability related to the guarantee of its suitability for any purpose. The information in this document can be changed without notice. RosettaCNC Motion® shall not be held liable for any error or omission in this document. RosettaCNC Motion® is a registered trademark.

Information

<table>
<thead>
<tr>
<th>Release documento</th>
<th>Descrizione</th>
<th>Note</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>First release</td>
<td></td>
<td>17/01/2018</td>
</tr>
<tr>
<td>02</td>
<td>Minor changes</td>
<td></td>
<td>29/01/2019</td>
</tr>
<tr>
<td>03</td>
<td>Update for 1.6 version</td>
<td></td>
<td>02/09/2019</td>
</tr>
<tr>
<td>04</td>
<td>Update for 1.7 version</td>
<td></td>
<td>29/11/2019</td>
</tr>
<tr>
<td>05</td>
<td>Update for 1.7.6 version</td>
<td></td>
<td>04/02/2020</td>
</tr>
<tr>
<td>06</td>
<td>Update for 1.7.7 version</td>
<td></td>
<td>04/03/2020</td>
</tr>
</tbody>
</table>

References

This manual explains the control software used by the RosettaCNC software.

All the implementation details are refer to the software version: 1.7.7.

The actual control software version can be identified by open the “Help” menu → “About RosettaCNC”.

### Coordinate Systems

- **8.1 The 5 Steps G-Code Coordinate Pipeline**
  - 8.1.1 Step 1: Unit Conversion
  - 8.1.2 Step 2: Conversion from Relative or Polar to Absolute Coordinates
  - 8.1.3 Step 3: Offsets: G52, G54 and G92
  - 8.1.4 Step 4: Scaling and Mirroring: G51
  - 8.1.5 Step 5: Rotation: G68

- **8.2 Cartesian & Polar Coordinates**
  - 8.2.1 Use on different planes
  - 8.2.2 Standard syntax
  - Plane independent syntax
9. Rotary axis options ............................................................... 51
   Example .............................................................................. 51
10. Messages & Media ............................................................... 52
    10.1 Parameters meaning ..................................................... 52
        Notes .............................................................................. 52
    10.2 Supported HTML syntax ............................................. 52
        Bold tag ........................................................................ 52
        Underline tag ................................................................ 52
        Italic tag ....................................................................... 52
        Strikeout tag .................................................................. 53
        Line break ...................................................................... 53
        Subscript/Superscript tags .............................................. 53
        List tags & List items ....................................................... 53
        Text with shadow ............................................................ 53
        Highlight ........................................................................ 53
        Error marking ................................................................ 53
    10.3 Supported Not Standard HTML Tags ......................... 54
        Tool Info ......................................................................... 54
    10.4 Examples .................................................................... 54
        Modal window messages ............................................... 54
        HUD messages ............................................................... 55
        Use piece thickness to update WCS offsets ..................... 55
11. Probing .............................................................................. 56
    11.1 Error Cases ................................................................... 56
        Notes .............................................................................. 56
        Examples .......................................................................... 56
        Detect piece position ...................................................... 56
12. RTCP .................................................................................. 58
    12.1 Gcode .......................................................................... 58
        G43.4 ............................................................................. 58
        G43.7 ............................................................................. 58
        Example .......................................................................... 59
    12.2 Supported Kinematics .................................................... 59
13. Acknowledgement ................................................................ 61
1. Supported G and M Codes

1.1 Supported G Codes

The following table describes supported G-Code commands. The G-codes and M-codes called in the same line of a G-code file are executed accordingly to G Code Order of Execution.

In the following axes means one or more of X,Y,Z,A,B,C, along with a corresponding floating point value for a specified axis.

<table>
<thead>
<tr>
<th>Gcode</th>
<th>Parameters</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>G0</td>
<td>axes</td>
<td>Straight traverse</td>
<td>Traverse at maximum velocity</td>
</tr>
<tr>
<td>G1</td>
<td>axes F</td>
<td>Straight feed</td>
<td>Move at feed rate F</td>
</tr>
<tr>
<td>G2</td>
<td>axes F P IJK or R</td>
<td>Clockwise arc feed</td>
<td>Arc at feed rate F</td>
</tr>
<tr>
<td>G3</td>
<td>axes F P IJK or R</td>
<td>Counterclockwise arc feed</td>
<td>Arc at feed rate F</td>
</tr>
<tr>
<td>G4</td>
<td>P</td>
<td>Dwell</td>
<td>Pause for P seconds</td>
</tr>
<tr>
<td>G9</td>
<td>axes F</td>
<td>Exact Stop (non-modal)</td>
<td>Move at feed rate F and stop at the end</td>
</tr>
<tr>
<td>G10 L1</td>
<td>P axes</td>
<td>Set Tool Table Entry</td>
<td>Update the tool table adding/updating the tool info with the tool offset set by the user. Arguments:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• P tool id</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• X tool offset X</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Y tool offset Y</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Z tool offset Z</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• D tool diameter</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 0 tool type</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 1 tool parameter 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• J tool parameter 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• K tool parameter 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• V tool slot: this should be specified only if the tool is not already present in the tool table.</td>
</tr>
<tr>
<td>G10 L10</td>
<td>P axes</td>
<td>Set Tool Table, Calculated, Workpiece</td>
<td>Update the tool table adding/updating the tool info with the tool offset calculated by the G-code interpreter. Arguments:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• P tool id</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• X the X position that should be considered 0 when the tool is loaded and tool offset are used</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Y the Y position that should be considered 0 when the tool is loaded and tool offset are used</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Z the Z position that should be considered 0 when the tool is loaded and tool offset are used</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• D tool diameter</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 0 tool type</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• 1 tool parameter 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• J tool parameter 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• K tool parameter 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• V tool slot: this should be specified only if the tool is not already present in the tool table.</td>
</tr>
<tr>
<td>G10 L2</td>
<td>P axes</td>
<td>Coordinate System Origin Setting</td>
<td>G10 L2 offsets the origin of the coordinate system specified by P to the value of the axis word. The offset is from the machine origin established during homing. The offset value will replace any current offsets in effect for the coordinate system specified. Axis words not used will not be changed.</td>
</tr>
<tr>
<td>G10 L20</td>
<td>P axes</td>
<td>Coordinate Origin Setting Calculated</td>
<td>G10 L20 is similar to G10 L2 except that instead of setting the WCS origin offset, it calculates and sets the values that makes the current coordinates corresponds to the specified arguments.</td>
</tr>
<tr>
<td>Gcode</td>
<td>Parameters</td>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>------------------</td>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>G10 L100</td>
<td>P&lt;parameter&gt; V&lt;value&gt;</td>
<td>G10 L100</td>
<td>Set the value of some special parameters</td>
</tr>
<tr>
<td>G15</td>
<td></td>
<td>G15</td>
<td>Switch to Cartesian coordinates</td>
</tr>
<tr>
<td>G16</td>
<td></td>
<td>G16</td>
<td>Switch to polar coordinates</td>
</tr>
<tr>
<td>G17</td>
<td>axes</td>
<td>G17</td>
<td>Select XY arc plane</td>
</tr>
<tr>
<td>G18</td>
<td>axes</td>
<td>G18</td>
<td>Select XZ arc plane</td>
</tr>
<tr>
<td>G19</td>
<td></td>
<td>G19</td>
<td>Select YZ arc plane</td>
</tr>
<tr>
<td>G20</td>
<td></td>
<td>G20</td>
<td>Select inches mode</td>
</tr>
<tr>
<td>G21</td>
<td></td>
<td>G21</td>
<td>Select mm mode</td>
</tr>
<tr>
<td>G28</td>
<td>axes</td>
<td>G28</td>
<td>Go to G28 position</td>
</tr>
<tr>
<td>G28.1</td>
<td></td>
<td>G28.1</td>
<td>Set G28 position</td>
</tr>
<tr>
<td>G30</td>
<td>axes</td>
<td>G30</td>
<td>Go to G30 position</td>
</tr>
<tr>
<td>G30.1</td>
<td></td>
<td>G30.1</td>
<td>Set G30 position</td>
</tr>
<tr>
<td>G38.2</td>
<td>G38.3</td>
<td>G38.2 G38.3</td>
<td>Probing</td>
</tr>
<tr>
<td>G38.4</td>
<td>G38.5</td>
<td>G38.4 G38.5</td>
<td>Disable Cutter Compensation</td>
</tr>
<tr>
<td>G40</td>
<td></td>
<td>G40</td>
<td>Disable Cutter Compensation</td>
</tr>
<tr>
<td>G41</td>
<td>D I</td>
<td>G41</td>
<td>Enable Cutter Compensation left of programmed path</td>
</tr>
<tr>
<td>G41.1</td>
<td>D L</td>
<td>G41.1</td>
<td>Enable Dynamic Cutter Compensation left of programmed path</td>
</tr>
<tr>
<td>G42</td>
<td>D I</td>
<td>G42</td>
<td>Enable Cutter Compensation right of programmed path</td>
</tr>
<tr>
<td>G42.1</td>
<td>D L</td>
<td>G42.1</td>
<td>Enable Dynamic Cutter Compensation right of programmed path</td>
</tr>
<tr>
<td>G43</td>
<td>H</td>
<td>G43</td>
<td>Enable Tool Length Compensation</td>
</tr>
<tr>
<td>G43.1</td>
<td>X Y Z</td>
<td>G43.1</td>
<td>Enable Dynamic Tool Length Compensation</td>
</tr>
<tr>
<td>G43.2</td>
<td>H</td>
<td>G43.2</td>
<td>Apply additional Tool Length Offset</td>
</tr>
<tr>
<td>G43.4</td>
<td>H</td>
<td>G43.4</td>
<td>Enable RTCP feature</td>
</tr>
<tr>
<td>G43.7</td>
<td>H &lt;X&gt; &lt;Y&gt; &lt;Z&gt;</td>
<td>G43.7</td>
<td>Enable RTCP feature and override tool offsets using user defined arguments</td>
</tr>
<tr>
<td>G49</td>
<td></td>
<td>G49</td>
<td>Cancel Tool Length Compensation and disable RTCP feature</td>
</tr>
<tr>
<td>G50</td>
<td></td>
<td>G50</td>
<td>Disable scaling</td>
</tr>
<tr>
<td>G51</td>
<td>X Y Z and I J K or P</td>
<td>G51</td>
<td>Enable scaling</td>
</tr>
<tr>
<td>G52</td>
<td>axes</td>
<td>G52</td>
<td>Local Work Shift</td>
</tr>
<tr>
<td>G53</td>
<td></td>
<td>G53</td>
<td>Select absolute coordinates</td>
</tr>
<tr>
<td>G54</td>
<td></td>
<td>G54</td>
<td>Select coord system 1</td>
</tr>
<tr>
<td>G55</td>
<td></td>
<td>G55</td>
<td>Select coord system 2</td>
</tr>
<tr>
<td>G56</td>
<td></td>
<td>G56</td>
<td>Select coord system 3</td>
</tr>
<tr>
<td>G57</td>
<td></td>
<td>G57</td>
<td>Select coord system 4</td>
</tr>
<tr>
<td>G58</td>
<td></td>
<td>G58</td>
<td>Select coord system 5</td>
</tr>
<tr>
<td>G59</td>
<td></td>
<td>G59</td>
<td>Select coord system 6</td>
</tr>
<tr>
<td>G59.1</td>
<td></td>
<td>G59.1</td>
<td>Select coord system 7</td>
</tr>
<tr>
<td>G59.2</td>
<td></td>
<td>G59.2</td>
<td>Select coord system 8</td>
</tr>
<tr>
<td>G59.3</td>
<td></td>
<td>G59.3</td>
<td>Select coord system 9</td>
</tr>
<tr>
<td>G61</td>
<td></td>
<td>G61</td>
<td>Set exact path mode</td>
</tr>
<tr>
<td>G61.1</td>
<td></td>
<td>G61.1</td>
<td>Set exact stop mode</td>
</tr>
<tr>
<td>G64</td>
<td>P Q</td>
<td>G64</td>
<td>Continuous path mode</td>
</tr>
<tr>
<td>G65</td>
<td>P A B C ...</td>
<td>G65</td>
<td>Macro call</td>
</tr>
<tr>
<td>G66</td>
<td>P A B C ...</td>
<td>G66</td>
<td>Macro modal call</td>
</tr>
<tr>
<td>G67</td>
<td></td>
<td>G67</td>
<td>Macro modal call cancel</td>
</tr>
<tr>
<td>G68</td>
<td>X Y Z R</td>
<td>G68</td>
<td>Coordinate System Rotation</td>
</tr>
<tr>
<td>G69</td>
<td></td>
<td>G69</td>
<td>Cancel Coordinate System Rotation</td>
</tr>
<tr>
<td>G73</td>
<td>X Y Z R Q &lt;L&gt;</td>
<td>G73</td>
<td>High Speed Peck Drilling Cycle Chip Break Drilling Cycle</td>
</tr>
<tr>
<td>G80</td>
<td></td>
<td>G80</td>
<td>Cancel motion mode</td>
</tr>
<tr>
<td>Gcode</td>
<td>Parameters</td>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>------------</td>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>G81</td>
<td>XYZRL</td>
<td>G81</td>
<td>Drilling Cycle</td>
</tr>
<tr>
<td>G82</td>
<td>XYZRLP</td>
<td>G82</td>
<td>Drilling Cycle, Dwell</td>
</tr>
<tr>
<td>G83</td>
<td>XYZRLQ</td>
<td>G83</td>
<td>Peck Drilling Cycle</td>
</tr>
<tr>
<td>G85</td>
<td>XYZRL</td>
<td>G85</td>
<td>Boring Cycle, Feed Out</td>
</tr>
<tr>
<td>G86</td>
<td>XYZRLP</td>
<td>G86</td>
<td>Boring Cycle, Spindle Stop, Rapid Move Out</td>
</tr>
<tr>
<td>G88</td>
<td></td>
<td>G88</td>
<td>Boring Cycle, Spindle Stop, Manual Out</td>
</tr>
<tr>
<td>G89</td>
<td></td>
<td>G89</td>
<td>Boring Cycle, Dwell, Feed Out</td>
</tr>
<tr>
<td>G90</td>
<td></td>
<td>G90</td>
<td>Set absolute distance mode</td>
</tr>
<tr>
<td></td>
<td></td>
<td>G90.1</td>
<td>Set absolute arc distance mode</td>
</tr>
<tr>
<td>G91</td>
<td></td>
<td>G91</td>
<td>Set incremental distance mode</td>
</tr>
<tr>
<td>G91.1</td>
<td></td>
<td>G91.1</td>
<td>Set incremental arc distance mode</td>
</tr>
<tr>
<td>G92</td>
<td>axes</td>
<td>G92</td>
<td>Set origin offsets</td>
</tr>
<tr>
<td></td>
<td></td>
<td>G92.1</td>
<td>Reset origin offsets</td>
</tr>
<tr>
<td></td>
<td></td>
<td>G92.2</td>
<td>Suspend origin offsets</td>
</tr>
<tr>
<td></td>
<td></td>
<td>G92.3</td>
<td>Resume origin offsets</td>
</tr>
<tr>
<td>G93</td>
<td></td>
<td>G93</td>
<td>Enable feed per inverse of time</td>
</tr>
<tr>
<td>G94</td>
<td></td>
<td>G94</td>
<td>Enable feed per minute</td>
</tr>
<tr>
<td>G98</td>
<td></td>
<td>G98</td>
<td>Canned cycle return mode</td>
</tr>
<tr>
<td>G99</td>
<td></td>
<td>G99</td>
<td>Canned cycle return mode</td>
</tr>
<tr>
<td>G100</td>
<td>PABC...</td>
<td>G100</td>
<td>Internal PLC function Call</td>
</tr>
<tr>
<td>G200+G499</td>
<td>ABC...</td>
<td>G200+G499</td>
<td>User defined G codes</td>
</tr>
</tbody>
</table>
### 1.2 Supported M Codes

<table>
<thead>
<tr>
<th>Mcode</th>
<th>Parameters</th>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>M0</td>
<td></td>
<td>Program Stop</td>
<td>Pause a running program temporarily</td>
</tr>
<tr>
<td>M2</td>
<td></td>
<td>Program End</td>
<td>End a G-code program and reset the machine state: switch off I/O, mist, flood and spindle.</td>
</tr>
<tr>
<td>M3</td>
<td></td>
<td>Start spindle clock wise</td>
<td></td>
</tr>
<tr>
<td>M4</td>
<td></td>
<td>Start spindle counter clock wise</td>
<td></td>
</tr>
<tr>
<td>M5</td>
<td></td>
<td>Stop spindle turning</td>
<td></td>
</tr>
<tr>
<td>M6</td>
<td></td>
<td>Tool change</td>
<td>See also User Tool Change Subprogram</td>
</tr>
<tr>
<td>M7</td>
<td></td>
<td>Mist On</td>
<td></td>
</tr>
<tr>
<td>M8</td>
<td></td>
<td>Flood On</td>
<td></td>
</tr>
<tr>
<td>M9</td>
<td></td>
<td>Mist and Flood Off</td>
<td></td>
</tr>
<tr>
<td>M17</td>
<td></td>
<td>Turn On Torch Height Control (THC)</td>
<td></td>
</tr>
<tr>
<td>M18</td>
<td></td>
<td>Turn Off Torch Height Control (THC)</td>
<td></td>
</tr>
<tr>
<td>M30</td>
<td></td>
<td>Pallet shuttle and program end</td>
<td>If the correspondent settings is enabled the user G-code macro pallet_shuttle.ngc can be called.</td>
</tr>
<tr>
<td>M47</td>
<td></td>
<td>Restart program execution</td>
<td></td>
</tr>
<tr>
<td>M60</td>
<td></td>
<td>Pallet shuttle</td>
<td>If the correspondent settings is enabled the user G-code macro pallet_shuttle.ngc can be called.</td>
</tr>
<tr>
<td>M61</td>
<td>Q</td>
<td>Set the current tool without performing a tool change.</td>
<td>Could be called from user G-code defined tool change macro, see also User Tool Change Subprogram.</td>
</tr>
<tr>
<td>M62</td>
<td>P</td>
<td>Turn out ON</td>
<td></td>
</tr>
<tr>
<td>M63</td>
<td>P</td>
<td>Turn out OFF</td>
<td></td>
</tr>
</tbody>
</table>
| M66   | P L Q      | Wait input | Parameters:  
- P: the number of the user input signal to be waited  
- L:  
  - 0: waits for the selected input to reach the LOW state  
  - 1: waits for the selected input to reach the HIGH state  
  - 2: waits for the selected input to perform a FALL event  
  - 3: waits for the selected input to perform a RISE event  
  - 4: return immediately and the input value is stored in #5720  
  - 10 to 13: wait for the input to be LOW, HIGH, FALL or RISE and generate a CNC alarm if timeout elapses while waiting  
- Q: timeout in seconds (optional)  
  - If the correspondent compiler setting is enabled the input value is stored in parameter #5720 and can be used in the following control flow statements (IF, WHILE,...).  
  - If the Q parameter is missing, the instruction waits until the condition is satisfied. |
| M68   | P          | Read analog input (Not yet supported) | |
| M98   | P L        | Call Subroutine | Note: a named external subroutine can be called |
| M99   |            | Return from Subroutine | |
| M102  |            | End Program without reset | End a program but does not perform an automatic reset switching off mist, flood, spindle status, ... |
| M106  |            | Execute PLC internal tool change procedure | This M code is intended to be called by the user from the file named tool_change.ngc. This code inform the PLC to start the internal tool change procedure that can perform a few automatic actions not described by the procedure described using the G-code. See also User Tool Change Subprogram. |
| M107  |            | Inform the PLC that the tool change procedure written in the User Tool Change Subprogram has started. | |
| M108  |            | Inform the PLC that the tool change procedure written in the User Tool Change Subprogram has ended. | |
| M109  | P Q D      | Show user message | |
| M120  | P Q D      | Show user media | |
| M200-M299 | A B C ... | User defined M codes | User defined M codes require a user edited G-code file with the macro name and .ngc extension to be place in the directory <Documents/RosettaCNC-1>/machines/<machine_name>/macros (eg: m210.ngc). |
1.3 Other Codes

Simple G-code commands are used for setting the speed, feed, and tool parameters.

“F” for “Feed”

The F command sets the feed rate; the machine operates at the set feed rate when G1 is used, and subsequent G1 commands will execute at the set F value.

If the feed rate (F) is not set once before the first G1 call, either an error will occur or the machine will operate at its “default” feed rate.

An example of a valid F command: G1 F1500 X100 Y100

To see how RosettaCNC handled the feed rate when for single rotary axis moves or for mixed moves please check Feed Management

“S” for “Spindle Speed”

The S command sets the spindle speed, typically in revolutions per minute (RPM). An example of a valid S command: S10000

“T” for “Tool”

The T command is used to set the id of the tool to be loaded with the M6 command.

The typical syntax to load the tool with id 1 would be:

M6 T1

Notes

- If the automatic tool change feature is enabled in the board settings, the M6 command produces a tool change handled by RosettaCNC internal code that can be adapted using dedicated parameters.

- Setting the Tool Change Type option to Custom Macro in the board settings the user can customise the tool change procedure. If the option is enabled the M6 command will look into the machine macro folder and execute the G-code file named tool_change.ngc. In this file the user can specify any supported G-code command to perform the tool change procedure as required by the specific machine. To see a reference implementation of the file tool_change.ngc please take a look to User Tool Change Subprogram.

1.4 G-Code Comments

The following syntaxes are supported:

- (…) : simple comment between brackets
- ; : simple comment till the end of the line started with a semi column

1.5 Block delete

Block Delete, also called Optional Skip, determines what happens when a line of code has a forward slash mark (/). In RosettaCNC integrated G-Code editor there is a dedicated icon to enable/disable this feature. When the feature is enabled and a line of G-Code begins with a forward slash the line is ignored and the execution skips to the next line of code.
1.6 Go to predefined positions

1.6.1 G28 and G28.1

G28 uses the values stored in parameters 5161-5166 as the X Y Z A B C final point to move to. The parameter values are absolute machine coordinates

- **G28** - makes a rapid move from the current position to the absolute position of the values in parameters 5161-5166. Since the position stored in the parameters 5161-5166 is considered absolute the tool offset enabled with G43 influences the position. Example: \#5163 = 0, tool_offset_z = 50 \(\rightarrow\) target_position = \#5163 - tool_offset_z = -50.
- **G28 axes** - makes a rapid move to the position specified by axes including any offsets, then will make a rapid move to the absolute position of the values in parameters 5161-5166 for axes specified. Any axis not specified will not move.
- **G28.1** - stores the current absolute position into parameters 5161-5166. Note: G28.1 does not take any argument, the current absolute machine position is stored.

**Examples**

G28 (rapid move to the values specified into the stored parameters, moving all axes)
G28 Z10.0 (rapid move to Z10.0 then to location specified in the G28 stored parameters, moving only Z)
G28 G91 Z0 (rapid relative move to relative Z0.0 then to location specified in the G28 stored parameters, moving only Z)

The last example skip the intermediate position, since the movement is relative and with a displacement of 0. It is usually used to ensure that only axis Z will move to the homing position specified in G28 parameters.

1.6.2 G30 and G30.1

G30 uses the values stored in parameters 5181-5186 as the X Y Z A B C final point to move to. The parameter values are absolute machine coordinates

- **G30** - makes a rapid move from the current position to the absolute position of the values in parameters 5181-5186.
- **G30 axes** - makes a rapid move to the position specified by axes including any offsets, then will make a rapid move to the absolute position of the values in parameters 5181-5186 for axes specified. Any axis not specified will not move.
- **G30.1** - stores the current absolute position into parameters 5181-5186. Note: G30.1 does not take any argument, the current absolute machine position is stored.

**1.6.2.1 Examples**
G17 G21 G40 G49 G50 G54 G69 G90

#5161=20  ( G28 X )
#5162=40  ( G28 Y )
#5163=20  ( G28 Z )

G52 X20 Y20 G91 Z0
M98 P1000
; Since Z axis is specified only axis Z will be moved to the position stored in parameter 5163.
; Since the intermediate point Z0 is specified with G91 the intermediate point position will be skipped.
G28 G91 Z0
G90

G52 X50 Y50
M98 P1000
; If no axis is specified with the G28 the position stored in the parameters 5161-5166 is used.
G28
M2

( Square )
01000
G50
G8 X0 Y0 Z0
G1 X10
Y10
X10
Y10
M99
1.7 G Code Order of Execution

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

- the entire line is skipped if it starts with a forward slash / and the block delete toggle is active
- comments started with ( or ;
- if N is the first letter of a line the following number is interpreted as line number
- when a subroutine declaration is found (example O1001) the remaining part of the line is allowed only to be a comment.
- control flow statements like WHILE,IF
- set feed rate mode G93, G94
- set feed rate F
- set spindle speed S
- I/O handling: M62, M63, M66
- change tool: M6 if user tool change macro is disabled, M61, M106
- spindle on or off: M3, M4, M5
- coolant on or off: M7, M8, M9, M17, M18
- M48, M49
- M109, M120
- dwell G4
- set active plane: G17, G18, G19
- set length units: G20, G21
- cutter radius compensation on or off: G40, G41, G42
- cutter length compensation on or off: G43, G49
- coordinate system selection: G54, G55, G56, G57, G58, G59, G59.1, G59.2, G59.3
- set path control mode: G61, G61.1, G64
- set distance mode: G90, G91
- set arc mode: G90.1, G91.1
- set retract mode: G98, G99
- non modal G-codes: G10, G28, G28.1, G30, G30.1, G52, G92, G92.1, G92.2, G92.3
- scaling: G50, G51
- rotation: G68, G69
- motion: G0, G1, G2, G3, G9, G76, G80
- stop: M0, M1, M2, M30, M47
- control flow statements like GOTO
- subroutines and macros: M98, M6 only if user tool change macro is enabled

Some codes require to be the only G/M codes in the line, they are: G65, G100, user defined m codes and user defined g codes
1.8 Arcs & Helices

A circular or helical arc is specified using either \texttt{G2} (clockwise arc) or \texttt{G3} (counterclockwise arc) at the current feed rate. The direction (CW, CCW) is as viewed from the positive end of the axis about which the circular motion occurs.

The plane used for the circle or helix must be one between XY, YZ, or XZ. The plane is selected with \texttt{G17} (XY-plane), \texttt{G18} (XZ-plane), or \texttt{G19} (YZ-plane).

The \texttt{P} word can be used to specify the number of full turns plus the programmed arc. The \texttt{P} word must be an integer. If \texttt{P} is unspecified, the behaviour is as if \texttt{P1} was given: that is, only one full or partial turn will result. For each \texttt{P} increment above 1 an extra full circle is added to the programmed arc. Multi turn helical arcs are supported and give motion useful for milling holes or threads.

\textbf{Note} If a line of code makes an arc and includes rotary axis motion, the rotary axes turn so that the rotary motion starts and finishes when the XYZ motion starts and finishes.

1.8.1 Syntax

1.8.2 Center Format Arcs

Center format arcs are more accurate than radius format arcs and are the preferred format to use. The end point of the arc along with the offset to the center of the arc from the current location are used to program arcs that are less than a full circle.

If the end point of the arc is the same as the current location a full circle is generated.

\texttt{G2 or G3 axes offsets P}
\texttt{G2 or G3 offsets P} can be used for full circles

\textbf{Incremental Arc Distance Mode}

This is the default Distance Mode for the arc center offsets (I, J, K) that are relative distances from the start location of the arc. One or more axis words and one or more offsets must be programmed for an arc that is less than 360 degrees. This mode is enabled with \texttt{G91.1}. 

**Absolute Arc Distance Mode**

Arc center offsets (I, J, K) are relative distances from the current origin of the axes. One or more axis words and both offsets must be programmed for arcs less than 360 degrees. This mode is enabled with **G90.1**.

### 1.8.3 Radius Format Arcs

G2 or G3 axes R- P

### 1.8.4 Examples

### 1.8.5 Center format arcs incremental mode
1.8.6 Full circles and helices

{ © 2018 by RosettaCNC Motion
file name: full_circles_and_helices.ngc }

G21 G40 G49 G90 G54 G50 G69
M3 F2000
G0 X0 Y0 Z0
G17 ; XY plane
G2 I0 J25
G18 ; XZ plane
G2 I0 K25
G19 ; YZ plane
G2 J0 K25
T3 M6
G52 X100
G0 X0 Y0 Z0
G17 ; helixes
G2 I0 J25 Z100 P2
G0 X0 Y0 Z0
G18 ; XZ plane
G2 I0 K25 Y100 P2
G0 X0 Y0 Z0
G19 ; YZ plane
G2 J0 K25 X100 P2
M2
2. G-code variables

Rosetta CNC supports **Macro programming** (following Fanuc Macro B style). Your G-code programs or sub-programs can include a few non G-code commands that use variables, arithmetic, logic statements, and looping are available.

Rosetta CNC supports 7000 variables that can be accessed from the G-code. These variables are divided in 5 groups as described in the following table.

<table>
<thead>
<tr>
<th>Variable Number</th>
<th>Type of Variable</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>#0</td>
<td>Null</td>
<td>#0 is read only and its value is always &quot;null&quot; that means &quot;no value&quot;.</td>
</tr>
<tr>
<td>#1-#33</td>
<td>Local Variables</td>
<td>Local variables are used to pass arguments to macros and as temporary scratch storage.</td>
</tr>
<tr>
<td>#100-#3999</td>
<td>Input Variables</td>
<td>The value of these variables can be set by the user through the GUI and the value in the table will never be changed by the controller.</td>
</tr>
<tr>
<td>#500-#4999</td>
<td>Program Variables</td>
<td>The value of these variables can be read/write by G-code programs and are initialised with #0 before executing a program.</td>
</tr>
<tr>
<td>#4000-#4999</td>
<td>Shared Variables</td>
<td>The value of these variables is shared between the GUI interface and the controller. The result is that if during a program one of these variables is changed the user can see the updated final value in the table.</td>
</tr>
<tr>
<td>#5000-#5999</td>
<td>System Variables</td>
<td>Updated runtime by the compiler</td>
</tr>
<tr>
<td>#6000-#6999</td>
<td>Protected Variables</td>
<td>Variables that can be modified by the user only before compilation and that are password protected. A G-code program can only read these variables and not write them.</td>
</tr>
</tbody>
</table>

2.1 System variables description

<table>
<thead>
<tr>
<th>System Variable</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Current Position Variables</strong></td>
<td>Current position X - C, Interpreter position = work position. <a href="#">See position information table</a></td>
</tr>
<tr>
<td>5001-5006</td>
<td>Sequence number of lines executed</td>
</tr>
<tr>
<td>5100</td>
<td>Group 01: G0, G1, G2, G3, G38.X, G80, G81, G82, G83, G84, G85</td>
</tr>
<tr>
<td>5102</td>
<td>Group 02: G17, G18, G19</td>
</tr>
<tr>
<td>5103</td>
<td>Group 03: G90, G91</td>
</tr>
<tr>
<td>5104</td>
<td>Group 04: G90.1, G91.1</td>
</tr>
<tr>
<td>5105</td>
<td>Group 05: G93, G94</td>
</tr>
<tr>
<td>5106</td>
<td>Group 06: G20, G21</td>
</tr>
<tr>
<td>5107</td>
<td>Group 07: G40, G41, G41.1, G42, G42.1</td>
</tr>
<tr>
<td>5108</td>
<td>Group 08: G43, G43.1, G43.2, G43.4, G43.7, G49</td>
</tr>
<tr>
<td>5110</td>
<td>Group 10: G98, G99</td>
</tr>
<tr>
<td>5111</td>
<td>Group 11: G50, G51</td>
</tr>
<tr>
<td>5112</td>
<td>Group 12: G54, G55, G56, G57, G58, G59, G59.1, G59.2, G59.3</td>
</tr>
<tr>
<td>5113</td>
<td>Group 13: G61, G61.1, G64</td>
</tr>
<tr>
<td>5116</td>
<td>Group 16: G68, G69</td>
</tr>
<tr>
<td>5117</td>
<td>Group 17: G15, G16</td>
</tr>
<tr>
<td><strong>Active G-codes Variables</strong></td>
<td>Active G-codes Variables Meaning</td>
</tr>
<tr>
<td>5130</td>
<td>Active feed rate (F)</td>
</tr>
<tr>
<td>5131</td>
<td>Active spindle speed (S)</td>
</tr>
<tr>
<td>5132</td>
<td>Selected tool (T)</td>
</tr>
<tr>
<td>5133</td>
<td>Selected slot</td>
</tr>
<tr>
<td>5134</td>
<td>Current tool</td>
</tr>
<tr>
<td>5135</td>
<td>Current slot</td>
</tr>
<tr>
<td>5140</td>
<td>Tolerance set with G64 P</td>
</tr>
<tr>
<td>5141</td>
<td>Points removal threshold set with G64 Q</td>
</tr>
<tr>
<td>5148</td>
<td>While a modal macro is active it stores how many times a modal macro (G66) has been called.</td>
</tr>
<tr>
<td>5149</td>
<td>Executing sub-program. Flag set to 1 while G-code is executing a sub-program called from the main program.</td>
</tr>
<tr>
<td><strong>Active M-codes</strong></td>
<td>Active M-codes meaning</td>
</tr>
<tr>
<td>5150</td>
<td>M0, M2, M30, M47, M60</td>
</tr>
<tr>
<td>5151</td>
<td>M3, M4, M5</td>
</tr>
<tr>
<td>5152</td>
<td>M6, M106</td>
</tr>
<tr>
<td>5153</td>
<td>M7, M9</td>
</tr>
<tr>
<td>5154</td>
<td>M8, M9</td>
</tr>
<tr>
<td><strong>G28, G28.1 Variables</strong></td>
<td>G28, G28.1 Variables Meaning</td>
</tr>
<tr>
<td>5161</td>
<td>G28.1 position X</td>
</tr>
<tr>
<td>5162</td>
<td>G28.1 position Y</td>
</tr>
<tr>
<td>System Variable</td>
<td>Meaning</td>
</tr>
<tr>
<td>--------------------------</td>
<td>----------------------------------</td>
</tr>
<tr>
<td><strong>Current Position Variables</strong></td>
<td><strong>Current Position variables Meaning</strong></td>
</tr>
<tr>
<td>5163</td>
<td>G28.1 position Z</td>
</tr>
<tr>
<td>5164</td>
<td>G28.1 position A</td>
</tr>
<tr>
<td>5165</td>
<td>G28.1 position B</td>
</tr>
<tr>
<td>5166</td>
<td>G28.1 position C</td>
</tr>
<tr>
<td><strong>G30, G30.1 Variables</strong></td>
<td><strong>G30, G30.1 Variables Meaning</strong></td>
</tr>
<tr>
<td>5181</td>
<td>G30.1 position X</td>
</tr>
<tr>
<td>5182</td>
<td>G30.1 position Y</td>
</tr>
<tr>
<td>5183</td>
<td>G30.1 position Z</td>
</tr>
<tr>
<td>5184</td>
<td>G30.1 position A</td>
</tr>
<tr>
<td>5185</td>
<td>G30.1 position B</td>
</tr>
<tr>
<td>5186</td>
<td>G30.1 position C</td>
</tr>
<tr>
<td><strong>WCS Offsets Variables</strong></td>
<td><strong>WCS Offsets Variables Meaning</strong></td>
</tr>
<tr>
<td>5201 - 5206</td>
<td>G52 offset X - C</td>
</tr>
<tr>
<td>5210</td>
<td>G92 enabled (0 ÷ 1)</td>
</tr>
<tr>
<td>5211 - 5216</td>
<td>G92 offset X - C</td>
</tr>
<tr>
<td><strong>WCS Variables</strong></td>
<td><strong>WCS Variables Meaning</strong></td>
</tr>
<tr>
<td>5220</td>
<td>Coord. System number</td>
</tr>
<tr>
<td>5221 - 5226</td>
<td>Coord. System 1 X – C</td>
</tr>
<tr>
<td>5241 - 5246</td>
<td>Coord. System 2 X – C</td>
</tr>
<tr>
<td>5261 - 5266</td>
<td>Coord. System 3 X – C</td>
</tr>
<tr>
<td>5281 - 5286</td>
<td>Coord. System 4 X – C</td>
</tr>
<tr>
<td>5301 - 5306</td>
<td>Coord. System 5 X – C</td>
</tr>
<tr>
<td>5321 - 5326</td>
<td>Coord. System 6 X – C</td>
</tr>
<tr>
<td>5341 - 5346</td>
<td>Coord. System 7 X – C</td>
</tr>
<tr>
<td>5361 - 5366</td>
<td>Coord. System 8 X – C</td>
</tr>
<tr>
<td>5381 - 5386</td>
<td>Coord. System 9 X – C</td>
</tr>
<tr>
<td><strong>Tool Variables</strong></td>
<td><strong>Tool Variables Meaning</strong></td>
</tr>
<tr>
<td>5400</td>
<td>Current tool id</td>
</tr>
<tr>
<td>5401</td>
<td>Current tool offset X</td>
</tr>
<tr>
<td>5402</td>
<td>Current tool offset Y</td>
</tr>
<tr>
<td>5403</td>
<td>Current tool offset Z / Current tool length</td>
</tr>
<tr>
<td>5410</td>
<td>Current tool diameter</td>
</tr>
<tr>
<td>5411</td>
<td>Current tool type</td>
</tr>
<tr>
<td>5412</td>
<td>Current tool parameter 1</td>
</tr>
<tr>
<td>5413</td>
<td>Current tool parameter 2</td>
</tr>
<tr>
<td>5414</td>
<td>Current tool parameter 3</td>
</tr>
<tr>
<td>5420</td>
<td>Tool compensation offset X (Set using G43, G43.1, G43.2, G49)</td>
</tr>
<tr>
<td>5421</td>
<td>Tool compensation offset Y (Set using G43, G43.1, G43.2, G49)</td>
</tr>
<tr>
<td>5422</td>
<td>Tool compensation offset Z (Set using G43, G43.1, G43.2, G49)</td>
</tr>
<tr>
<td>5426</td>
<td>The id of the tool used for RTCP compensation (G43.4)</td>
</tr>
<tr>
<td><strong>Scaling &amp; Rotation Variables</strong></td>
<td><strong>Scaling and Rotation Variables Meaning</strong></td>
</tr>
<tr>
<td>5501</td>
<td>G51 scaling factor X</td>
</tr>
<tr>
<td>5502</td>
<td>G51 scaling factor Y</td>
</tr>
<tr>
<td>5503</td>
<td>G51 scaling factor Z</td>
</tr>
<tr>
<td>5504</td>
<td>G51 offset X</td>
</tr>
<tr>
<td>5505</td>
<td>G51 offset Y</td>
</tr>
<tr>
<td>5506</td>
<td>G51 offset Z</td>
</tr>
<tr>
<td>5510</td>
<td>G68 rotation plane</td>
</tr>
<tr>
<td>5511</td>
<td>G68 rotation XY</td>
</tr>
<tr>
<td>5512</td>
<td>G68 rotation XZ</td>
</tr>
<tr>
<td>5513</td>
<td>G68 rotation YZ</td>
</tr>
<tr>
<td>5514</td>
<td>G68 offset X</td>
</tr>
<tr>
<td>5515</td>
<td>G68 offset Y</td>
</tr>
<tr>
<td>5516</td>
<td>G68 offset Z</td>
</tr>
<tr>
<td><strong>Runtime External Variables</strong></td>
<td><strong>Runtime External Variables Meaning</strong></td>
</tr>
<tr>
<td>5700</td>
<td>Probe result at the end of a G38.X. Values: 1 probing procedure succeed, -1 failed: sensor not tripped before reaching the target position, -2 failed: sensor already tripped</td>
</tr>
<tr>
<td>5701 - 5706</td>
<td>Probed position loaded at the end of a G38.X with respect to the active WCS See position information table</td>
</tr>
</tbody>
</table>
2.2 Named variables

Named variables work like normal numbered variables but are easier to read.

Syntax:
- Named variables must be enclosed between < > marks.
- All variable names are converted to lower case and have spaces and tabs removed, so <named variable> and < Named Var i Able> represent the very same variable.
- A named variable starts to exist when it is assigned a value for the first time.
- You can check if a named parameter already exists with the unary operation EXISTS[arg]. Example: IF [EXISTS[#<_args.c>] EQ 0] THEN GOTO 10
- When a macro is called the passed arguments can be read using the correspondent named variable preceded by _args. Example: to get the value of the x argument you can use #<_args.x>.

2.2.1 Global and local scopes

A named parameter whose name starts with _ is local to the scope in which it is created. A local named variable vanishes when its scope is left. Indeed, when a local variable is declared in a subroutine and the subroutine returns, the variable is deleted and cannot be referred to anymore.

- #<named_variable> is a global named variable.
- #<_named_variable> is a local named variable.

2.2.2 Indexing support

Both global and local variables support indexing with a syntax similar to C style arrays. Every expression between brackets ([,]) in a variable name is evaluated by the G-code interpreter.

Examples:
- #<named_variable[#<index>]> is evaluated as #<named_variable[10]> if #<index> has been previously set to 10.
- #<named_variable[#<index1> + 1][#<index2>]>is evaluated depending on the variables #<index1> and #<index2>

2.2.3 Pre-defined Named Variables

The G-code compiler has a pre-defined set of read-only named variables which can be useful in program and macro editing. There are five groups:

1. #<sys.xxx> which contains info about system.
2. #<math.xxx> which contains usefull math constants.
3. #<cnc.xxx> which contains some of CNC setup settings.
4. #<tool.xxx> which contains usefull tool constants.
5. #<probe.xxx> which contains usefull probe constants.
<table>
<thead>
<tr>
<th>Named Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#&lt;sys.version&gt;</td>
<td>The version of CNC system.</td>
</tr>
<tr>
<td>#&lt;sys.customer_id&gt;</td>
<td>The customer ID (usually values 0).</td>
</tr>
<tr>
<td>#&lt;sys.interface_level&gt;</td>
<td>Interface Level.</td>
</tr>
<tr>
<td>#&lt;math.e&gt;</td>
<td>Euler's number.</td>
</tr>
<tr>
<td>#&lt;math.pi&gt;</td>
<td>Archimedes constant.</td>
</tr>
<tr>
<td>#&lt;math.ln_2&gt;</td>
<td>Natural Log of 2.</td>
</tr>
<tr>
<td>#&lt;math.ln_10&gt;</td>
<td>Natural Log of 10.</td>
</tr>
<tr>
<td>#&lt;math.ln_pi&gt;</td>
<td>Natural Log of pi.</td>
</tr>
<tr>
<td>#&lt;math.to_mm&gt;</td>
<td>Factor to convert inches to mm (25.4).</td>
</tr>
<tr>
<td>#&lt;math.to_in&gt;</td>
<td>Factor to convert mm to inches (1/25.4).</td>
</tr>
<tr>
<td>#&lt;cnc.machine_type&gt;</td>
<td>Machine Type: 0 = Mill</td>
</tr>
<tr>
<td>#&lt;cnc.kinematics_model&gt;</td>
<td>Kinematics Model: 0 = Trivial 1 = Independent Rotational Axes 2 = Rotary Table AC 3 = Rotary Table BC 4 = Tilting Spindle AC</td>
</tr>
<tr>
<td>#&lt;cnc.x.enabled&gt;</td>
<td>X Axis Enabled State: 0 = Disabled 1 = Enabled</td>
</tr>
<tr>
<td>#&lt;cnc.x.type&gt;</td>
<td>X Axis Type: 0 = Linear Axis</td>
</tr>
<tr>
<td>#&lt;cnc.x.max_vel&gt;</td>
<td>X Axis Max Velocity [mm/min]</td>
</tr>
<tr>
<td>#&lt;cnc.x.acc&gt;</td>
<td>X Axis Acceleration [mm/s²]</td>
</tr>
<tr>
<td>#&lt;cnc.x.min_lim&gt;</td>
<td>X Axis Min Limit [mm]</td>
</tr>
<tr>
<td>#&lt;cnc.x.max_lim&gt;</td>
<td>X Axis Max Limit [mm]</td>
</tr>
<tr>
<td>#&lt;cnc.y.enabled&gt;</td>
<td>Y Axis Enabled State: 0 = Disabled 1 = Enabled</td>
</tr>
<tr>
<td>#&lt;cnc.y.type&gt;</td>
<td>Y Axis Type: 0 = Linear Axis</td>
</tr>
<tr>
<td>#&lt;cnc.y.max_vel&gt;</td>
<td>Y Axis Max Velocity [mm/min]</td>
</tr>
<tr>
<td>#&lt;cnc.y.acc&gt;</td>
<td>Y Axis Acceleration [mm/s²]</td>
</tr>
<tr>
<td>#&lt;cnc.y.min_lim&gt;</td>
<td>Y Axis Min Limit [mm]</td>
</tr>
<tr>
<td>#&lt;cnc.y.max_lim&gt;</td>
<td>Y Axis Max Limit [mm]</td>
</tr>
<tr>
<td>#&lt;cnc.z.enabled&gt;</td>
<td>Z Axis Enabled State: 0 = Disabled 1 = Enabled</td>
</tr>
<tr>
<td>#&lt;cnc.z.type&gt;</td>
<td>Z Axis Type: 0 = Linear Axis</td>
</tr>
<tr>
<td>#&lt;cnc.z.max_vel&gt;</td>
<td>Z Axis Max Velocity [mm/min]</td>
</tr>
<tr>
<td>#&lt;cnc.z.acc&gt;</td>
<td>Z Axis Acceleration [mm/s²]</td>
</tr>
<tr>
<td>#&lt;cnc.z.min_lim&gt;</td>
<td>Z Axis Min Limit [mm]</td>
</tr>
<tr>
<td>#&lt;cnc.z.max_lim&gt;</td>
<td>Z Axis Max Limit [mm]</td>
</tr>
<tr>
<td>#&lt;cnc.a.enabled&gt;</td>
<td>A Axis Enabled State: 0 = Disabled 1 = Enabled</td>
</tr>
<tr>
<td>#&lt;cnc.a.type&gt;</td>
<td>A Axis Type: 0 = Linear Axis 1 = Rotary Axis Free 2 = Rotary Axis for Head 3 = Rotary Axis for Table</td>
</tr>
<tr>
<td>#&lt;cnc.a.max_vel&gt;</td>
<td>A Axis Max Velocity [mm/min]</td>
</tr>
<tr>
<td>#&lt;cnc.a.acc&gt;</td>
<td>A Axis Acceleration [mm/s²]</td>
</tr>
<tr>
<td>#&lt;cnc.a.min_lim&gt;</td>
<td>A Axis Min Limit [mm]</td>
</tr>
<tr>
<td>#&lt;cnc.a.max_lim&gt;</td>
<td>A Axis Max Limit [mm]</td>
</tr>
<tr>
<td>#&lt;cnc.a.motion_mode&gt;</td>
<td>A Axis Motion Mode: 0 = Continuous 1 = Indexing</td>
</tr>
<tr>
<td>Named Variable</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td><code>&lt;cnc.a.convention&gt;</code></td>
<td>A Axis Convention:</td>
</tr>
<tr>
<td></td>
<td>0 = Normal</td>
</tr>
<tr>
<td></td>
<td>1 = Inverse</td>
</tr>
<tr>
<td><code>&lt;cnc.a.wrapped_rotary&gt;</code></td>
<td>A Axis Wrapped Rotary State:</td>
</tr>
<tr>
<td></td>
<td>0 = Disabled</td>
</tr>
<tr>
<td></td>
<td>1 = Enabled</td>
</tr>
<tr>
<td><code>&lt;cnc.a.parallel_to&gt;</code></td>
<td>A Axis Parallel to:</td>
</tr>
<tr>
<td></td>
<td>0 = X</td>
</tr>
<tr>
<td></td>
<td>1 = Y</td>
</tr>
<tr>
<td></td>
<td>2 = Z</td>
</tr>
<tr>
<td><code>&lt;cnc.a.origin_mode&gt;</code></td>
<td>A Axis Origin Mode:</td>
</tr>
<tr>
<td></td>
<td>0 = Custom</td>
</tr>
<tr>
<td></td>
<td>1 = WCS 1 - G54</td>
</tr>
<tr>
<td></td>
<td>2 = WCS 2 - G55</td>
</tr>
<tr>
<td></td>
<td>3 = WCS 3 - G56</td>
</tr>
<tr>
<td></td>
<td>4 = WCS 4 - G57</td>
</tr>
<tr>
<td></td>
<td>5 = WCS 5 - G58</td>
</tr>
<tr>
<td></td>
<td>6 = WCS 6 - G59</td>
</tr>
<tr>
<td></td>
<td>7 = WCS 7 - G59.1</td>
</tr>
<tr>
<td></td>
<td>8 = WCS 8 - G59.2</td>
</tr>
<tr>
<td></td>
<td>9 = WCS 9 - G59.3</td>
</tr>
<tr>
<td><code>&lt;cnc.a.origin_x&gt;</code></td>
<td>A Axis Origin X [mm]</td>
</tr>
<tr>
<td><code>&lt;cnc.a.origin_y&gt;</code></td>
<td>A Axis Origin Y [mm]</td>
</tr>
<tr>
<td><code>&lt;cnc.a.origin_z&gt;</code></td>
<td>A Axis Origin Z [mm]</td>
</tr>
<tr>
<td><code>&lt;cnc.b.enabled&gt;</code></td>
<td>B Axis Enabled State:</td>
</tr>
<tr>
<td></td>
<td>0 = Disabled</td>
</tr>
<tr>
<td></td>
<td>1 = Enabled</td>
</tr>
<tr>
<td><code>&lt;cnc.b.type&gt;</code></td>
<td>B Axis Type:</td>
</tr>
<tr>
<td></td>
<td>0 = Linear Axis</td>
</tr>
<tr>
<td></td>
<td>1 = Rotary Axis Free</td>
</tr>
<tr>
<td></td>
<td>3 = Rotary Axis for Table</td>
</tr>
<tr>
<td><code>&lt;cnc.b.max_vel&gt;</code></td>
<td>B Axis Max Velocity [mm/min]</td>
</tr>
<tr>
<td><code>&lt;cnc.b.acc&gt;</code></td>
<td>B Axis Acceleration [mm/s²]</td>
</tr>
<tr>
<td><code>&lt;cnc.b.min_lim&gt;</code></td>
<td>B Axis Min Limit [mm]</td>
</tr>
<tr>
<td><code>&lt;cnc.b.max_lim&gt;</code></td>
<td>B Axis Max Limit [mm]</td>
</tr>
<tr>
<td><code>&lt;cnc.b.motion_mode&gt;</code></td>
<td>B Axis Motion Mode:</td>
</tr>
<tr>
<td></td>
<td>0 = Continuous</td>
</tr>
<tr>
<td></td>
<td>1 = Indexing</td>
</tr>
<tr>
<td><code>&lt;cnc.b.convention&gt;</code></td>
<td>B Axis Convention:</td>
</tr>
<tr>
<td></td>
<td>0 = Normal</td>
</tr>
<tr>
<td></td>
<td>1 = Inverse</td>
</tr>
<tr>
<td><code>&lt;cnc.b.wrapped_rotary&gt;</code></td>
<td>B Axis Wrapped Rotary State:</td>
</tr>
<tr>
<td></td>
<td>0 = Disabled</td>
</tr>
<tr>
<td></td>
<td>1 = Enabled</td>
</tr>
<tr>
<td><code>&lt;cnc.b.parallel_to&gt;</code></td>
<td>B Axis Parallel to:</td>
</tr>
<tr>
<td></td>
<td>0 = X</td>
</tr>
<tr>
<td></td>
<td>1 = Y</td>
</tr>
<tr>
<td></td>
<td>2 = Z</td>
</tr>
<tr>
<td><code>&lt;cnc.b.origin_mode&gt;</code></td>
<td>B Axis Origin Mode:</td>
</tr>
<tr>
<td></td>
<td>0 = Custom</td>
</tr>
<tr>
<td></td>
<td>1 = WCS 1 - G54</td>
</tr>
<tr>
<td></td>
<td>2 = WCS 2 - G55</td>
</tr>
<tr>
<td></td>
<td>3 = WCS 3 - G56</td>
</tr>
<tr>
<td></td>
<td>4 = WCS 4 - G57</td>
</tr>
<tr>
<td></td>
<td>5 = WCS 5 - G58</td>
</tr>
<tr>
<td></td>
<td>6 = WCS 6 - G59</td>
</tr>
<tr>
<td></td>
<td>7 = WCS 7 - G59.1</td>
</tr>
<tr>
<td></td>
<td>8 = WCS 8 - G59.2</td>
</tr>
<tr>
<td></td>
<td>9 = WCS 9 - G59.3</td>
</tr>
<tr>
<td><code>&lt;cnc.b.origin_x&gt;</code></td>
<td>B Axis Origin X [mm]</td>
</tr>
<tr>
<td><code>&lt;cnc.b.origin_y&gt;</code></td>
<td>B Axis Origin Y [mm]</td>
</tr>
<tr>
<td><code>&lt;cnc.b.origin_z&gt;</code></td>
<td>B Axis Origin Z [mm]</td>
</tr>
<tr>
<td><code>&lt;cnc.c.enabled&gt;</code></td>
<td>C Axis Enabled State:</td>
</tr>
<tr>
<td></td>
<td>0 = Disabled</td>
</tr>
<tr>
<td></td>
<td>1 = Enabled</td>
</tr>
<tr>
<td><code>&lt;cnc.c.type&gt;</code></td>
<td>C Axis Axis Type:</td>
</tr>
<tr>
<td></td>
<td>0 = Linear Axis</td>
</tr>
<tr>
<td></td>
<td>1 = Rotary Axis Free</td>
</tr>
<tr>
<td></td>
<td>2 = Rotary Axis for Head</td>
</tr>
<tr>
<td></td>
<td>3 = Rotary Axis for Table</td>
</tr>
<tr>
<td><code>&lt;cnc.c.max_vel&gt;</code></td>
<td>C Axis Max Velocity [mm/min]</td>
</tr>
<tr>
<td>Named Variable</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>--------------------------------------------</td>
</tr>
<tr>
<td>#&lt;cnc.c.acc&gt;</td>
<td>C Axis Acceleration [mm/s²]</td>
</tr>
<tr>
<td>#&lt;cnc.c.min_lim&gt;</td>
<td>C Axis Min Limit [mm]</td>
</tr>
<tr>
<td>#&lt;cnc.c.max_lim&gt;</td>
<td>C Axis Max Limit [mm]</td>
</tr>
<tr>
<td>#&lt;cnc.c.motion_mode&gt;</td>
<td>C Axis Motion Mode: 0 = Continuous 1 = Indexing</td>
</tr>
<tr>
<td>#&lt;cnc.c.convention&gt;</td>
<td>C Axis Convention: 0 = Normal 1 = Inverse</td>
</tr>
<tr>
<td>#&lt;cnc.c.wrapped_rotary&gt;</td>
<td>C Axis Wrapped Rotary State: 0 = Disabled 1 = Enabled</td>
</tr>
<tr>
<td>#&lt;cnc.c.parallel_to&gt;</td>
<td>C Axis Parallel to: 0 = X 1 = Y 2 = Z</td>
</tr>
<tr>
<td>#&lt;cnc.c.origin_mode&gt;</td>
<td>C Axis Origin Mode: 0 = Custom 1 = WCS 1 - G54 2 = WCS 2 - G55 3 = WCS 3 - G56 4 = WCS 4 - G57 5 = WCS 5 - G58 6 = WCS 6 - G59 7 = WCS 7 - G59.1 8 = WCS 8 - G59.2 9 = WCS 9 - G59.3</td>
</tr>
<tr>
<td>#&lt;cnc.c.origin_x&gt;</td>
<td>C Axis Origin X [mm]</td>
</tr>
<tr>
<td>#&lt;cnc.c.origin_y&gt;</td>
<td>C Axis Origin Y [mm]</td>
</tr>
<tr>
<td>#&lt;cnc.c.origin_z&gt;</td>
<td>C Axis Origin Z [mm]</td>
</tr>
<tr>
<td>#&lt;cnc.u.enabled&gt;</td>
<td>U Axis Enabled State: 0 = Disabled 1 = Enabled</td>
</tr>
<tr>
<td>#&lt;cnc.u.type&gt;</td>
<td>U Axis Type: 4 = Gantry Axis for X 5 = Gantry Axis for Y 6 = Gantry Axis for Z</td>
</tr>
<tr>
<td>#&lt;cnc.u.max_vel&gt;</td>
<td>U Axis Max Velocity [mm/min]</td>
</tr>
<tr>
<td>#&lt;cnc.u.acc&gt;</td>
<td>U Axis Acceleration [mm/s²]</td>
</tr>
<tr>
<td>#&lt;cnc.u.min_lim&gt;</td>
<td>U Axis Min Limit [mm]</td>
</tr>
<tr>
<td>#&lt;cnc.u.max_lim&gt;</td>
<td>U Axis Max Limit [mm]</td>
</tr>
<tr>
<td>#&lt;cnc.v.enabled&gt;</td>
<td>V Axis Enabled State: 0 = Disabled 1 = Enabled</td>
</tr>
<tr>
<td>#&lt;cnc.v.type&gt;</td>
<td>V Axis Type: 4 = Gantry Axis for X 5 = Gantry Axis for Y 6 = Gantry Axis for Z</td>
</tr>
<tr>
<td>#&lt;cnc.v.max_vel&gt;</td>
<td>V Axis Max Velocity [mm/min]</td>
</tr>
<tr>
<td>#&lt;cnc.v.acc&gt;</td>
<td>V Axis Acceleration [mm/s²]</td>
</tr>
<tr>
<td>#&lt;cnc.v.min_lim&gt;</td>
<td>V Axis Min Limit [mm]</td>
</tr>
<tr>
<td>#&lt;cnc.v.max_lim&gt;</td>
<td>V Axis Max Limit [mm]</td>
</tr>
<tr>
<td>#&lt;cnc.w.enabled&gt;</td>
<td>W Axis Enabled State: 0 = Disabled 1 = Enabled</td>
</tr>
<tr>
<td>#&lt;cnc.w.type&gt;</td>
<td>W Axis Type: 4 = Gantry Axis for X 5 = Gantry Axis for Y 6 = Gantry Axis for Z</td>
</tr>
<tr>
<td>#&lt;cnc.w.max_vel&gt;</td>
<td>W Axis Max Velocity [mm/min]</td>
</tr>
<tr>
<td>#&lt;cnc.w.acc&gt;</td>
<td>W Axis Acceleration [mm/s²]</td>
</tr>
<tr>
<td>#&lt;cnc.w.min_lim&gt;</td>
<td>W Axis Min Limit [mm]</td>
</tr>
<tr>
<td>#&lt;cnc.w.max_lim&gt;</td>
<td>W Axis Max Limit [mm]</td>
</tr>
<tr>
<td>#&lt;cnc.spindle.max_speed&gt;</td>
<td>Spindle Max Speed [rpm]</td>
</tr>
</tbody>
</table>
### Named Variable

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#&lt;cnc.rotary_table.d_x&gt;</td>
<td>Rotary Table D:X [mm]</td>
</tr>
<tr>
<td>#&lt;cnc.rotary_table.d_y&gt;</td>
<td>Rotary Table D:Y [mm]</td>
</tr>
<tr>
<td>#&lt;cnc.rotary_table.d_z&gt;</td>
<td>Rotary Table D:Z [mm]</td>
</tr>
<tr>
<td>#&lt;cnc.tilting_spindle.h_x&gt;</td>
<td>Tilting Spindle H:X [mm]</td>
</tr>
<tr>
<td>#&lt;cnc.tilting_spindle.h_y&gt;</td>
<td>Tilting Spindle H:Y [mm]</td>
</tr>
<tr>
<td>#&lt;cnc.tilting_spindle.h_z&gt;</td>
<td>Tilting Spindle H:Z [mm]</td>
</tr>
<tr>
<td>#&lt;cnc.tilting_spindle.j_x&gt;</td>
<td>Tilting Spindle J:X [mm]</td>
</tr>
<tr>
<td>#&lt;cnc.tilting_spindle.j_y&gt;</td>
<td>Tilting Spindle J:Y [mm]</td>
</tr>
<tr>
<td>#&lt;cnc.tilting_spindle.j_z&gt;</td>
<td>Tilting Spindle J:Z [mm]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#&lt;tool.type.generic&gt;</td>
<td>Tool Type: Generic (0)</td>
</tr>
<tr>
<td>#&lt;tool.type.flat_end_mill&gt;</td>
<td>Tool Type: Flat End Mill (1)</td>
</tr>
<tr>
<td>#&lt;tool.type.ball_nose_end_mill&gt;</td>
<td>Tool Type: Ball Nose End Mill (2)</td>
</tr>
<tr>
<td>#&lt;tool.type.drill&gt;</td>
<td>Tool Type: Drill (3)</td>
</tr>
<tr>
<td>#&lt;tool.type.saw&gt;</td>
<td>Tool Type: Saw (5)</td>
</tr>
<tr>
<td>#&lt;tool.type.plasma&gt;</td>
<td>Tool Type: Plasma (6)</td>
</tr>
<tr>
<td>#&lt;tool.type.drag_knife&gt;</td>
<td>Tool Type: Drag Knife (7)</td>
</tr>
<tr>
<td>#&lt;tool.type.lathe&gt;</td>
<td>Tool Type: Lathe (8)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#&lt;probe.state.succeed&gt;</td>
<td>Probe State: Succeeded (0). Probe state is available at #5700.</td>
</tr>
<tr>
<td>#&lt;probe.state.not_tripped&gt;</td>
<td>Probe State: Not Tripped (-1). Probe state is available at #5700.</td>
</tr>
<tr>
<td>#&lt;probe.state.already_tripped&gt;</td>
<td>Probe State: Already Tripped (-2). Probe state is available at #5700.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>#&lt;wait_input.low&gt;</td>
<td>M66 L parameter: Waits for the selected input to reach the LOW state (0).</td>
</tr>
<tr>
<td>#&lt;wait_input.high&gt;</td>
<td>M66 L parameter: Waits for the selected input to reach the HIGH state (1).</td>
</tr>
<tr>
<td>#&lt;wait_input.fail&gt;</td>
<td>M66 L parameter: Waits for the selected input to perform a FALL event (2).</td>
</tr>
<tr>
<td>#&lt;wait_input.rise&gt;</td>
<td>M66 L parameter: Waits for the selected input to perform a RISE event (3).</td>
</tr>
<tr>
<td>#&lt;wait_input.immediate&gt;</td>
<td>M66 L parameter: Return immediately and the input value is stored in #5720.</td>
</tr>
<tr>
<td>#&lt;wait_input.alarm_low&gt;</td>
<td>M66 L parameter: Waits for the selected input to reach the LOW state and generate a CNC alarm if timeout elapses while waiting (10).</td>
</tr>
<tr>
<td>#&lt;wait_input.alarm_high&gt;</td>
<td>M66 L parameter: Waits for the selected input to reach the HIGH state and generate a CNC alarm if timeout elapses while waiting (11).</td>
</tr>
<tr>
<td>#&lt;wait_input.alarm_fail&gt;</td>
<td>M66 L parameter: Waits for the selected input to perform a FALL event and generate a CNC alarm if timeout elapses while waiting (12).</td>
</tr>
<tr>
<td>#&lt;wait_input.alarm_rise&gt;</td>
<td>M66 L parameter: Waits for the selected input to perform a RISE event and generate a CNC alarm if timeout elapses while waiting (13).</td>
</tr>
<tr>
<td>#&lt;wait_input.success&gt;</td>
<td>Status of the last M66 in #5722: “Wait Input” operation ended with success state (0).</td>
</tr>
<tr>
<td>#&lt;wait_input.failure&gt;</td>
<td>Status of the last M66 in #5722: “Wait Input” operation ended with failure state (1).</td>
</tr>
</tbody>
</table>

### 2.2.4 Examples

```gcode
G17 G21 G40 G49 G80 G90
F1000
#<variable1> = 123
#<variable2> = 456
IF [#<variable1> EQ 123] THEN1
   G1 X=#<variable1>
END1

CALL the subroutine O1000
M98 P1000
G1 X=#<variable1>
G1 Y=#<variable2>
IF EXISTS [#<variable2>] THEN2
   M109 P Named parameter _variable2 exists and its value is #<_variable2>.
END2

CALL the subroutine O1002 as a macro to pass the arguments A and B
G65 P10002 A0 B0
   Named variables indexing support.
   P1 = 8
   #< index > = 1
   #< array >[#<index>] = 12
   #< array >[#<index>2] = 34
```
2.3 Position Information

<table>
<thead>
<tr>
<th>Variable Number</th>
<th>Description</th>
<th>Coordinate System</th>
<th>Tool compensation offset</th>
<th>Update</th>
</tr>
</thead>
<tbody>
<tr>
<td>5001 - 5006</td>
<td>Current position</td>
<td>Workpiece offset</td>
<td>Included</td>
<td>Updated during motion</td>
</tr>
<tr>
<td>5701 - 5706</td>
<td>Probed position</td>
<td>Workpiece offset</td>
<td>Not included</td>
<td>At the end of a G38 X</td>
</tr>
<tr>
<td>5711 - 5716</td>
<td>Probed position</td>
<td>Machine coordinate system</td>
<td>Not included</td>
<td>At the end of a G38 X</td>
</tr>
</tbody>
</table>

- Each range of variable numbers is for 1 to 6 axes. The first number is for the X-axis the second number is for the Y-axis and so on up to the C axis.

2.4 Vacant or Empty Variables

In many cases, a variable may also be undefined. In this case, the variable is set to #0, which identifies a null variable (empty / not initialized). Indeed #0 is a read-only variable used mainly for two purposes:

- check if a variable has been initialized
- reset a variable

At the beginning of the compilation every non system variable is set to #0.

A null variable has no value, it should not be confused with a variable that has a zero value.

| #101 = 0       | Variable #101 has a zero value                      |
| #102 = #0      | Variable #102 is vacant (empty), has no value and cannot be used for some operations |

The following piece of G-code provide an example of the operations that can be used with an empty variable:

```
; A null variable
#1 = #0          ; Define #1 as null (that means #1 is empty/not initialized)
IF [#1] EQ 0      ; #1 is defined as null (that means #1 is vac
THEN M109 P"IF[#1] EQ 0 Should return TRUE" ; Should return TRUE
IF [#1] NE #0     ; #1 is defined as null (that means #1 is vac
THEN M109 P"IF[#1] NE 0 Should return FALSE"
IF [#1] GT #0     ; #1 is defined as null (that means #1 is vac
THEN M109 P"IF[#1] GT 0 Should return FALSE"
IF [#1] GE #0     ; #1 is defined as null (that means #1 is vac
THEN M109 P"IF[#1] GE 0 Should return TRUE"
IF [#1] LT #0     ; #1 is defined as null (that means #1 is vac
THEN M109 P"IF[#1] LT 0 Should return FALSE"
IF [#1] LE #0     ; #1 is defined as null (that means #1 is vac
THEN M109 P"IF[#1] LE 0 Should return TRUE"

; A zero variable
#1 = 0           ; Define #1 as zero that means #1 is equal to 0
IF [#1] EQ 0      ; #1 is defined as null (that means #1 is vac
THEN M109 P"IF[#1] EQ 0 Should return FALSE"
IF [#1] NE #0     ; #1 is defined as null (that means #1 is vac
THEN M109 P"IF[#1] NE 0 Should return FALSE"
IF [#1] GT #0     ; #1 is defined as null (that means #1 is vac
THEN M109 P"IF[#1] GT 0 Should return FALSE"
IF [#1] GE #0     ; #1 is defined as null (that means #1 is vac
THEN M109 P"IF[#1] GE 0 Should return TRUE"
IF [#1] LT #0     ; #1 is defined as null (that means #1 is vac
THEN M109 P"IF[#1] LT 0 Should return FALSE"
IF [#1] LE #0     ; #1 is defined as null (that means #1 is vac
THEN M109 P"IF[#1] LE 0 Should return TRUE"
```
RosettaCNC G-code language

IF[#1 NE 0] THEN M109 P"IF[#1 NE 0] Should return TRUE"
IF[#1 GT 0] THEN M109 P"IF[#1 GT 0] Should return FALSE"
IF[#1 LT 0] THEN M109 P"IF[#1 LT 0] Should return FALSE"
IF[#1 LE 0] THEN M109 P"IF[#1 LE 0] Should return TRUE"

; Comparing a zero to a zero:
; #1 is defined as zero (that means #1 is equal to 0)
IF[#1 EQ 0] THEN M109 P"IF[#1 EQ 0] Should return TRUE"
IF[#1 NE 0] THEN M109 P"IF[#1 NE 0] Should return FALSE"
IF[#1 GT 0] THEN M109 P"IF[#1 GT 0] Should return FALSE"
IF[#1 LE 0] THEN M109 P"IF[#1 LE 0] Should return FALSE"

; Comparing a positive number to a null variable:
IF[#1 EQ #0] THEN M109 P"IF[#1 EQ #0] Should return FALSE"
IF[#1 NE #0] THEN M109 P"IF[#1 NE #0] Should return TRUE"
IF[#1 GT #0] THEN M109 P"IF[#1 GT #0] Should return TRUE"
IF[#1 GE #0] THEN M109 P"IF[#1 GE #0] Should return TRUE"
IF[#1 LT #0] THEN M109 P"IF[#1 LT #0] Should return FALSE"
IF[#1 LE #0] THEN M109 P"IF[#1 LE #0] Should return FALSE"

; Comparing a negative number to a null variable:
IF[#1 EQ #0] THEN M109 P"IF[#1 EQ #0] Should return FALSE"
IF[#1 NE #0] THEN M109 P"IF[#1 NE #0] Should return TRUE"
IF[#1 GT #0] THEN M109 P"IF[#1 GT #0] Should return FALSE"
IF[#1 GE #0] THEN M109 P"IF[#1 GE #0] Should return FALSE"
IF[#1 LT #0] THEN M109 P"IF[#1 LT #0] Should return TRUE"
IF[#1 LE #0] THEN M109 P"IF[#1 LE #0] Should return TRUE"

m2 ( Program End)
2.5 Local variables

The local variables #1..#33 are kept in what are called “levels”. When G65 or G66 is called:

1. the current values of all those locals are copied to a level
2. any word used when calling G65 and G66 are transferred into the local variables.

The following table shows how the words are mapped to local variables:

<table>
<thead>
<tr>
<th>Argument List</th>
<th>Local Variable in a Macro</th>
<th>Named Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>#1</td>
<td>#&lt;_args.a&gt;</td>
</tr>
<tr>
<td>B</td>
<td>#2</td>
<td>#&lt;_args.b&gt;</td>
</tr>
<tr>
<td>C</td>
<td>#3</td>
<td>#&lt;_args.c&gt;</td>
</tr>
<tr>
<td>D</td>
<td>#7</td>
<td>#&lt;_args.d&gt;</td>
</tr>
<tr>
<td>E</td>
<td>#8</td>
<td>#&lt;_args.e&gt;</td>
</tr>
<tr>
<td>F</td>
<td>#9</td>
<td>#&lt;_args.f&gt;</td>
</tr>
<tr>
<td>H</td>
<td>#11</td>
<td>#&lt;_args.h&gt;</td>
</tr>
<tr>
<td>I</td>
<td>#4</td>
<td>#&lt;_args.i&gt;</td>
</tr>
<tr>
<td>J</td>
<td>#5</td>
<td>#&lt;_args.j&gt;</td>
</tr>
<tr>
<td>K</td>
<td>#6</td>
<td>#&lt;_args.k&gt;</td>
</tr>
<tr>
<td>M</td>
<td>#13</td>
<td>#&lt;_args.m&gt;</td>
</tr>
<tr>
<td>Q</td>
<td>#17</td>
<td>#&lt;_args.q&gt;</td>
</tr>
<tr>
<td>R</td>
<td>#18</td>
<td>#&lt;_args.r&gt;</td>
</tr>
<tr>
<td>S</td>
<td>#19</td>
<td>#&lt;_args.s&gt;</td>
</tr>
<tr>
<td>T</td>
<td>#20</td>
<td>#&lt;_args.t&gt;</td>
</tr>
<tr>
<td>U</td>
<td>#21</td>
<td>#&lt;_args.u&gt;</td>
</tr>
<tr>
<td>V</td>
<td>#22</td>
<td>#&lt;_args.v&gt;</td>
</tr>
<tr>
<td>W</td>
<td>#23</td>
<td>#&lt;_args.w&gt;</td>
</tr>
<tr>
<td>X</td>
<td>#24</td>
<td>#&lt;_args.x&gt;</td>
</tr>
<tr>
<td>Y</td>
<td>#25</td>
<td>#&lt;_args.y&gt;</td>
</tr>
<tr>
<td>Z</td>
<td>#26</td>
<td>#&lt;_args.z&gt;</td>
</tr>
</tbody>
</table>
3. Macro programming

Rosetta CNC supports Macro programming (following Fanuc Macro B style).

Your G-code programs or sub-programs can include a few non G-code commands that use:

- Variables
- Arithmetic Logic & Statements
- Subroutines
- Custom Macro calls (subroutines with arguments)
- Looping & Branching

Subroutines, Macros and WHILE statements can be nested up to 100 times.

The following example provides an overview of the supported macro programming features:

```
F5000
G0 x0 y0 z0
G1 x100
; G-code subroutines support
; To invoke a subroutine type "M98 P<subroutine id> L<repetitions>".
; The following line invokes the subroutine with id 1 for 2 times.
M98 P1 L2
; Named external G-code subroutine call
M98 P"named_sub.ngc" L2

#1 = 10
#2 = 20
#3 = 30
#4 = -10
; Macro call support:
; - The arguments are loaded in the correspondent parameters.
; - When the call is finished the parameters 1-33 are restored to the value
;   they had before the call
A = #1
B = #2
C = #3
G65 P2 A3 B5 C2
IF [#4 EQ -10] THEN M109 P"Parameter 4 is restored to #4 when the macro is finished"
G65 P"named_sub.ngc" A3 B5 C2
"CALL" can be used as a more readable alias for G65.
CALL P"local_named_sub"
M2 (Program End)
```

RosettaCNC supports G-code subroutines:

```
O1 ( Subroutine body that contains G-code instructions )
G0 x0 y0 z0
G0 x0 y0 z50
G0 x0 y0 z0
M99
O2 ( Subroutine body can access and modify local parameters )
A4 = ([#1 + #2] x #3)
IF [#4 EQ #0] THEN M109 P"Parameter 5 has not been set when the macro has been called"
IF [#4 EQ 16] THEN M109 P"Parameter 4 is equal to #4"
M99
```

Local named subroutines can be defined as follows, where:
- "SUB" and "O" can be both used for the subroutine declaration
- "ENDSUB" and "M99" can be both used for subroutine end
- "RETURN" and "M99" can be both used to return from a subroutine

```
SUB "local_named_sub"
G1 Y100
IF [#1 EQ 0] THEN
  RETURN
END1
ENDSUB
```
3.1 Arithmetic Logic & Statements

3.1.1 Binary Operators

Binary operators only appear inside expressions.

There are:

- **Mathematical operations**: addition (+), subtraction (-), multiplication (*), and division (/), modulus operation (MOD) and power operation (**)
- **Logical operations**: non-exclusive or (OR), exclusive or (XOR), and logical and (AND)
- **Relational operators**: equality (EQ), inequality (NE), strictly greater than (GT), greater than or equal to (GE), strictly less than (LT), and less than or equal to (LE)

Their precedence is defined accordingly to the following table:

<table>
<thead>
<tr>
<th>Operators</th>
<th>Precedence</th>
</tr>
</thead>
<tbody>
<tr>
<td>**</td>
<td>Highest</td>
</tr>
<tr>
<td>* / MOD</td>
<td></td>
</tr>
<tr>
<td>+ -</td>
<td></td>
</tr>
<tr>
<td>EQ NE GT GE LT LE</td>
<td></td>
</tr>
<tr>
<td>AND OR XOR</td>
<td>Lowest</td>
</tr>
</tbody>
</table>

**About equality and floating-point values**

The RS274/NGC language only supports floating-point values of finite precision. The interpreter considers values equal if their absolute difference is less than **0.0001**.

3.1.2 Functions

The following table shows the available functions.

Unary operations arguments which take angle measures (COS, SIN, and TAN) are in degrees.

Values returned by unary operations which return angle measures (ACOS, ASIN, and ATAN) are also in degrees.

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATAN[arg]/[arg]</td>
<td>Four quadrant inverse tangent.</td>
</tr>
<tr>
<td>ABS[arg]</td>
<td>Absolute value.</td>
</tr>
<tr>
<td>ACOS[arg]</td>
<td>Inverse cosine.</td>
</tr>
<tr>
<td>ASIN[arg]</td>
<td>Inverse sine.</td>
</tr>
<tr>
<td>COS[arg]</td>
<td>Cosine.</td>
</tr>
<tr>
<td>EXP[arg]</td>
<td>(e^{\text{arg}}) raised to the given power.</td>
</tr>
<tr>
<td>FIX[arg]</td>
<td>Round down to integer.</td>
</tr>
<tr>
<td>FUP[arg]</td>
<td>Round up to integer.</td>
</tr>
<tr>
<td>ROUND[arg]</td>
<td>Round to nearest integer.</td>
</tr>
<tr>
<td>LN[arg]</td>
<td>Base-e logarithm.</td>
</tr>
<tr>
<td>SIN[arg]</td>
<td>Sine.</td>
</tr>
<tr>
<td>SQRT[arg]</td>
<td>Square Root.</td>
</tr>
<tr>
<td>EXISTS[arg]</td>
<td>Check if a named parameter exists. Returns 1 if it exists otherwise returns 0.</td>
</tr>
</tbody>
</table>
3.2 Loopying & Branching

3.2.1 Unconditional Branching

<table>
<thead>
<tr>
<th>GOTOon</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
</tr>
<tr>
<td>GOTO10</td>
</tr>
<tr>
<td>...</td>
</tr>
<tr>
<td>N10</td>
</tr>
<tr>
<td>...</td>
</tr>
<tr>
<td>; Code that will be executed</td>
</tr>
</tbody>
</table>

3.2.2 Conditional Branching

<table>
<thead>
<tr>
<th>IF</th>
<th>CONDITION IS TRUE</th>
<th>GOTOon</th>
</tr>
</thead>
<tbody>
<tr>
<td>IF</td>
<td>[#7 LT 0]</td>
<td>GOTO65</td>
</tr>
<tr>
<td>; Code execution will jump to the line that starts with the label N65</td>
<td></td>
<td></td>
</tr>
<tr>
<td>N65</td>
<td>; Target block of the IF conditional statement</td>
<td></td>
</tr>
</tbody>
</table>

3.2.3 IF-THEN Option

Two different IF-THEN options are supported:

**Single line syntax**

```
IF [ condition is true ] THEN [ argument ]
```

**Multi line syntax**

```
IF [ ... Condition 1 is true ... ] THEN1 ; Start of IF block 1
... <... Body of block 1 - Part 1 ...>
... IF [ ... Condition 2 is true ... ] THEN2 ; Start of IF block 2
... <... Body of block 2 ...>
... END2 ; End of IF block 2
... <... Body of block 1 - Part 2 ...>
ELSE1 ... <... Body of block 1 - else ...>
END1 ; End of IF block 1
```

**Multi line syntax with ELIF**

```
IF [ ... Condition 1 is true ... ] THEN1 ; Start of IF block 1
... <... Body of block 1 - Part 1 ...>
... IF [ ... Condition 2 is true ... ] THEN2 ; Start of IF block 2
... <... Body of block 2 ...>
... END2 ; End of IF block 2
... <... Body of block 1 - Part 2 ...>
ELIF [ ... Condition 3 is true ... ] THEN1
... <... Body of block 3 - elseif ...>
ELIF [ ... Condition 4 is true ... ] THEN1
... <... Body of block 4 - elseif ...>
ELSE1
... <... Body of block 5 - else ...>
END1 ; End of IF block 1
```

3.2.4 While Loop

```
WHILE [condition] DOn
```

While loops can be nested as follows

```
WHILE [ ... Condition 1 is true ... ] D01 ; Start of WHILE loop 1
... 
```
To exit a while loop \texttt{BREAKn} can be used.

\texttt{WHILE [...] Condition 1 is true [...] DO1 ; Start of WHILE loop 1 } \\
\texttt{... ; End of WHILE loop 1 }
\texttt{END1 ; End of WHILE loop 1 }

\textbf{3.2.5 Example}
3.3 Custom Macro calls

Custom macros are direct extensions of subroutines and subprograms.

As subroutines and subprograms:

- they can be defined into the same file where they are called
- they can be defined into an external file named O<subroutine number>.ngc
- they can be defined into an external file with a custom name
- they are ended with M99 or ENDSUB
- you can return from them before the end calling M99 or RETURN

Differently from subroutines and subprograms:

- they are called using the Gcode G65 or CALL instead of the M code M98
- they support arguments

3.3.1 Non modal Macro calls

Non modal macro calls are initiated with G65 G-Code instead of M98 G-Code and they provide what are called “arguments”. The values of the “arguments” are transferred to the Local Variables. If an argument is not set the correspondent variable will be initialised to Null.

```gcode
; Main program
T5 M6
G0 G90 G40 G17 G94 G80
G65 P1000 A3 B2 C0.01
M30

G1 x#1 y#2 z#3
M99
```

3.3.2 Modal Macro calls

Modal macro calls are similar to non modal Macro calls but once the macro is enabled using G66 it will be executed automatically every time an X, Y, Z, A, B, C motion is programmed.

Modal macro calls can be used to obtain customised cycles. Indeed instead of performing the standard G81, G82, ... cycles the user can specify all the operations performed by the cycle.

With parameter G10 L100 P5800 V<value> the motion mode used to perform the motion between two macro repetitions can be specified (#5800 = 0 → G0; #5800 = 1 → G1).

While a modal macro is active the parameter #5148 store the information about how many times the macro has been called.

Example
G90
G0
X0 Y0 Z0 F5000
G1
X100.0
; Enable the modal macro defined in O9110.ngc
G66 P9110 W5 Z10.0 F500
Y20.0
Y40.0
G67
; Disable the macro
M30

G00
G91
Z#26
(Stores G00/G01)

G99_P18
G0 Z#5
(G03 G04 F#4) (Restores modal information)
M99

G10
G1 X#23
Y#23
Z#23
M99

G91 Z#26 F#9
G01 Z#18
(Stores the cutting feedrate)
(G01 Z#26 (Positioning at position delta R)

G00 G91 Z#18
(Stores Z coordinate at the start of drilling)

G91 G90 Z#26
(Stores G90/G91)

G01 Z#26 F#9
G01 Z#18
(Positions at position R)

IF[#110 EQ 98] GOTO O10
(Return to position I)

G00 Z#18
(Positions at position R)

GOTO 2

N1 G90 Z#5
(Positions at position I)

N2 G01 G03 F#4
(Restores Modal information)
M99
3.4 Subroutines

To make g-code convenient for re-use, you can centralise it and then access it from the main program. G-Code language provides two different methods for doing this: subprogram/subroutines calls and macro calls.

With Rosetta CNC you can:

- call a subroutine defined in the same G-Code file by number. Example: M98 P100 L10
- call a subprogram defined into an external file by number. Example: M98 P100 L10 where in the macro folder you have defined a file 0100.ngc.
- call a subprogram defined into an external file by name. Example: M98 P“named_sub.ngc” L10

Parameters:

- P: identifies the subroutine/subprogram to be called. It can be a number or a string with the name of the external file.
- L: identifies how many times the G-Code commands inside the subroutine should be repeated before returning.

```gcode
; © 2019 by RosettaCNC Motion
; ( file name: subroutines.ngc )

F5000
G0 x0 y0 z0
G1 x100

( RosettaCNC supports G-code subroutines)
( To call a subroutine type “M98 P<subroutine id> L<repetitions>”.)
( The following line calls the subroutine with id 1 for 2 times. )
M98 P1 L2

( RosettaCNC supports G-code numbered external subroutines. )
( The subprogram 0101.ngc should exist in the macro folder. )
M98 P101 L2

( RosettaCNC supports G-code named external subroutines )
M98 P“named_sub.ngc” L2

m2 ( Program End )

( The following lines declare a subroutine with id 1 )
01 ( Subroutine body that contains G-code instructions )
G0 x0 y0 z0
G8 x0 y0 z50
G0 x0 y0 z0

( The following line marks the end of the subroutine )
M99
```
3.5 User Tool Change Subprogram

Setting the Tool Change Type option to Custom Macro in the board settings the user can customise the tool change procedure. If the option is enabled the M6 command will look into the machine macro folder and execute the G-code file named tool_change.ngc. In this file the user can specify any supported G-code command to perform the tool change procedure as required by the specific machine.

When the tool_change.ngc is used it could be useful to consider using 4 M codes:

- M61: Set the current tool
- M106: Execute PLC internal tool change procedure
- M107: Inform the PLC that the following commands are part of the user tool change macro. This is used for visualisation purposes and for axes limits checking.
- M108: Inform the PLC that the user tool change macro has ended. This is used for visualisation purposes and for axes limits checking.

Examples

Manual Tool Change

```gcode
; (c) 2016-2019 by RosettaCNC Motion
; User defined tool change subprogram:
; will be called when M6 is called if the parameter “Tool Change Type”
; is set to one of the custom macro modes.
; Arguments
; =========
; #1 : tool id of the tool to be loaded (corresponds to #5132)
; #2 : slot of the tool to be loaded (corresponds to #5133)
; #3 : tool id of the tool in use (corresponds to #5134)
; #4 : slot of the tool in use (corresponds to #5135)
; Skip tool change if the tool to be loaded is already loaded
IF [#1 EQ #3] THEN M99
; Store actual the G code of the modal group 1: G0, G1, ...
#4101=#5101
#4151=#5151
#4153=#5153
#4154=#5154
; Store current positions (X, Y, Z, A, B, C)
#4001=#5001
#4002=#5002
#4003=#5003
#4004=#5004
#4005=#5005
#4006=#5006
; Disable spindle, flood & mist
M5 M9
; Move upwards to a “safe position”
G53 G0 Z200
; Move to tool change position
G53 G0 X-100 Y-100
; Display the message to the user
M109 P"Insert tool T#1" Q2
G4 P1
; Call M61 or M106:
; - call M106 to use part of the RosettaCNC internal tool change procedure
; - call M61 if you have handled the tool change procedure entirely in your code
; to inform RosettaCNC that the new tool has been loaded
M61 Q#1
; Move back above the original position (keep Z to a “safety position”)
G0 X#4001 Y#4002
; Restore previous states
IF [#4151 EQ 3] THEN m3
IF [#4151 EQ 4] THEN m4
IF [#4151 EQ 7] THEN m7
IF [#4154 EQ 8] THEN m8
; Enable tool offset compensation
G43 H#1
; Move Z back to the original position
G1 Z#4003
G4101
M99
```

Automatic Tool Change

```gcode
; (c) 2016-2019 by RosettaCNC Motion
; User defined tool change subprogram:
; will be called when M6 is called if the parameter “Tool Change Type”
; is set to one of the custom macro modes.
; Arguments
; =========
; #1 : tool id of the tool to be loaded (corresponds to #5132)
; #2 : slot of the tool to be loaded (corresponds to #5133)
; #3 : tool id of the tool in use (corresponds to #5134)
; #4 : slot of the tool in use (corresponds to #5135)
; Skip tool change if the tool to be loaded is already loaded
```
Manual Tool Change with tool length compensation

; User defined tool change subroutine:
; will be called when M6 is called if the parameter "Tool Change Type"
; is set to one of the custom macro modes.
; Call M61 or M106:
; - call M106 to use part of the RosettaCNC internal tool change procedure
; - call M61 if you have handled the tool change procedure entirely in your code
; to inform RosettaCNC that the the new tool has been loaded

IF [#1 EQ #3] THEN M99
       ; Store actual the G code of the modal group 1: G0, G1, ...
       [#4101]=#5101
IF [#1 EQ #3] THEN M99
       ; Store actual state M3, M4, M5
       [#4101]=#5111
IF [#1 EQ #3] THEN M99
       ; Store actual state M7, M9
       [#4101]=#5113
IF [#1 EQ #3] THEN M99
       ; Store current positions (X, Y, Z, A, B, C)
       [#4001]=#5001
       [#4002]=#5002
       [#4003]=#5003
       [#4004]=#5004
       [#4005]=#5005
       [#4006]=#5006
   ; Disable spindle, flood & mist
   M5 M9
       ; Move upwards to a "safe position"
   G53 G0 Z200
   ; Move to intermediate tool change position
   G53 G0 X0 Y0
       ; Move to a position that depends on the active slot to drop the current tool
   IF [#1 EQ #3] THEN G53 G1 X-10 Y9
   IF [#1 EQ #3] THEN G53 G1 X-10 Y10
   IF [#1 EQ #3] THEN G53 G1 X-10 Y20
   IF [#1 EQ #3] THEN G53 G1 X-10 Y30
       ; Set aux1 output to drop the current tool
   M62
   ; Inform PLC that the tool has been dropped
   ; Call M61 or M106:
   ; - call M106 to use part of the RosettaCNC internal tool change procedure
   ; - call M61 if you have handled the tool change procedure entirely in your code
   ; to inform RosettaCNC that the new tool has been loaded
   M61 Q1
       ; Move back above the original position (keep Z to a "safety position")
   G0 X=#4001 Y=#4002
   ; Restore previous states
   IF [#4151] EQ 3 THEN M3
   IF [#4151] EQ 4 THEN M4
   IF [#4151] EQ 7 THEN M7
   IF [#4151] EQ 8 THEN M8
       ; Enable tool offset compensation
   G43 M1
       ; Move Z back to the original position
   G1 Z=#4101
   G4001
   M9

(c) 2016-2019 by RosettaCNC Motion

; Store tool id of the tool to be loaded (corresponds to #5132)
; #2 : slot of the tool to be loaded (corresponds to #5133)
; #3 : tool id of the tool in use (corresponds to #5134)
; #4 : tool id of the tool in use (corresponds to #5135)

; System parameters
; #6002: system parameter: Set to 1 if the tool change should be skipped if the same tool is already loaded.
; #6003: system parameter: Feed during tool change procedure
; #6010: system parameter: 0: perform tool change; 1: perform tool change and enable tool compensation from table;
; 2: perform tool change, measure tool length and apply it
; #6011: system parameter: Tool change position X
; #6012: system parameter: Tool change position Y
; #6013: system parameter: Tool change position Z
; #6014: system parameter: Feed fast approaching sensor
; #6015: system parameter: Feed slow approaching sensor
; #6021: system parameter: Approaching sensor target Z (reached with fast feed #6021)
; #6022: system parameter: Sensor Z BCS position (reached with slow feed #6022)
; #6031: system parameter: Pre-measuring position X
; #6032: system parameter: Pre-measuring position Y
; #6033: system parameter: Pre-measuring position Z

IF [#6002] GE 1 THEN10
   ; Skip tool change if the tool to be loaded is already loaded
   IF [#1 EQ #3] THEN M99
END10
IF [#6002] EQ 0 OR [#6003] EQ 0 OR [#6010] EQ 0 OR [#6011] EQ 0 OR [#6012] EQ 0 OR [#6013] EQ 0 THEN10
   M109 P"One or more compulsory system parameters are not specified." Q1
END10
IF [#6021] EQ 0 OR [#6022] EQ 0 OR [#6031] EQ 0 OR [#6032] EQ 0 OR [#6033] EQ 0 THEN10
   M109 P"One or more compulsory system parameters are not specified." Q1
END10
The following is a usage example in RosettaCNC G-code language:

```gcode
; Store the tool data.
G44 P5101
G44 P5139
G44 P5151
G44 P5153
G44 P5154
G44 P5155
G44 P5156
; Restore the current position (X, Y, Z, A, B, C)
G403 P5001
G403 P5002
G403 P5003
G403 P5004
G403 P5005
G403 P5006
; Measure tool offset Z
G401 P5422
; Inform the PLC that the tool change procedure is starting.
M107
M109 P"Tool change procedure" Q4
; Disable spindle, flood & mist.
M5 M6
G6003
; Move upwards to a "safe position".
G53 G1 Z48013
; Move to tool change position.
G53 G1 X6001 Y60012
IF [#1 NE #3] THEN10
; A different tool should be inserted. Display the message to the user.
M109 P"Insert tool #1 and press OK" Q2
G4 P1
M651 Q1
END10
IF [#1 = #6] THEN10
; Check if a G38.X is used. In that case convert to a G80 to prevent an error when setting G#4101.
IF [#6105 CE 38] AND [#5110 LT 39] THEN10
M101 = #6
END10
IF [#6106 EQ 2] THEN10
; Measure tool offset.
M109 P"Check tool length procedure" Q4
G53 G1 Z48013
G53 G1 X6831 Y68012
G53 F6002 G30.3 Z4803
IF [#5708 EQ -1] THEN10 M109 P"Error updating piece position: Sensor already tripped!" Q1 M108
; Sensor should not be triggered during the fast approach.
IF [#5708 EQ -1] THEN M109 P"Error during tool measuring procedure: Sensor tripped during fast approach!" Q1 M108
G53 F6002 G30.3 Z4803
IF [#5708 NE 1] THEN10 M109 P"Error updating piece position: Sensor already tripped!" Q1 M108
G53 F6002 G30.3 Z4803
; Calculate the tool length offset as the difference between when the sensor was tripped and the sensor Z position.
M1401 = [#5130 - #6003]
M1402 = [#5151 - #6003]
; Move upwards to a "safe position".
G60030
G53 G1 Z48013
M109 P"Check tool length procedure has ended. Detected length is #4010." Q4
END10
; Move back above the original position (keep Z to a "safety position").
G1 X48001 Y48002
IF [#6109 EQ 1] THEN10
; Enable tool offset compensation considering the tool offset written in the tool table.
G43 H9
END10
IF [#6109 EQ 2] THEN10
; Enable tool length compensation.
G43.1 Z4810
END10
; Restore previous states.
IF [#4141 EQ 1] THEN M3
IF [#4141 EQ 2] THEN M4
IF [#4141 EQ 3] THEN M5
IF [#4141 EQ 4] THEN M6
END10
; Move Z back to the original position.
G1 Z48003
; Restore original target feed.
F4810
; Restore original G code if it is possible (G0 or G1 or G2 or G3 or ...).
G4810
; Reset HUD message.
M109 P"" Q4
; Inform the PLC that the tool change procedure has ended.
M108
M98
```

Usage example:

- G54 G49 F1000: Restart G54’s offsets.
- G10 L2 P1 X8 Y8 Z0: Ensure that the current position can be reached when the tool compensation is activated.
- G0 X8 Y0 Z0: Tool length compensation is enabled if parameter #6010 is set to 1 or 2.
- M109 P"Compensated tool length is #5422" Q4
- G100
4. Cutter compensation

The cutter radius compensation capabilities of the Interpreter enable the programmer to specify that a cutter should travel to the right or left of an open or closed contour composed of arcs of circles and straight line segments, all planes are supported XY, YZ and XZ.

4.1 G41, G42 Cutter Compensation

- G41 <D> <I> left of programmed path
- G42 <D> <I> right of programmed path

**Notes**

- D - tool number
- I - dynamic radius offset
- The D word is optional; if there is no D word the radius of the currently loaded tool will be used (if no tool is loaded and no D word is given, a radius of 0 will be used).
- The I word is optional; if there is I word the resulting radius will be: “table diameter value” / 2 - “I value”.
- If supplied, the D word is the tool number to use. This would normally be the number of the tool in the spindle (in which case the D word is redundant and need not be supplied), but it may be any valid tool number.
- It is an error if:
  - The D number is not a valid tool number or 0.
  - Cutter compensation is commanded to turn on when it is already on.

4.2 G41.1, G42.1 Dynamic Cutter Compensation

- G41.1 D <L> (left of programmed path)
- G42.1 D <L> (right of programmed path)

**Notes**

- D - cutter diameter
- L - tool orientation (see lathe tool orientation)

G41.1 & G42.1 function the same as G41 & G42 with the added scope of being able to program the tool diameter. The L word defaults to 0 if unspecified.

- It is an error if:
• The YZ plane is active.
• The L number is not in the range from 0 to 9 inclusive.
• The L number is used when the XZ plane is not active.
• Cutter compensation is commanded to turn on when it is already on.

4.3 Tool compensation entry options

Two compensation entry option are supported:

• **Default**: the default NIST mode
• **Easy lead-in**: delays the first movement of the entry move until the following line or arc is specified. When the second entry motion is specified it performs the entry move to a position tangent to the beginning of the second line or arc. This option works well with some CADs such as Fusion360 indeed not gouging errors are generated.
• **Auto**: its the only options that can handle concave corners and concave arcs.

The **Easy lead-in** option is needed when a concave corner is present between the first and the second move that are part of the Lead-in.

Common Cutter Radius Compensation Errors

When the first two options are selected and cutter radius compensation is on, it must be physically possible for a circle whose radius is the half the diameter given in the tool table to be tangent to the contour at all points of the contour. In particular, the Interpreter treats concave corners and concave arcs into which the circle will not fit as errors, since the circle cannot be kept tangent to the contour in these situations. This error detection does not limit the shapes which can be cut, but it does require that the programmer specify the actual shape to be cut (or path to be followed), not an approximation.

In both examples, the line represents a contour, and the circle represents the cross section of a tool following the contour using cutter radius compensation (tangent to one side of the path.)
4.4 Examples

4.4.1 Easy Lead-in

```
( © 2018 by RosettaCNC Motion )
( file name: cutter_compensation.ngc )
G21
G40
G49
G90
G54
G50
G69
G0
x10y0z0
T0 M6
F2000
( Non compensated square )
M98 P1000
G0 x30y10z0
T1 M6
G41 D1
( Lead- in moves )
G1X20Y10
G1X30Y0I10J0
( Compensated square )
M98 P1000
G40
G0X30y10z0
M30
( Square with rounded corners )
O1000
G1X40
G1X50Y10I0J10
G1Y40
G1X40Y50I 10J0
G1X0
G1X0Y40I0J 10
G1Y0
G3X10Y0I10J0
G1X30
M99
```
4.4.2 Auto

; XY plane
G17
G00
Z50
G00
X50 Y50
M6
T0
F1000. S500
G00 Z10
M98 P1 ; Normal window
G00 X50 Y50
M6 T3
F1000. S500
G01 Z10
G41
M98 P2 ; Compensated window
G40
G52 X150
G00 X0 Y0 Z0
; XZ Plane
G18
G00 Y50
G00 Z50 X50
M6 T8
G00 Y10
M98 P2 ; Normal window
G00 Z50 X50
M6 T3
F1000. S500
G01 Y10
G41
M98 P2 ; Compensated window
G40
G52 X-150
G00 X0 Y0 Z0
; YZ Plane
RosettaCNC G-code language

G19
G00 X50
G00 Y50 Z50
M6 T0
G00 X10
M98 P3 ; Normal window
G00 Y50 Z50
M6 T3 F1000, S500
G01 X10
M98 P3 ; Compensated window
M40

G01 ; Gothic window XY
G01 Y-50
X50
Y100
G01 X-100 R50
G01 Y-100
X50
G50

G99

G02 ; Gothic window XZ
G01 X-50
Z50
X100
G01 X-100 R50
G01 X-100
Z50
G50

G99

G03 ; Gothic window YZ
G01 Z-50
Y50
Z100
G01 Y-100 R50
G01 Z-100
Y50
G50

G99
5. Motion Control Modes

RosettaCNC supports three different motion control modes:

- **Exact path mode** enabled with G61 that forces the CNC to follow the programmed path using the **look ahead** feature to diminish program duration.
- **Exact stop mode** enabled with G61.1 that forces the CNC to stop at the end of every motion block.
- **Continuous path mode** enabled with G64 that enables both **look ahead** and **trajectory blending** features to reach maximum speed performances. This mode supports two optional parameters:
  - P: Trajectory deformation/blending tolerance
  - Q: Points removal threshold that can be used to decrease the number of points generated by the CAM.

5.1 Examples

**Exact stop one shot example**

RosettaCNC supports **G9** command to force exact stop at the end of a specific block.

```
G00 G9 P5.0 F5000
G0 X-10 Y-10 Z0
G0400
G001B
G91B
Y0
(G9 example: Exact stop one shot)
```

```
G0 X-10 Y-10
G00 X-10 Y-10
G1 X0 Y0
G64
G64
G92.3
M30
```

(© 2018 by RosettaCNC Motion)
Path smoothing support

Using the motion control mode **Continuous path mode** and specifying the parameter P the path can be smoothed to satisfy user needs.
6. Canned Cycles

A canned cycle is a command that gives the machine instructions for a pattern of movements. It’s meant to automate and simplify repetitive and common tasks, such as drilling holes.

Instead of programming every movement and function individually, a canned cycle controls a set of motions.

In this section you can find a quick reference list followed by a paragraph for every cycle explaining how to use it properly.

6.1 List of supported Canned Cycles

<table>
<thead>
<tr>
<th>G-code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>G73</td>
<td>Drilling - High Speed Peck Drilling Cycle / Chip Break Drilling Cycle</td>
</tr>
<tr>
<td>G80</td>
<td>Cancel Motion Mode / Canned Cycle Cancel</td>
</tr>
<tr>
<td>G81</td>
<td>Drilling - Standard Drilling</td>
</tr>
<tr>
<td>G82</td>
<td>Drilling Cycle, Dwell</td>
</tr>
<tr>
<td>G83</td>
<td>Drilling - Peck Drilling Cycle</td>
</tr>
<tr>
<td>G85</td>
<td>Boring - Boring Cycle, Feed Out</td>
</tr>
<tr>
<td>G86</td>
<td>Boring - Boring Cycle, Spindle Stop, Rapid Move Out</td>
</tr>
<tr>
<td>G88</td>
<td>Boring - Boring Cycle, Spindle Stop, Manual Out</td>
</tr>
<tr>
<td>G89</td>
<td>Boring - Boring Cycle, Dwell, Feed Out</td>
</tr>
<tr>
<td>G98</td>
<td>Canned Cycle - Retract Back To The Initial Z</td>
</tr>
<tr>
<td>G99</td>
<td>Canned Cycle - Retract Back To R Plane</td>
</tr>
</tbody>
</table>

6.2 G73 High Speed Peck Drilling Cycle

G73 High Speed Peck Drilling cycle performs high-speed peck drilling. It performs intermittent cutting feed to the bottom of a hole while removing chips from the hole.

**Programming**

G73 X Y Z R Q F <L>

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X Y</td>
<td>Hole position data</td>
</tr>
<tr>
<td>Z</td>
<td>Z-depth (feed to Z-depth starting from R plane)</td>
</tr>
<tr>
<td>R</td>
<td>The distance from the initial level to point R level (Position of the R plane)</td>
</tr>
<tr>
<td>Q</td>
<td>Depth of cut for each cutting feed (depth of each peck)</td>
</tr>
<tr>
<td>F</td>
<td>Cutting feedrate</td>
</tr>
<tr>
<td>L</td>
<td>Number of repeats (if required)</td>
</tr>
</tbody>
</table>

Cycle Operation

The tool dips into the workpiece for the infeed Q, drives back with rapid feed of 0.254mm (d retraction) to break chips, dips in again, until end depth is reached, then retracts with rapid feed.

**Example**

N10 G99 G73 X10 Y10 Z-8 R2 Q1 F100
G98 G99
When G98 is active, the Z-axis will return to the start position (initial plane) every time it completes a single operation. When G99 is active, the Z-axis will be returned to the R point (plane) every time the canned cycle completes a single hole. Then the machine will go to the next hole.
Generally, G99 is used for the first drilling operation and G98 is used for the last drilling operation.

6.3 G80 Cancel motion mode / Canned Cycle Cancel

To cancel a canned cycle you can use G80 or one of the following G-codes from Group 01:

<table>
<thead>
<tr>
<th>G-code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>G0</td>
<td>Straight traverse</td>
</tr>
<tr>
<td>G1</td>
<td>Straight feed</td>
</tr>
<tr>
<td>G2</td>
<td>Clockwise arc feed</td>
</tr>
<tr>
<td>G3</td>
<td>Counterclockwise arc feed</td>
</tr>
<tr>
<td>G38.x</td>
<td>Probing</td>
</tr>
<tr>
<td>G80</td>
<td>Cancel motion mode / Canned Cycle Cancel</td>
</tr>
</tbody>
</table>
7. Feed Management

7.1 Rotary Axes Indexer option

If you set the axis to be an Indexer the G-code interpreter will check the G-code and ensure that the axis is moved only with single axis fast moves (G0). The most obvious case for indexing is to gain better access to the part.

7.2 Rotary Axes Continuous Machining

RosettaCNC supports “Rotary Axes Continuous Machining”, which may also be referred to as “Rotary Axes Contouring”. It enables all new kinds of machining and can also make existing jobs run faster and require less setup.

Single-axis feedrate moves

To perform feedrate moves on the rotary fourth axis G1 can be used, likewise a linear axis, except that distances are in degrees and feedrates are in degrees per minute.

Multi-axis feedrate "mixed moves"

“Mixed moves” are moves which combine rotary axis movement with linear axis movement. These moves too are programmed like normal linear moves. However, the formula for calculating the combined inch/degree feedrate is a little more complicated.

RosettaCNC "mixed moves feed handling" feature

If the origin of the rotational axis has been specified the feature named “Rotary Feed Handling” can be enabled. In this case RosettaCNC considers the distance from the origin of the rotational axis to the tool to perform a motion with the specified target speed.
Notes:

- RosettaCNC considers even the tool offset if enabled with G43.
- No need to use any special macro or perform complicated calculations

Standard macro approach

Since RosettaCNC provides Macro B compatible parametric programming we have created for you an arithmetic function within the control that determines feed rate in degrees per minute.

The function accepts input arguments in the form of desired per minute feed rate, the tool's position relative to the center of rotation, the amount of angular departure and the variable number in which the calculated degrees per minute feed rate will be stored. The function will calculate the degrees per minute feed rate based on the input data and store the results in the specified variable. This variable can be used to specify the feed rate in the rotary axis command.

The code of the macro is the following one

```gcode
; Calculate feed axis based on both linear and rotary movement
; A = #1 = rotary axis movement (optional argument)
; X = #24 = linear movement (optional argument)
; D = #7 = distance from center
; F = #9 = desired tool feed
; R = #18 = return value

O1001
IF [#24 EQ 0] THEN #24 = 0
IF [#1 EQ 0] THEN #1 = 0
IF [#1 NE 0] AND [#24 NE 0] THEN1
#100 = [3.1416 * #7] ; calc circumference
#100 = [ABS[#1] / ([100 / #9])]
END1
#100 = [ABS[#24] / #100] ; new linear feed
END1
```

It could be stored in a dedicated file in the macros folder and used as follows

```gcode
; (c) 2019 by RosettaCNC Motion
M90 S12000 F1000
G56 G90 G21
G00 X100 Z35
G01 Z5 F700
; Single rotary axis feedrate move with a target feed of 700 mm/min
G65 P1001 A370 F700 D30 R101
G01 A370 F#101
; Multi-axis mixed move with a target feed of 700 mm/min
G65 P1001 A-720 X-60 F700 D30 R101
G01 X-60 A-720 F#101
G00 G90 Z35
M30
```
8. Coordinate Systems

RosettaCNC G-Code has some powerful features that allow the user to transform coordinates in order to create complex shapes with just a few lines.

8.1 The 5 Steps G-Code Coordinate Pipeline

Coordinates are handled in 5 different steps/levels, every step influences all the steps at is right.

8.1.1 Step 1: Unit Conversion

The first calculation involves handling the length units used your program that can be mm or inches as defined by G20 and G21.

8.1.2 Step 2: Conversion from Relative or Polar to Absolute Coordinates

The following step involves converting relative or polar coordinates to absolute coordinates.

- **Relative coordinates** are enabled with G90 and disabled with G91
- **Polar coordinates** consist in specifying a radius and an angle instead of X and Y Cartesian coordinates and are enabled by G15 and disabled by G16. When polar coordinates are enabled X is the radius relative to the current position and Y is the angle. (For further explanations see Cartesian & Polar Coordinates

8.1.3 Step 3: Offsets: G52, G54 and G92

RosettaCNC supports three different kinds of offsets:

- **Work Offsets** define different places a part zero may be. They’re typically handled via G54, G55, ...
- **Local Offsets** are defined by G52. They make it easily to temporarily move the zero to a new place. For example, the center of a bolt circle while you’re busy drilling it so you don’t have to offset each hole from part zero.
- **Workpiece Coordinate System** setting set using G92 is another facility for setting offsets.

Examples

To update the WCS during a G-code program

```gcode
#<wcs.index> = 1; Set a variable to store the desired WCS index.
; Notes:
; - index is 1 for G54, 2 for G55, ...
```
8.1.4 Step 4: Scaling and Mirroring: G51

A G51 applies scaling/mirror to all positions, lines, and arcs following this G-code until a G50 are entered. A different scaling factor can be specified for every axis using I (for X axis) J (for Y axis) and K (for Z axis) while parameter P can be used if scaling factors are the same for all the axes. The X, Y, and Z parameters are the coordinates of the scaling center. If the scaling center is not specified, the default scaling center is the current cutter position (not the current origin). To mirror, enter a negative value for the scaling factor.

Notes:

- If the arc radius was specified with R, the radius will be scaled by the larger of the two circular plane scale factors. The result will be a circular arc between the scaled arc start and the scaled arc end.
- If the arc center was specified with I, J, and/or K, the centres will be scaled by the appropriate axis scale factors. The result will be a circular arc from the scaled arc start, around the scaled center, and usually with a line from the end of the circular arc to the scaled arc end.
- In no case can an ellipse be generated using scaling.
8.1.5 Step 5: Rotation: G68

With **G68**, you can rotate the coordinates an arbitrary number of degrees about an arbitrary center. The X, Y, and Z parameters are the coordinates of the scaling center. **If the rotation center is not specified, the default rotation center is the current cutter position (not the current origin).**

![Diagram of rotation with center of rotation highlighted](image)

---

© 2018 by RosettaCNC Motion

( file name: rotation.ngc )

( G-code rotation using G68 and G69 )

G0 X0 Y0 M98 P2 L1

G0 X300 M98 P2 L1

G0 X0 Y0 M30

G01 X-50 Y100.0

G02 X100 Y0 I50 ; or I-5 J0

G01 Y-100

M99

---

O1

F20. S500

G17 G0 X0 M98 P1 L1

G0 X300 M98 P2 L1

G0 X0 Y0 M30

O2

; XY Draw an entire flower with 4 petals

#1 = 0

WHILE [#1 LT 4] DO1

#3 = [#1 + 1]

M98 P1 L1

G68 R#3 ; The current point is set as origin if X,Y

; (or A,B using HAAS notation plane independent) are not specified.

EN01

G69 ; Reset origin offsets and rotation

M99

O1

; Draw a single petal in the XY plane

G91

X-50 Y100.0

G01 X100 Y0 I50

G01 X50 Y-100

G00 ; switch to absolute

M99
8.2 Cartesian & Polar Coordinates

With Cartesian Coordinates X, Y, and Z represent distances from part zero (absolute coordinates) or from the current position (relative coordinates).

With polar coordinates, we use an angle and a distance relative to the origin.

The following picture gives a graphical overview.

8.2.1 Use on different planes

8.2.2 Standard syntax

- **XY Plane**: X is used to identify the radius and Y to identify the angle
- **YZ Plane**: Y is used to identify the radius and Z to identify the angle
- **XZ Plane**: X is used to identify the radius and Z to identify the angle

**Plane independent syntax**

It is also possible to use a plane independent syntax using two special symbols:

- @ for the radius
- ^ for the angle
8.2.3 Examples

8.2.4 Polar Coordinates standard syntax

The following example uses polar coordinates to generate a rotated square and a spiral.

```
( © 2018 by RosettaCNC Motion
Polar coordinates example:
G17 G21 G40 G49 G54 G69
G90
G54
F1000
G0
X0 Y0 Z0
(Polar square)
G16 (Enable polar coordinates)
G1 X50 Y90
G91 (Switch to relative)
Y90
Y90
Y90
Y90
G90 (Switch to absolute)
G15 (Disable polar coordinates)
G0 X0 Y0 Z0
(Polar spiral)
G16 (Enable polar coordinates)
G1 X10 Y90
G91 (Switch to relative)
X10 Y90
X10 Y90
X10 Y90
G90 (Switch to absolute)
G15 (Disable polar coordinates)
G0 X0 Y0 Z0
M2
```
8.2.5 Polar Coordinates complete example

```gcode
; © 2018 by RosettaCNC Motion
; Polar coordinates example:
G17 G21 G40 G49 G54 P1000
G0 X0 Y0 Z0

(Polar coordinates plane independent style)
G17 M98 P1000
G18 M98 P1000
G19 M98 P1000
X100 Y0 Z0
G52 X100

(Polar coordinates Fanuc style)
G17 M98 P1001
G18 M98 P1002
G19 M98 P1003
M2

O1000
(Polar square)
G1 X50 Y90
G91 (Switch to relative)
Y90
Y90
Y90
G90 (Switch to absolute)
G6 X0 Y0 Z0

(Polar spiral)
G1 X10 Y90
G91 (Switch to relative)
X10 Y90
X10 Y90
X10 Y90
G90 (Switch to absolute)
G6 X0 Y0 Z0

M99

(Polar coordinates Fanuc style XY plane)
O1001
(Polar square)
G18 (Enable polar coordinates)
G1 X50 Y90
G91 (Switch to relative)
Y90
Y90
Y90
G90 (Switch to absolute)
G15 (Disable polar coordinates)
G6 X0 Y0 Z0

(Polar spiral)
G18 (Enable polar coordinates)
G1 X10 Y90
G91 (Switch to relative)
X10 Y90
X10 Y90
X10 Y90
X10 Y90
G90 (Switch to absolute)
G15 (Disable polar coordinates)
G6 X0 Y0 Z0

M99

(Polar coordinates Fanuc style XZ plane)
```
O1002
(Polar square)
G16 (Enable polar coordinates)
G1 X50 Z90
G91 (Switch to relative)
Z90
Z90
Z90
G90 (Switch to absolute)
G15 (Disable polar coordinates)
G6 X0 Y0 Z0
(Polar spiral)
G16 (Enable polar coordinates)
G1 X10 Z90
G9 (Switch to relative)
X10 Z90
X10 Z90
X10 Z90
G90 (Switch to absolute)
G15 (Disable polar coordinates)
G6 X0 Y0 Z0
(Polar coordinates Fanuc style YZ plane)
O1003
(Polar square)
G16 (Enable polar coordinates)
G1 Y50 Z90
G91 (Switch to relative)
Z90
Z90
Z90
G90 (Switch to absolute)
G15 (Disable polar coordinates)
G6 X0 Y0 Z0
(Polar spiral)
G16 (Enable polar coordinates)
G1 Y10 Z90
G9 (Switch to relative)
Y10 Z90
Y10 Z90
Y10 Z90
Y10 Z90
G90 (Switch to absolute)
G15 (Disable polar coordinates)
G6 X0 Y0 Z0
M99
9. Rotary axis options

Rotary axis visualisation can be:

- **linear**: rotary axis position is displayed as a normal linear axis position, only units are different
- **rounded**: rounded by the angle corresponding to one rotation

Independently on how the rotary axis position is visualised the user can chose among 3 modes to set the target position. The modes are set using parameters \#5802 for axis A, \#5803 for axis B and \#5804 for axis C calling G10 L100 P<parameter> V<value>.

The supported modes are described by the following table.

<table>
<thead>
<tr>
<th>Parameter value</th>
<th>Mode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Rotary axis Roll-over disabled</td>
<td>Rotary axis is handled as a linear one.</td>
</tr>
<tr>
<td>1</td>
<td>Rotary axis Roll-over</td>
<td>For an incremental command, the tool moves the angle specified in the command. For an absolute command, the coordinates after the tool has moved are values set in parameter #5801 rounded by the angle corresponding to one rotation.</td>
</tr>
<tr>
<td>2</td>
<td>Rotary axis Roll-over &amp; shorter path</td>
<td>For an incremental command, the tool moves the angle specified in the command. For an absolute command, the coordinates after the tool has moved are values set in parameter #5801 rounded by the angle corresponding to one rotation. The direction of the movement is the direction in which the final coordinates are closest.</td>
</tr>
<tr>
<td>3</td>
<td>Rotary Axis Control</td>
<td>This mode influences a rotary axis movement only for absolute commands. When this mode is enabled, the sign of the value specified in the command is interpreted as the direction of rotation, and the absolute value of the specified value is interpreted as the coordinates of the target end position. The target end position is considered modulo the value specified by parameter #5801.</td>
</tr>
</tbody>
</table>

If a mode different from 0 is specified the modulus used for all of them (default 360.0) is stored in parameter \#5801.

**Example**

```
G90 G49 F1000 G0 X0 Y0 Z0 A0
G1 Z20
G10 L180 P5802 V2 ; Set the "Rotary axis Roll-over & shorter path" mode for axis A.
G90 ; Set position mode to absolute
A158 ; Actual movement: -150.0 Absolute coordinate rounded visualisation value: 210
A540 ; Actual movement: -30.0 Absolute coordinate rounded visualisation value: 180
A858 ; Actual movement: -80.0 Absolute coordinate rounded visualisation value: 100
G91 ; Set position mode to relative
A380 ; Actual movement: +380.0 Absolute coordinate rounded visualisation value: 120
A-840 ; Actual movement: -840.0 Absolute coordinate rounded visualisation value: 0
M2
```
10. Messages & Media

The user can edit the G-code to specify a message or a media to be displayed using the following syntax.

M109 P“text message to be displayed” Q1
M120 P“image.png” Q1

10.1 Parameters meaning

- P the text to be displayed for M109 and the path to the image to be displayed using M120.
- Q the optional window type to be used to display the message or the media. The following types are available:
  - 0: modal window with a Resume button.
  - 1: modal window with a Stop button.
  - 2: modal window with a Stop and Resume button.
  - 3: modal window where the user can insert a value that will be stored in parameter #5721.
  - 4: message will be displayed in the HUD.
- D the optional default value to be used when Q is set to 3. The specified is used to generate the initial preview.

A special case of M109 is M109 p“user error message” Q-1 used to stop compilation reporting a custom error message placed between double quotes.

When no message is present after M109 (Eg: M109 Q-1) the default text [E0345] in line x - generic user error with m109 q-1 will be showed.

Notes

- Only with M109 Q4 (HUD message) you have to use an empty string to remove previous showed message: M109 P“” Q4
- It is possible to show the current value of a parameter within a message, to do that just add to the message text #<parameter_number> where parameter_number is the number of the parameter you want to display.
  
  Example: M109 P“The current value of the parameter 5001 is #5001” Q0
- The images to be show using M120 should be placed in the folder <Documents/RosettaCNC-1/machines/media>
- The text messages with q0÷3 can accept a mini-html subset of tags.

10.2 Supported HTML syntax

In the text field of a message the HTML syntax can be used.

**Bold tag**

*B* : start bold text
* /B* : end bold text

*Example:* This is a *test* → This is a **test**

**Underline tag**

*U* : start underlined text
* /U* : end underlined text

*Example:* This is a *<u>test</u>* → This is a **<u>test</u>**

**Italic tag**

*I* : start italic text
* /I* : end italic text
Example: This is a `<I>`test`</I>` → This is a test

**Strikeout tag**

`<S>` : start strike-through text  
`</S>` : end strike-through text  
**Example:** This is a `<S>`test`</S>` → This is a test

**Line break**

`<BR>` : inserts a line break  
**Example:** This is a `<BR>` test →  
This is a test

**Subscript/Superscript tags**

`<SUB>` : start subscript text  
`</SUB>` : end subscript text  
`<SUP>` : start superscript text  
`</SUP>` : end superscript text  
**Example:** This is `<SUP>`9`</SUP>`/ `<SUB>`16`</SUB>` → This is $\frac{9}{16}$

**List tags & List items**

`<UL>` : start un ordered list tag  
`</UL>` : end un ordered list  
`<LI [type="specifier"] [color="color"]>` : new list item  
specifier can be "square" or "circle" bullet  
specifier "color" sets the color of the square or circle bullet  
**Example:**

```
<LI> List item 1  
<LI> List item 2  
<UL>  
<LI> Sub list item A  
<LI> Sub list item B  
</UL>  
<LI> List item 3  
</UL>
```

becomes:  
1. List item 1  
2. List item 2  
   ○ Sub list item A  
   ○ Sub list item B  
3. List item 3

**Text with shadow**

`<SHAD>` : start text with shadow  
`</SHAD>` : end text with shadow

**Highlight**

`<HI>` : start text highlighting  
`</HI>` : stop text highlighting

**Error marking**

`<E>` : start error marker
10.3 Supported Not Standard HTML Tags

In the text field of a message a set of non HTML standard tags can be used. The non HTML standard tags have a @ as prefix.

Tool Info

<@TI=tool_id> : Insert info of tool defined in tool_id
<@TD=tool_id> : Insert description of tool defined in tool_id

10.4 Examples

Modal window messages

User message of type 0 (full screen view)

User message of type 0 (window)

User message of type 1 (window)
User message of type 2 (window)

HUD messages

User message of type 4 (HUD on preview)

Use piece thickness to update WCS offsets

(c) 2016-2019 by RosettaCNC Motion

G54 G49 P1000
G10 L2 P1 X0 Y0 Z0 : Set G54 WCS offsets
G10 L2 P2 X0 Y0 Z 20 : Set G55 WCS offsets
G10 L2 P3 X0 Y0 Z 30 : Set G56 WCS offsets
: Set initial position.
G0 X0 Y0 Z0

G65 P1000
M02 : Prompt the user to insert piece thickness and update the WCS.
: The code could be moved to a dedicated file.
G0189 P"Set piece thickness" Q3 D0.0
G10 L2 P3 Z H5211 + #3721 : Update G54 WCS offsets
G10 L2 P3 Z H5233 + #3721 : Update G55 WCS offsets
G10 L2 P3 Z H5263 + #3721 : Update G56 WCS offsets
M99
11. Probing

When this command is invoked the machine moves towards the target position along a straight line at the current feed (F). The move stops, within machine acceleration limits, when the target position is reached, or when the requested change in the probe input takes place.

Target position is interpreted considering the active WCS if G53 is not specified in the same line.

After successful probing, parameters #5701-#5706 (and #5711-#5716) will be set to the program coordinates of X, Y, Z, A, B, C considering the active WCS.

Parameter 5700 is set to 1 or 0 depending on the probe state at the end of the motion.

There are 4 probing options:

- G38.2: probe toward workpiece, stop on contact, signal error if failure
- G38.3: probe toward workpiece, stop on contact
- G38.4: probe away from workpiece, stop on loss of contact, signal error if failure
- G38.5: probe away from workpiece, stop on loss of contact

11.1 Error Cases

There are some cases in which G38.x can raise a compilation error.

<table>
<thead>
<tr>
<th>Error</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>E044: cannot move rotary axes during probing</td>
<td>This occurs when in the G38.x is defined a rotary axis with target position different than actual position.</td>
</tr>
<tr>
<td>E047: cannot probe in inverse time feed mode</td>
<td>This occurs when is active G93 (enable feed per inverse of time).</td>
</tr>
<tr>
<td>E048: cannot probe with cutter radius comp on</td>
<td>This occurs when a cutter radius compensation G40/G41.1/G42/G42.1 is active.</td>
</tr>
<tr>
<td>E049: cannot probe with zero feed rate</td>
<td>This occurs when feed rate F is set to zero.</td>
</tr>
<tr>
<td>E062: start point too close to probe point</td>
<td>This occurs when the delta interpolated movement from start point to probe point is lower 0.254mm (0.01in).</td>
</tr>
</tbody>
</table>

Notes

- G38.2 and G38.4 trigger an alarm if the requested state change of the probe is not performed
- G38.3 and G38.5 probe result can be checked using G-code control statements reading the parameter #5700
  - 1: probing procedure succeed [ #<probe.state.succeed> ]
  - -1: probing procedure failed, sensor not tripped before reaching the target position [ #<probe.state.not_tripped> ]
  - -2: probing procedure failed, sensor already tripped before starting the procedure [ #<probe.state.already_tripped> ]
- after a successful probing parameters:
  - #5701-#5706 are set to the probed position accordingly to the active WCS
  - #5711-#5716 are set to the probed position accordingly to machine coordinates.

Examples

Detect piece position

This example shows how the macro “update_piece_position.ngc” is used to detect the Z position of a piece.
RosettaCNC G-code language

M109 P“Update piece position” Q4

; Uncomment the following code if one of the WCS should be updated
; IF [#1 EQ #0] THEN1
;    M109 P“Argument A should corresponds to a WCS index: 1 -> G54, 2 -> G55, ... ” Q1
;    M2
; END1

IF [#5700 EQ #0] OR [#5613 EQ #0] OR [#5621 EQ #0] OR [#5622 EQ #0] OR [#5623 EQ #0] OR [#5624 EQ #0] OR [#5635 EQ #0] THEN10
    M189 P“One or more compulsory system parameters are not specified.” Q1
    M2
END10

; Check if a G38.X is used. In that case convert to a G80 to prevent an error when setting G#4101
IF [#5101 GE #38] AND [#5101 LT #39] THEN10
    #10 = #0
END10

; Disable spindle, flood & mist
M5 M9
F #6003
; Move upwards to a ‘safe position’
G53 G1 Z #6013
; Move the a position near the sensor ’Approaching sensor target Z position’
G53 F #6021 G38.3 Z #6023
IF [#5700 EQ #0] THEN M109 P“Error updating piece position: Sensor already tripped!” Q1
    ; Sensor will not be tripped during the fast approach
IF [#5700 EQ #1] THEN M109 P“Error updating piece position: Sensor tripped during fast approach!” Q1

; Calculate the offset using the BCS position when the sensor was tripped #5173,
; the BCS position of the sensor and the length of the current tool.
#10 = [#5713 - #6035 - #5403]

; Update the G54, G55 or...
; Uncomment the following line if one of the WCS should be updated
; G10 L2 #11 Z #10
; or use G52 or G92 offsets
G52 Z #10
F #6003
; Move upwards to a “safe position”
G53 G1 Z #6013
; Restore original target feed
F #6130
; Restore original G code (G0 or G1 or G2 or G3 or ...) G #4101 ; or reset setting G80
; Restore previous states
IF [#4151 EQ #3] THEN M3
IF [#4151 EQ #4] THEN M4
IF [#4151 EQ #7] THEN M7
IF [#4154 EQ #8] THEN M8
M109 P”” Q4
M93

Usage example.

(c) 2016-2019 by RosettaCNC Motion
G54 F 1000
G0 X0 Y0 Z 20
G65 P“update_piece_position.ngc”
; Go to the origin of the WCS
G1 X0 Y0 Z 20
M109 P“G52 offsets have been used. Program Z position is #5003 and Machine Z position depends on where the piece is placed.”
M30
12. RTCP

In 5 axes machining (3 linear axes + 2 rotary axes) the main goal is to keep under control the contact point between tool and piece. When generating the part-program, the CAD-CAM system calculates the points coordinates on the piece surface and the rotary axes orientation (swiveling head or table).

RosettaCNC, thanks to optional RTCP (Rotating Tool Center Point) feature, calculates the axes movements in order to keep the tool tip in the right position, taking automatically into account the machine geometry and the tool length. This automatic calculation allows to execute the same part-program with different tool length or different machine geometry without needing a regeneration of the part-program with the CAD-CAM: less down-time, increased productivity.

12.1 Gcode

**G43.4**

G43.4 is the standard G code to enable RTCP compensation. When RTCP is enabled using G43.4 the CNC compensates tool offsets and machine kinematics but not the tool radius.

```
G54  G49  G21
G0  X0  Y0  Z110  A0  C0
; Enable RTCP for the tool 1.
; RTCP will compensate machine kinematics and tool length.
G43.4  H1 ; To compensate for the loaded tool write H#5134
F1000  G45
X100
Y100
M2

; The CNC will rotate axis A keeping the tool tip in the current position.
Z0
X100
C
-90
; The CNC will rotate axis C keeping the tool tip in the current position.
Y100
M2
```

**G43.7**

G43.7 is similar to G43.4 but the user can override the values of the tool offsets. Overriding tool offsets can be useful to compensate for the tool radius, when the cutting point of the tool is known. This G code is useful to write easily Gcodes programs by hand when a saw tool is used.

```
G54  G49  G21
; Assumptions:
; 1) The saw is already in the desired position.
; 2) We have just stored the current TCP position in the G54 WCS
; so that our program is based on the point where the saw touches
; the material.
G43.7  H1  Y[-1 * (#5410/2)] ; The "Y[-1 * (#5410/2)]" is used to enable the radius compensation for the current saw.
#10  = 500
#100  = [#10*\cos(\#5006)]
#101  = [#10*\sin(\#5006)]
F1000
G1  X100  Y100  Z0
G1  C0  F1000
A90
#10  = 500
#100  = [#10*\cos(\#5006)]
#101  = [#10*\sin(\#5006)]
G91
G1  X100  Y101  F5000
G90
M2
```
Example

Suppose you are looking from the right side a C/A 5-Axis Head and you set a target position of -90 for axis A.

<table>
<thead>
<tr>
<th>RTCP disabled</th>
<th>RTCP enabled</th>
</tr>
</thead>
<tbody>
<tr>
<td>A - 90.0</td>
<td>A - 90°</td>
</tr>
<tr>
<td>causes a movement of axis A without any linear axis movements that would keep the tool in contact with the part.</td>
<td>causes the swing of -90° for the axis A and the displacements of the linear axes to maintain the tool tip at the same location</td>
</tr>
</tbody>
</table>

- **RTCP disabled**: A - 90.0 causes a movement of axis A without any linear axis movements that would keep the tool in contact with the part.
- **RTCP enabled**: A - 90° causes the swing of -90° for the axis A and the displacements of the linear axes to maintain the tool tip at the same location.

### 12.2 Supported Kinematics

Machines with 5 axes may have different types of kinematic motions to be controlled. RosettaCNC supports:

- rotating-tilting table A/C with vertical head
- rotating-tilting table B/C with vertical head
- rotating-tilting head A/C
Machine origin should be placed exactly where the C axis center of rotation is located.

Settings:
- D.x : the offset along the X axis between the axis C and axis A centres of rotation
- D.y : the offset along the Y axis between the axis C and axis A centres of rotation
- D.z : the offset along the Z axis between the axis C and axis A centres of rotation

Machine origin should be placed exactly where the C axis center of rotation is located.

Settings:
- D.x : the offset along the X axis between the axis C and axis B centres of rotation
- D.y : the offset along the Y axis between the axis C and axis B centres of rotation
- D.z : the offset along the Z axis between the axis C and axis B centres of rotation

Machine origin should be placed considering that the tool holder (tip of the head without a tool mounted) should be at Z = 0 when A is 0.

Settings:
- H.x : the offset along the X axis between A and C centres of rotation
- H.y : the offset along the Y axis between A and C centres of rotation
- H.z : the offset along the Z axis between A and C centres of rotation
- J.x : the offset along the X axis between A centre of rotation and the center of the tool holder
- J.y : the offset along the Y axis between A centre of rotation and the center of the tool holder
- J.z : the offset along the Z axis between A centre of rotation and the center of the tool holder
13. Acknowledgement

All those who desire to contribute improving this documentation are encouraged to report inaccuracies or incorrect content. Write to the address: support@rosettacnc.com