Introduction

G-code or **G programming language** is the common name for the most widely used numerical control (NC) programming language. It is used mainly in computer-aided manufacturing to control automated machine tools. There are several variants (manufacturer specific implementations) of this language, but the main features are standardized (ANSI standard number RS-274). It has now become the work-horse of the machining and manufacturing community. G-code programs are typically auto-generated by CAD/CAM software.

G-code Support by Click&Move®

To provide an easy way for programming multi-dimensional geometric motion profiles, Click&Move® supports G-codes. Since G-code processing function blocks (FBs) are embedded in the much more powerful language of C&M, **only those parts of RS-274 are implemented, which serve to create a path.** Machine functions, e.g. tool change, control of spindles, coolant flow control etc., can be implemented by the user. Practically any standard or user-specific commands **can be implemented by simply creating a FB diagram.** This user-created FBD converts RS-274 commands into the appropriate commands for the corresponding I/O or servo axes. In standard G-code programming, the F-word establishes the effective feed-rate of the tool-tip (how fast it moves). However, Click&Move specific codes like *Accel, Decel, Jerk* etc., **provide for sophisticated motion control that can't be described by standard G codes**.

G-code Language Syntax

G-code programs are simple text files composed of consecutive lines called **blocks** or **sentences**. The blocks (or sentences) act as execution units that are executed sequentially. G-code doesn't specify a special code for the start of a program, but the end of the program can be marked by certain end-codes. Whenever the interpreter encounters one of these end-codes it assumes that the current line is the last line of the program. Lines of a file that occur after the end-code are not to be executed. G-code blocks (or sentences) consist of one or more **words**. G-code words begin with a **letter-code** (also called **address**), which is followed by a numerical value (e.g. G3 or P47 or X0.25). Words can be separated by spaces or tabs, but it is not required (e.g. "X17 Y23 Z41" and "X17Y23Z41" are equivalent). The unified syntax of "address+value" may suggest that all words represent a kind of value assignment, however certain word-types are interpreted differently. In this aspect, G-code words can be sorted into the following types:

- **Commands:** The words beginning with letter G (called **G-commands** or **G-codes**) provide all motion-commands, operation mode settings and other actions. The words beginning with letter M (called **M-commands** or **M-codes**) are reserved for machine- specific or user-defined operations. The number which follows the G or M letter is not a parameter: it is part of the command (it determines the meaning of the command). The type of the number is usually integer (standard g-code commands), but some commands (extended functions, additional to standard G-code) use a decimal point. (e.g. G2, G3 are standard, but G2.1, G3.1, G2.2, G3.2 are C&M extensions).
- **Parameters:** Most of the letter-codes (A, B, C, D, E, F, H, I, J, K, L, P, Q, R, S, T, U, V, W, X, Y, Z) can be thought of like numerical parameters. Therefore, any words beginning with these letters perform simple value assignment (e.g. A23 means A=23). Init value also can be set for each address (see G-code Property Settings). The number type is integer or float as well. Some parameters have dedicated meaning, e.g. F (feed-rate), S (spindle-speed) or T (tool-ID). These parameters can be used individually, while other parameters are attached to G or M commands as arguments.
- **Identifiers, Labels:** The O-letter followed by an integer is used as the program identifier (optional). The N-letter followed by an integer acts as line number. Line numbers may also be skipped, and that is normal practice. A line number is not required to be used, but must be in the proper place if used.

C&M specific parameters: In addition to standard letter-codes (A-Z), there are some

ADVANCED Motion Controls · 3805 Calle Tecate, Camarillo, CA, 93012
ph# 805-389-1935 · fx# 805-389-1165· www.a-m-c.com

Click&Move specific keywords, too. These are the implementations of some advanced functions, that are not included by standard G-code, such as: dynamic properties of motion (Accel, Decel, Jerk), tool-orientation along the path (Roll, Pitch, Yaw), wide range of transition settings (TransMode, TransParam, TransAccel) and others. These reserved keywords are recognized and act as dedicated parameters *(see later for more details)*.

User-specific commands: Users can create their own M or G commands. This technique is based on some general-purpose registers called **Misc-Values** (Misc1-Misc12). Any letters A-Z can be assigned to a Misc-register (*see G-code Property description*). In the course of G-code processing the desired (A-Z) values are copied to the selected Misc registers, therefore these values will be available for the user. For example, if a user wants to introduce M325 as a new command, which uses P and Q parameters, with a simple misc-configuration (e.g. Misc1=M, Misc2=P, Misc3=Q) all data will appear in the corresponding Misc-columns of the **MotionTask** file. During run-time, these values can be read at the Misc-outputs of the **MotionTaskRun** FB.

Click&Move® Specific Syntax Extensions

Comments: Printable characters and white-spaces inside parentheses are comments. Comments do not cause a machining center to do anything. A left parenthesis always starts a comment. The comment ends at the first right parenthesis found thereafter. Once a left parenthesis is placed on a line, a matching right parenthesis must appear before the end of the line. A G-code line may contain more than one comments, and can be inserted between words:

```
G01 Z15 (move up) G91 (relative)
```

Comments may be nested, which allows the simple disabling of a commented command:

```
M05 (G01 Z15 (move up))
```

Complete G-code lines can be disabled by inserting a semi-colon:

```
;G01 Z15 (move up) G91 (relative)
```

External Parameters: Constant numerical values in the words can be replaced by variables. These variables are actually external parameters collected in an array. This array supplies the parameter values for G-code processing, when it is started. The individual variables can be referenced by array index signed with a hashmark:

```
G01 X#1 Y#2 (X= param1, Y= param2)
```

Program Cycles: If some section of the program is to be executed a specific number of times, a cycle command can be used. This is actually not a g-code command, but a repetition control structure. The **cycle-header** begins with a **cycle** keyword followed by the number of repeats (e.g. Cycle25). External parameters also can be used as repeat number (e.g. Cycle#1). Then the **cycle-body** (one or more program lines) comes between curly brackets. The opening curly bracket may be placed after the cycle header directly or in new line. The closing curly bracket may be placed directly at the end of the cycle-body, or in new line. The opening and the closing brackets both must terminate their line (nothing is allowed to be placed after them). Cycles may be nested into each other at maximal depth of 20. Examples:

```
Cycle5
{
  G01 X17
  G01 Y23
}
```

```
Cycle5
{
  G01 X17
  G01 Y23}
```

```
Cycle5{
  G01 X17
  G01 Y23
}
```

```
Cycle5{
  G01 X17
  G01 Y23}
```

```
Cycle3
{
  Cycle4
  {
    G90 G00 X10 Y20
    G91
    G01 X80
    G01 Y60
    G01 X-80 Y-60
    G52 X100
  }
  G52 X-400 Y90
}
```

Command Structure and Behavior

Several G and M commands require one or more arguments (coordinates, offsets, rotation angles, feed rate, spindle speed, tool dimensions, etc.). Therefore, many G or M commands are followed by parameters, for example:

```
G01 X58 Y24 F300
```

Some addresses are used for dedicated purposes, but most of the addresses have various meaning in various contexts. For example, X, Y, and Z are always coordinate-values and F is always feed-rate, but the meaning of R alters in the following cases:

```
G02 X58 Y24 F300 R40     (circle, R=radius)
G81 X58 Y24 F300 Z-5 R2  (spot-drilling, R=retract level)
```

It is not necessary to specify all arguments for a command. The missing arguments are supplied by the default value or the former setting of the corresponding parameter. If the same argument-value is intended to be used for several consecutive commands, it is enough to perform the parameter-setting only once in the beginning. This is a characteristic feature of g-code language. In this aspect, G and M commands can be divided into 2 different classes:

- **Modal Commands:** Many G and M commands cause the machine to change from one operational mode to another, and the mode stays active until some other command changes it. Such commands are called "**modal**", they are also called preparatory functions. Modal codes are like a light switch. Flip it on and the lamp stays lit until someone turns it off. For example, the coolant commands are modal. If coolant is turned on (M08), it stays on until it is explicitly turned off (M09). The motion commands (G00, G01, G02, G03), plane selections (G17, G18, G19), coordinate units (G20, G21) or coordinate-system selections (G54...G59) are also modal. Modal commands are arranged in sets called "**modal groups**". Only one member of a modal group may be in force at any given time. In general, a modal group contains commands for which it is logically impossible for two members to be in effect at the same time. The simplest modal groups contain only 2 commands: one for turning the desired function on and a mate-command for turning it off (e.g. G68 and G69). A machine tool may be in many modes at the same time, with one mode from each group being in effect.
- **One-shot Commands:** A few G and M commands affect only the lines on which they occur, for example G4 (dwell) or G53 (select machine coordinate-system). Such commands are called non-modal or "**one-shot"**. Since these commands have an influence on a single block, they have no mate-command to turn them off. Also, the arguments of the command are applied only to the actual block, so there is no need to store them.

Detailed Description of G-commands

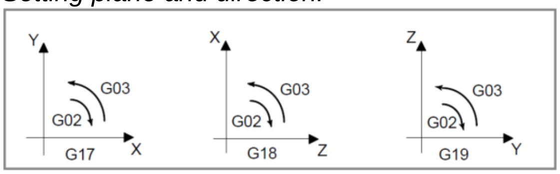
G-commands (or G-codes) are also called **general commands**. These are the commands that tells the machine what to do. Some of the G-codes deal with commanding motion in a certain way (such as: rapid move, controlled feed move in a straight line or arc); others toggle between settings (units, plane-selection, coordinate systems) or act as switches to influence the machine's motion planning logic (tool length/radius compensation).

Motion commands: In machine controls motions can be sorted in two main types: point-to-point and coordinated motions. In point-to-point motion, the end position is designated, but the path used to reach the end position is irrelevant. In coordinated motion, the path is also strictly defined. All type of motions start from the actual position and move to the end-position specified by X, Y, Z addresses. The feed-rate of all the coordinated motions is specified by the F address.

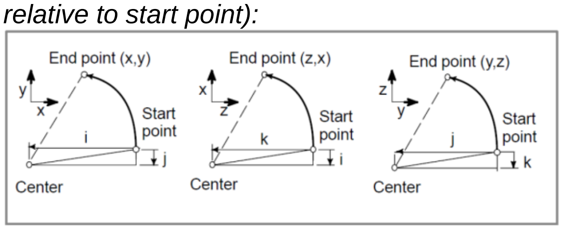
- **Rapid positioning (G00):** This command performs a point-to-point motion to the specified position at a rapid traverse rate. The simplest implementation is when each machine axis moves independently. In this case, the individual axes may finish their motion at a different time. Therefore, a more advanced implementation is used to simultaneously complete motion (by synchronized axes). In cartesian mechanics it results in a straight motion, but in common cases (e.g. scara, kuka, puma and other robots) motion path is undetermined. The rapid traverse rate cannot be specified by the address F.
- **Linear interpolation (G01):** Linear interpolation is used when the desired move is a straight line in XYZ space. The controller coordinates the motion of all axes, calculating a series of very small moves along each axis that result in a straight line when executed.

- **Circular interpolation (G02, G03):** Circular interpolation is used when the desired move is a circular arc in XY or YZ or XZ plane. It works much the same way as linear interpolation, but also requires the specification of the plane of the circle (G17=XY, G18=ZX, G19=YZ), the direction (G02=clockwise, G03=counterclockwise) and either the center point (I, J, K addresses) or the radius (R address). Number of turns can be specified optionally by P address (P default is 0, which can be used for an arc of angle up to 360°).

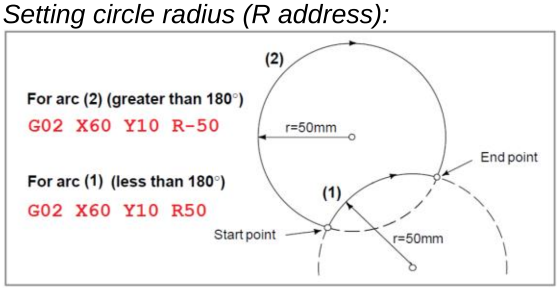
  Setting plane and direction:



  Setting center point (I, J, K addresses are relative to start point):



  Setting circle radius (R address):



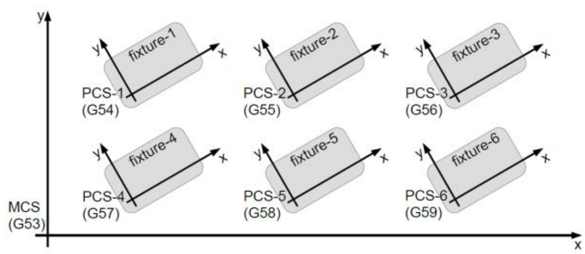
- **Helical interpolation:** A circular motion combined with a perpendicular linear motion results in a helical motion (e. g. circular motion in XY-plane + linear motion in Z direction). Helical motion can be implemented with a simple circular interpolation command (G2, G3) by specifying an end-point in a different plane (different from the plane of start-point).
- **Involute interpolation:** This can be implemented by circular interpolation commands (G02, G03) by specifying a center point with a different distance from the start and end-points.
- **Radius Vector type circle (G2.1, G3.1)** This is a generalization of the radius-type circle. The length of the radius vector specifies the radius of the circle. The direction of the radius vector specifies the plane of the circle (radius vector = normal-vector of the circle-plane). With this method, not only main planes (XY, YZ, XZ), but any general planes can be used. This type of circle also can turn into helical interpolation.
- **Border Point type circle (G2.2, G3.2)** Three different points (if they are not placed in a line) always specify a circle. For this circle-setting, besides start and end points a third point of the circular arc is required (called a border point). The circular arc turns into a straight line when the border point is placed on the connection line between the start and endpoints.

Homing Motions: Most machining centers use the machine's zero position (also called reference return position or home position) as a point of reference for certain machine functions. Some machines require that one or more axes be sent to this position prior to activating a function. The Z-axis zero return position is often the **tool change** position for vertical machining centers. Typically, every machining center equipped with a pallet changer will require one or more axes to be at its zero return position prior to activating a **pallet change**. Since certain machine accessories require axes to be located at the zero return position prior to activation, CNC programmers must often command axes to go to this position.

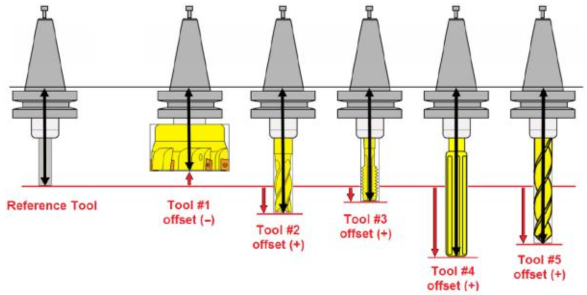
- **Return to Zero Position (G28)** With this function the tool can be moved to the zero point via an intermediate point with rapid traverse. G28 is a one-shot command, where X, Y, and Z refer to the intermediate point. Missing X, Y, and Z results in a direct motion to zero position.
- **Return from Zero Position (G29)** With this function the tool can be moved from zero point to the specified target point via an intermediate point with rapid traverse. G29 is a one-shot command, where X, Y, and Z refer to the target point. The

intermediate point is obtained from the previous G28 command.

Setting Position: The position statement in motion commands is affected by several modal settings. Distance mode defines if the coordinates are absolute (**G90**) or relative (**G91**). The measurement unit of the distances also can be chosen (**G20=inch, G21=mm**). Positions can be defined in Machine Coordinate System (MCS) or in Product Coordinate System (PCS). Some machines support processing multiple workpieces in one session. The part programmer doesn't need to know the position of the workpieces in the machine, because workpiece-holder (fixture) positions are stored in a data-table in the controller's memory (and the coordinate transformation can also be performed by the controller). This allows the programmer to design a shape in any PCS (**G54=PCS#1, G55=PCS#2 … G59=PCS#6**) or in MCS (**G53**).



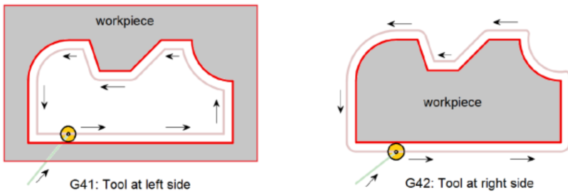
Tool Length Compensation (G43): For machining center applications, it would be very difficult for the programmer to predict the precise length of each tool used in the program. For this reason, the tool length compensation allows the programmer to ignore each tool's length as the program is written. The programmer can calculate with a reference tool length, or simply program the pure contour (i.e. motion of tool-tip instead of motion of TCP). At the time of setup, the machinist measures the length of each tool and stores the length offsets (the difference between the tool length assumed during programming and the actual tool length) into the controller's memory. With this data-table the controller can compensate the difference without changing the program. The length differences are all relative to the reference tool. Tools longer than the reference tool have positive offsets and tools shorter than the reference tool have negative offsets. The desired tool length offset can be found in the data-table by index (set by **H** address). **G44** is a rarely-used alternative to **G43**. It tells the control to begin applying tool length compensation, by subtracting the current length offset from the reference tool length. **G49** cancels any active tool length compensation.



Tool Radius Compensation (G41, G42): Just as tool length compensation allows the G-code programmer to forget about the tool's length, tool radius compensation allows the programmer to forget about the tool's radius as contours are programmed. While it may be obvious, let us point out that tool radius compensation is only used for milling cutters and only when milling on the periphery of the cutter. Therefore, this function is also called cutter radius compensation. You would never consider using cutter radius compensation for a drill, tap, reamer, or other hole machining tool.

There are several reasons why cutter radius compensation is very helpful to the CNC user. Without cutter radius compensation, machining center programmers must program the centerline path of all milling cutters. When programming the centerline path, the programmer must know the precise diameter of the milling cutter and calculate the program movements based on the tool's centerline path. With cutter radius compensation, the programmer can program the coordinates of the work surface, not the tool's centerline path. This eliminates the need for many calculations. If contours must be rough and finish milled, cutter radius compensation allows the programmer to use the same programmed coordinates needed to finish mill the workpiece to rough mill the workpiece. This keeps the programmer from having to calculate 2 sets of milling coordinates (one for roughing and one for finishing).

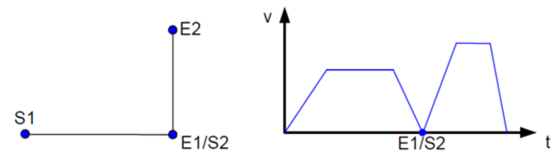
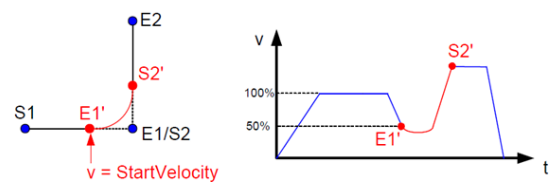
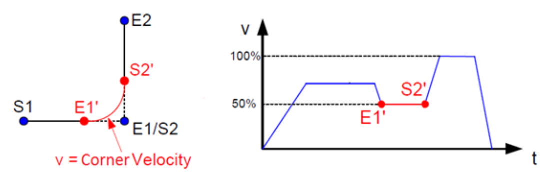
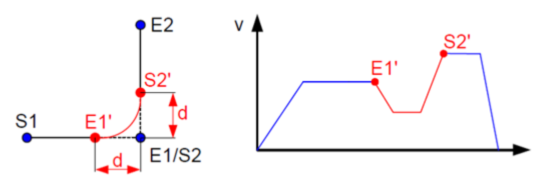
The desired tool radius can be found in the data-table by index (set by **D** address). If tool radius compensation is turned on, the tool may be on the left (**G41**) or the right (**G42**) of the cutting contour (in the feed direction). **G40** cancels any active tool radius compensation.

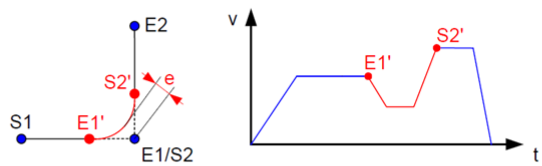
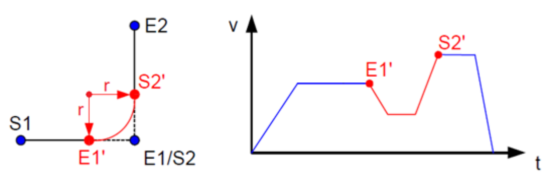
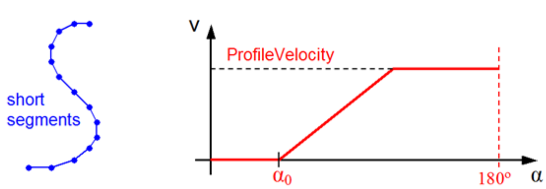


Tool Orientation: In simple machines the tool is always perpendicular to the product's surface. More advanced machines support higher degrees of freedom for tool motion. Besides the translational motion, the tool can roll left/right, pitch forth/back or yaw around its own axis. The desired orientation angles can be set by **Roll**, **Pitch** and **Yaw** parameters.

Path Mode (G61, G64): Consecutive motion commands compose a motion path. The connection of the individual motions can be performed in various ways. In **exact path mode** (**G61**) the motion is stopped at the end of each segment. This ensures exact path following, but the full stops can be harmful to the workpiece or the tooling, depending on the particular cut. In **continuous path mode** (**G64**) the motion never stops in the path, and the feed-rate is quasi-constant (will not stop, but can be reduced in the corners). Continuous motion can raise inaccuracy in path following (especially in sharp corners), but it can be handled by **transition modes** (see next section). Sometimes an exact stop may be needed even in continuous mode. It can be forced occasionally by **G09** (exact stop) or **G04** (dwell, i.e. stop+wait) one-shot commands.

Transition Modes: Connecting motion-segments may result in corners in the motion path. In order to avoid jerking, transition curves can be inserted in the corners. This results in a smoother motion, and a limited centripetal acceleration can be achieved by reducing the velocity in the transition curve. The desired transition mode can be chosen by **TransMode** keyword. **TransParam** and **TransAccel** are used as parameters. The following transition modes are available:

Description and Diagrams of Transition Modes (velocity profiles are simplified, in reality S-curves are used)
TransMode = 0 The motion segments are not modified and no transition curve is inserted. The motion stops in the corner. **TransParam** and **TransAccel** are ignored. 
TransMode = 1 **TransParam = StartVelocity** (in percentage of ProfileVelocity) This type of contouring is based on the deceleration of the original motion, as approaching the corner. The original segment is left at the position, where the velocity is decelerated to the specified value (the percentage value of the actual profile velocity). This velocity is applied as the starting velocity of the transition curve. **TransAccel** is ignored. 
TransMode = 2 **TransParam = CornerVelocity** (in percentage of ProfileVelocity) In this type of contouring, the radius of the transition curve is calculated in a manner, that the curve is traversed with a constant percentage value of the actual profile velocity, considering that centripetal acceleration is limited by **TransAccel** value. 
TransMode = 3 **TransParam = CornerDistance** In this type of contouring, the transition curve is specified by corner distance (the distance between the original corner point and the position where the original segment can be left). The velocity is calculated from curve radius, considering that centripetal acceleration is limited by **TransAccel** value. 

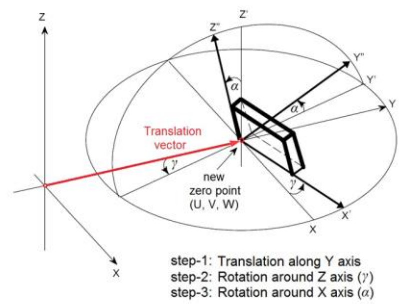
Description and Diagrams of Transition Modes (velocity profiles are simplified, in reality S-curves are used)
TransMode = 4 **TransParam = CornerDeviation** In this type of contouring, the transition curve is specified by corner deviation (the shortest distance between the original corner point and the transition curve). The velocity is calculated from curve radius, considering that centripetal acceleration is limited by **TransAccel** value. 
TransMode = 5 - 9 Reserved by PLC open
TransMode = 12 **TransParam = CornerRadius** In this type of contouring, the transition curve is specified directly by curve radius. The velocity is calculated from curve radius, considering that centripetal acceleration is limited by **TransAccel** value. 
TransMode = 13 **TransParam = CornerAngle** This transition mode can be used if the segments of the path are too short (for example curve approximation by lots of short lines). In this case, the transition curve insertion is limited by the path geometry. This transition mode does not insert transition curves, but the velocity in the corner can be modified depending on the angle of the corner (the sharper the corner, the lower the velocity). A reference angle is specified where at or below the angle, the corner velocity must be zero. Approaching 180 degrees, the corner velocity reaches the original profile value. The rise of the velocity curve is proportional to **TransAccel** value. 

Geometric Transformations: This function makes it easier to create programs. For example, to mill a shape into the workpiece in a sloped plane, we can design the shape in the XY plane, then it can be shifted and rotated to the desired place and orientation. The size of the programmed shape can be reduced or enlarged, or a mirror-image can also be created. All of these transformations are supported by some simple G-code commands, so the programmer doesn't have to recalculate the coordinates.

- **Relocation (G52):** This command changes the position and the orientation of the programmed figure by applying translation and up to 3 rotations. The components of the translation vector can be specified by X,Y, and Z addresses. The rotations are performed around X, Y and Z axes. Rotation angles are specified by A, B, and C addresses (measured in degrees). Any missing address denotes that the corresponding parameter is zero. The components of the entire transformation take place in a fixed order: beginning with the translation, then the rotation around Z, then rotation around Y, and lastly rotation around X.

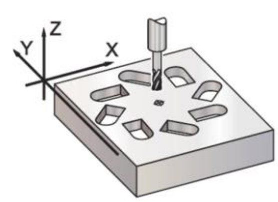
```
Relocation in a
single command:

G90 (absolute)
G52 Y80 Aα Cγ
```

```
Same relocation
in 3 steps:
G90 (absolute)
G52 Y80 (trans Y)
G91 (relative)
G52 Cγ (rot Z)
G52 Aα (rot X)
```



Relocation remains in effect until it is turned off. In absolute distance mode (G90), the G52 command deletes the current relocation and specifies a new one. In relative distance mode (G91), the G52 command specifies an incremental relocation, which is added to the current relocation. Since G52 is a one-shot command, there is no mate-command to cancel it. Nevertheless, the original location can be restored by G90 G52 with no parameters.

```
G90
G52 X100 Y-100 (set new origin)
Cycle8
{
  G90 (for absolute coords)
  G00 X-20 Y20
  G01 X-50 Y20
  G02 X-60 Y30 I0 J10
  G02 X-50 Y40 I10 J0
  G01 X-20 Y40
  G01 X-20 Y20
  G91 (for relative rotation)
  G52 C45 (rotation around Z)
}
G90 G52 (delete relocation)
```

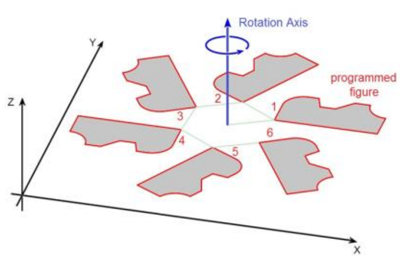


Milling the same shape repeatedly with rotation around Z axis in every step

- **Rotation around arbitrary axis (G68):** This command performs a single rotation on the programmed figure around an arbitrary axis. This axis is specified by a pivot point (X, Y, Z) and a vector (I, J, K). The angle of the rotation is specified by R address (measured in degrees). Any missing address denotes that the corresponding parameter is zero. The actual rotation remains in effect until it is turned off by the **G69** command. In absolute distance mode (G90), the G68 command deletes the current rotation and specifies a new one. In relative distance mode (G91), the G68 command specifies an incremental rotation, which is added to the current rotation (this can be effectively used in program-cycles: see the next examples).

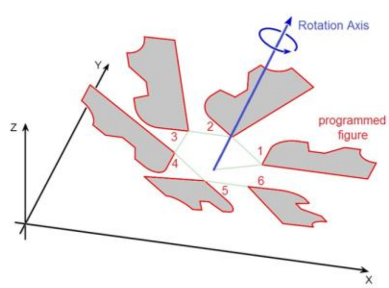
2D rotation: If the rotation-axis is perpendicular to the base-plane, then the rotated figure stays in the base-plane.

```
Cycle6
{
  G90 (absolute)
  G00 X40 Y20
  G01 X140
  G01 Y50
  G02 X120 Y70 J20
  G01 X105
  G01 X90 Y55
  G01 X75
  G01 X70 Y70
  G01 X60
  G03 X40 Y50 J-20
  G01 Y20
  G91 (relative)
  G68 I0 J0 K1 R60
}
```



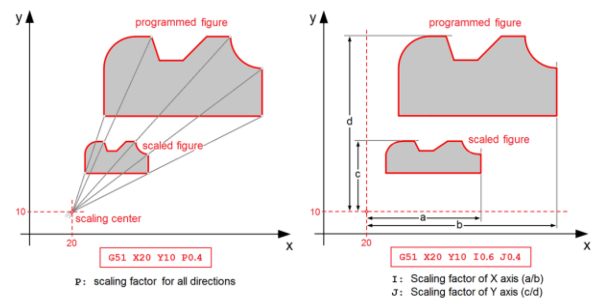
3D rotation: If the rotation-axis is not perpendicular to the base-plane, then the rotated figure emerges out of the base-plane.

```
Cycle6
{
  G90 (absolute)
  G00 X40 Y20
  G01 X140
  G01 Y50
  G02 X120 Y70 J20
  G01 X105
  G01 X90 Y55
  G01 X75
  G01 X70 Y70
  G01 X60
  G03 X40 Y50 J-20
  G01 Y20
  G91 (relative)
  G68 I1 J0 K2 R60
}
```

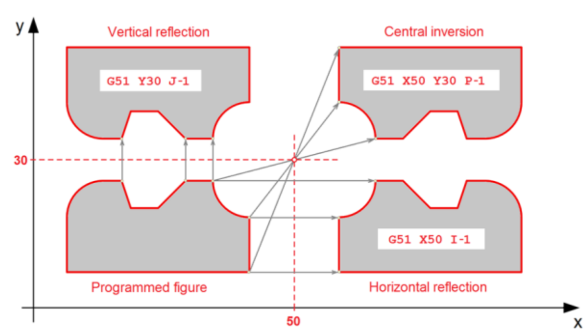


- **Scaling (G51):** With this function, the size of the programmed figure can be changed. If the scale factor (specified by P address) is greater than 1, then size is magnified. If the scale factor is less than 1, then size is reduced. Individual scaling factors for X, Y and Z directions also can be used (specified by I, J, and K addresses). There is always a point, which stays unchanged during this operation: the scaling center (specified by X, Y, Z addresses).

  If scaling is applied for only one direction, then not a single point, but an entire plane is unchanged. For example, if only J address is used (which means scaling in Y direction), the XZ plane stays unchanged. If scaling is applied for 2 directions, then a line will be unchanged. For example, if I and J addresses are used (which means scaling in X and Y direction), then all points of Z axis stays unchanged. Actual scaling remains in effect until it is turned off by the **G50** command.



- **Mirroring (G51):** If the previous command is applied with a scale factor of -1, then scaling turns into mirroring. If P address is used (with a value of -1), then a central inversion is performed. If I, J, or K addresses are used (with a value of -1), then reflection is done across a main plane (YZ, XZ or XY). An arbitrary reflection plane also can be used by rotation of the main planes (A, B, C addresses are used to rotate angles around X, Y, Z axes)



Transformations Quick Reference

Translations	
G52 X_	along X axis
G52 Y_	along Y axis
G52 Z_	along Z axis
G52 X_ Y_ Z_	along any vector

Rotations	
G52 A_	around X axis
G52 B_	around Y axis
G52 C_	around Z axis
G52 A_ B_ C_	around Z, Y, and X axes (in this order) by angles C, B, and A
G68 I_ J_ K_ R_	Around (I, J, and K) vector, by angle R

Central Scaling (P>0) or Mirroring (P=-1)	
G51 P_	Center point = zero point (0,0,0)
G51 P_ X_Y_Z_	Center point = any point (X,Y,Z)

Scaling (I>0) or Mirroring (I=-1) in X direction	
G51 I_	Base plane = main YZ plane (x=0)
G51 I_ X_	Base plane = any YZ plane (x=X)
G51 I_ X_ B_	Base plane = any plane sloping around Y axis
G51 I_ X_ C_	Base plane = any vertical plane

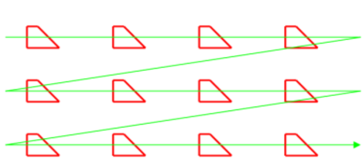
Scaling (J>0) or Mirroring (J=-1) in Y direction	
G51 J_	Base plane = main XZ plane (y=0)
G51 J_ Y_	Base plane = any XZ plane (y=Y)
G51 J_ Y_ A_	Base plane = any plane sloping around X axis
G51 J_ Y_ C_	Base plane = any vertical plane

Scaling (K>0) or Mirroring (K=-1) in Z direction	
G51 K_	Base plane = main XY plane (z=0)
G51 K_ Z_	Base plane = any XY plane (z=Z)
G51 K_ Z_ A_	Base plane = any plane sloping around X axis
G51 K_ Z_ B_	Base plane = any plane sloping around Y axis

Transformation Examples

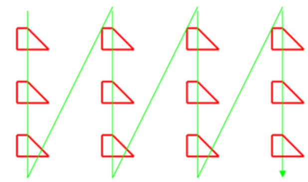
```
Cycle3
{
  Cycle4
  {
     G90 (absolute)
     G00 X0 Y0
     G01 X40
     G01 X15 Y25
     G01 X0
     G01 Y0
     G91 (relative)
     G52 X100
  }
  G52 X-400 Y60
}
```

Creating Matrix layout of a figure (horizontal order)



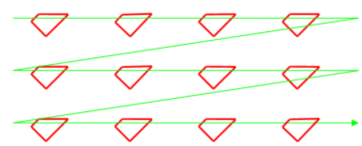
```
Cycle4
{
  Cycle3
  {
     G90 (absolute)
     G00 X0 Y0
     G01 X40
     G01 X15 Y25
     G01 X0
     G01 Y0
     G91 (relative)
     G52 Y60
  }
  G52 X100 Y-180
}
```

Creating Matrix layout of a figure (vertical order)



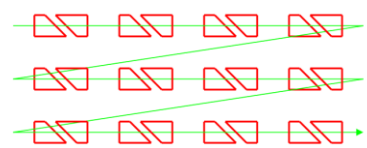
```
Cycle3
{
  Cycle4
  {
     G52 C45
     G90 (absolute)
     G00 X0 Y0
     G01 X40
     G01 X15 Y25
     G01 X0
     G01 Y0
     G91 (relative)
     G52 C-45
     G52 X100
  }
  G52 X-400 Y60
}
```

Creating Matrix layout of a figure with rotation of 45 degrees



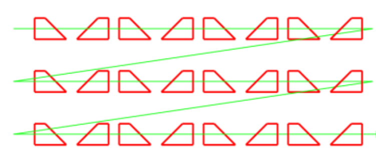
```
Cycle3{
  Cycle4{
     Cycle2{
       G90 (absolute)
       G00 X0 Y0
       G01 X40
       G01 X15 Y25
       G01 X0
       G01 Y0
       G91 (relative)
       G51 P-1 X30 Y12
     }
     G52 X100
  }
  G52 X-400 Y60
}
```

Creating Matrix layout of a figure with central inversion



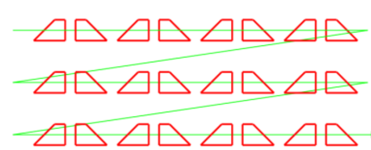
```
Cycle3{
  Cycle4{
     Cycle2{
       G90 (absolute)
       G00 X0 Y0
       G01 X40
       G01 X15 Y25
       G01 X0
       G01 Y0
       G91 (relative)
       G51 I-1 X50
     }
     G52 X100
  }
  G52 X-400 Y60
}
```

Creating Matrix layout of a figure and its mirror-image



```
Cycle3{
  Cycle4{
     Cycle2{
       G51 I-1 X-4
       G90 (absolute)
       G00 X0 Y0
       G01 X40
       G01 X15 Y25
       G01 X0
       G01 Y0
       G91 (relative)
     }
     G52 X100
  }
  G52 X-400 Y60
}
```

Creating Matrix layout of the mirror image and the mirror-mirror-image of a figure

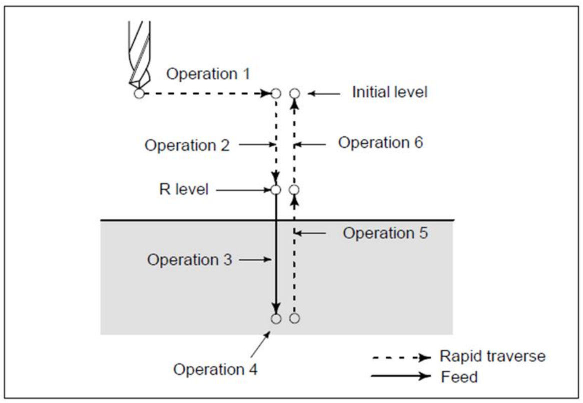


Canned Cycles

Canned cycles offer a way of conveniently performing repetitive CNC machine operations. With a canned cycle, a frequently-used machining function (such as drilling, boring, threading, etc.) can be specified in a single G-command. Without canned cycles, normally more than one block is required. In addition, the use of canned cycles can shorten the program to save memory.

A canned cycle consists of a sequence of the following operations:

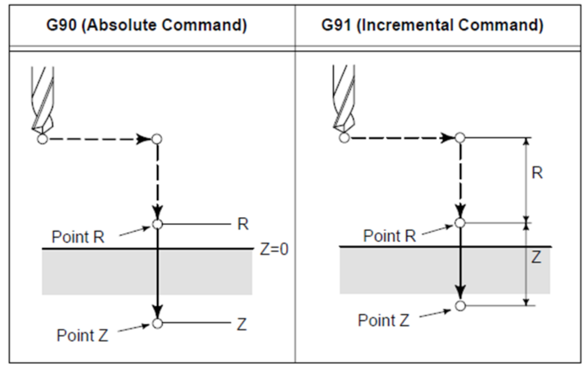
1. Positioning of axes X and Y (also including another axis)
2. Rapid traverse to R level (retract-point)
3. Hole machining
4. Operation at the bottom of a hole
5. Retraction to point R level (retract-point)
6. Rapid traverse up to the initial point



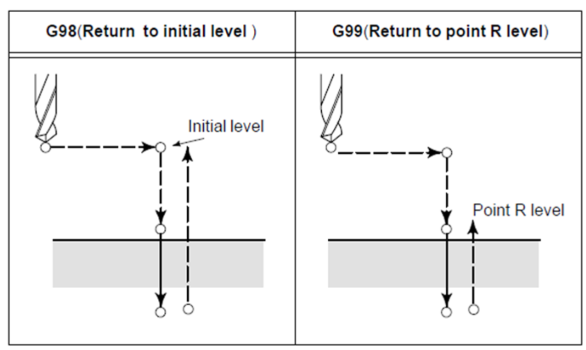
All canned cycles are performed with respect to the currently selected plane. Any of the three basic planes (XY, YZ, ZX) may be selected (G17, G18 or G19 commands). Throughout this section, most of the descriptions and figures assume that the XY-plane has been selected. The behavior is analogous in the case of YZ or XZ planes. The positioning (first operation of the canned cycle) is performed in the selected plane. The drilling axis is always perpendicular to the selected plane.

G-code	Positioning Plane	Drilling Axis
G17	X-Y plane	Z
G18	Z-X plane	Y
G19	Y-Z plane	X

All canned cycles use X, Y, Z and R addresses as parameters to determine X, Y, Z and R positions. The R (usually meaning retract) position is along the actual drilling axis (e.g. Z-axis for XY-plane). Some canned cycles use additional arguments. In absolute distance mode (G90), the X, Y, Z, and R values are absolute positions in the current coordinate system. In incremental distance mode (G91), the X, Y, Z and R values are relative to the previous position.



When the tool reaches the bottom of a hole, the tool may be returned to point R or to the initial level. These operations are specified with G98 and G99. The following illustrates how the tool moves when G98 or G99 is specified. Generally, G99 is used for the first drilling operation and G98 is used for the last drilling operation. The initial level does not change even when drilling is performed in the G99 mode.



Canned cycles are modal G codes and remain in effect until canceled. While in effect, the current state is the drilling mode. Once drilling data is specified in the drilling mode, the data is retained until modified or canceled. Specify all necessary drilling data at the first canned cycle; when canned cycles are being performed, specify data modifications only. Example:

```
M4 S1000        (Spindle starts rotating)
G81 G90 G99 X50 Y80 Z-40 R10 F200 (drilling hole#1, return to R)
G81 X150        (drilling hole#2, then return to R level)
G81 Y180        (drilling hole#3, then return to R level)
G81 G98 X50     (drilling hole#4, then return to the initial level)
G80 X0 Y0       (cancel drilling mode, return to home position)
M5              (Spindle stops rotating)
```

Canned cycles are usually permanently stored as a pre-program in the machine's controller and cannot be altered by the user. The following machining operations are supported:

- **Drilling** is a cutting process that uses a drill bit to cut a hole of circular cross-section in solid materials. The drill bit is usually a rotary cutting tool, often multipoint. The bit is pressed against the workpiece and rotated at rates from hundreds to thousands of revolutions per minute. This forces the cutting edge against the workpiece, cutting off chips (swarf) from the hole as it is drilled.

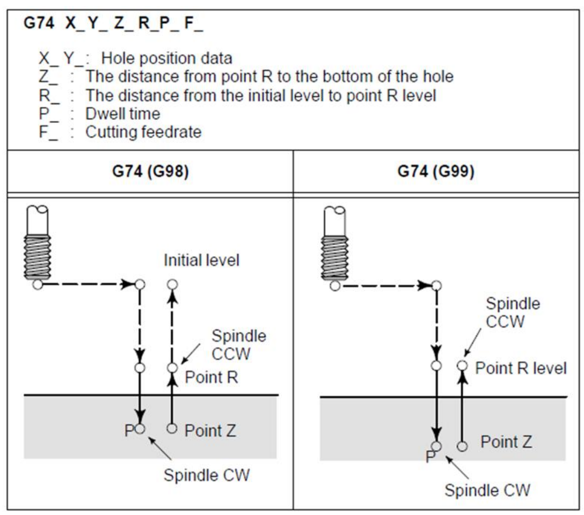
- **Boring** is the process of enlarging a hole that has already been drilled (or cast), by means of a single-point cutting tool (or of a boring head containing several such tools). Boring is used to achieve greater accuracy in the diameter of a hole, and can be used to cut a tapered hole. Boring can be viewed as the internal-diameter counterpart to turning, which cuts external diameters. There are various types of boring. **Backboring** is the process of reaching through an existing hole and then boring on the "back" side of the workpiece (relative to the machine headstock).

- **Threading** is the process of creating a screw thread. There are many methods of generating threads. Threads may be milled with a rotating milling cutter if the correct helical toolpath can be arranged. This was formerly arranged mechanically, and it was suitable for mass-production work. With the widespread dissemination of affordable, fast, precise CNC, it became much more common, and today internal and external threads are often milled. Some advantages of thread milling, as compared to single-point cutting, are faster cycle times, less tool breakage, and that a **left-** or **right-hand** thread can be created with the same tool.

G-code	Operation on the bottom of the hole	Retraction Feed	Application
G73*	-	rapid traverse	High-speed peck drilling (deep hole drilling)
G74	dwell, spindle CW	feed-rate	Counter-clockwise thread cutting (left-hand tapping)
G76*	oriented spindle stop	rapid traverse	Fine drilling
G81	-	rapid traverse	Simple drilling
G82	dwell	rapid traverse	Simple drilling with dwell
G83*	-	rapid traverse	Peck drilling (deep hole drilling)
G84	dwell, spindle CCW	feed-rate	Clockwise thread cutting (right-hand tapping)
G85	-	feed-rate	Boring, feed out
G86	spindle stop	rapid traverse	Boring, spindle stop, rapid out
G87*	spindle CW	rapid traverse	Back boring
G88	dwell, spindle stop	manual	Boring with dwell, spindle stop, manul out
G89	dwell	feed-rate	Boring with dwell, feed out

*Not implemented yet

- **G74** This cycle performs left-handed tapping. After positioning along X and Y axes, rapid traverse is performed to point R, then point Z is reached with the spindle rotating counter- clockwise. Then the spindle rotation is changed to clockwise for retraction. This creates a reverse thread. Before specifying G74, use the M4 command to rotate the spindle counter clockwise.

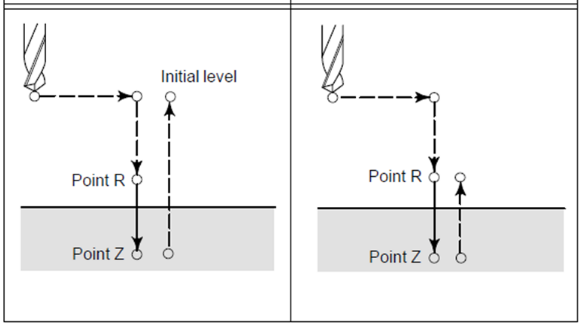


- **G81** This cycle is used for normal drilling. After positioning along X and Y axes, rapid traverse is performed to point R, then drilling is performed to point Z (feedrate=F). The tool is then retracted in rapid traverse. Before specifying G81, use M3/M4 to rotate the spindle.

G81 X_ Y_ Z_ R_ F_

X_ Y_: Hole position data
Z_ : The distance from point R to the bottom of the hole
R_ : The distance from the initial level to point R level
F_ : Cutting feedrate

G81 (G98)	G81 (G99)

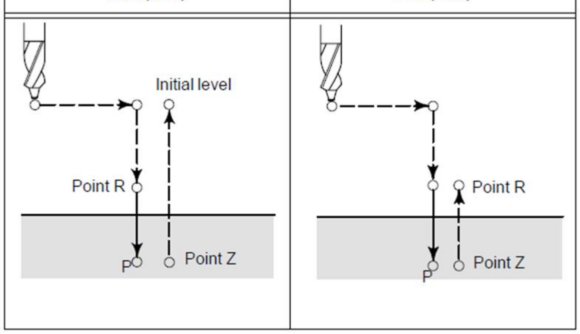


- **G82** This cycle is used for normal drilling with dwell. After positioning along X and Y axes, rapid traverse is performed to point R, then drilling is performed to point Z (feedrate=F). When the bottom of the hole has been reached, a dwell is performed (time=P). The tool is then retracted in rapid traverse. Before specifying G82, use M3/M4 to rotate the spindle.

G82 X_ Y_ Z_ R_ P_ F_

X_ Y_: Hole position data
Z_ : The distance from point R to the bottom of the hole
R_ : The distance from the initial level to point R level
P_ : Dwell time at the bottom of a hole
F_ : Cutting feed rate

G82 (G98)	G82 (G99)

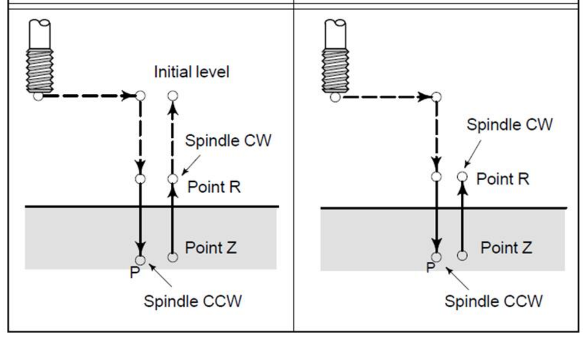


- **G84** This cycle performs normal tapping. After positioning along X and Y axes, rapid traverse is performed to point R, then the point Z is reached with the spindle rotating clockwise. Then the spindle rotation is changed to counter-clockwise for retraction. This operation creates threads. Before specifying G84, use M3 to rotate the spindle clockwise.

G84 X_ Y_ Z_ R_ P_ F_

X_ Y_: Hole position data
Z_ : The distance from point R to the bottom of the hole
R_ : The distance from the initial level to point R level
P_ : Dwell time
F_ : Cutting feedrate

G84 (G98)	G84 (G99)

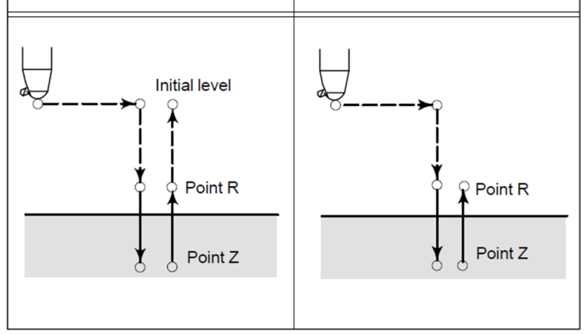


- **G85** This cycle is used to bore a hole. After positioning along X and Y axes, rapid traverse is performed to point R, then boring is performed to point Z (feedrate=F). The tool is then retracted in feedrate (without stopping the spindle). Before specifying G85, use M3/M4 to rotate the spindle.

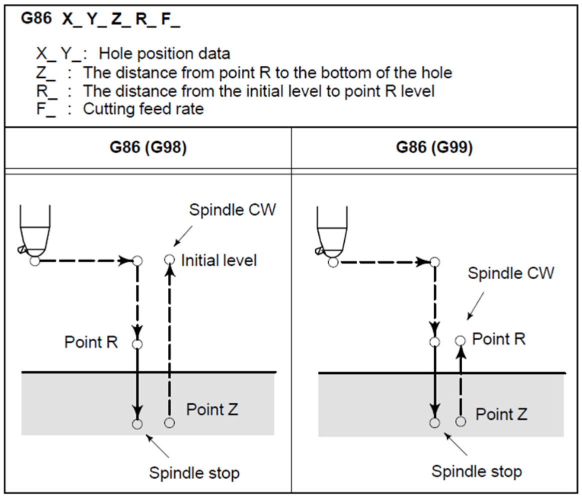
G85 X_ Y_ Z_ R_ F_

X_ Y_: Hole position data
Z_ : The distance from point R to the bottom of the hole
R_ : The distance from the initial level to point R level
F_ : Cutting feed rate

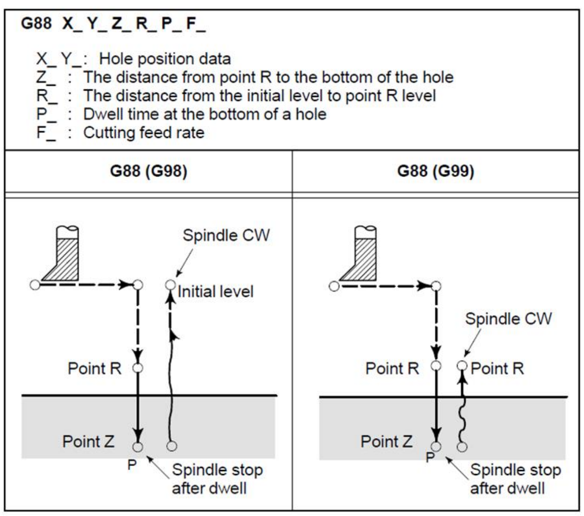
G85 (G98)	G85 (G99)



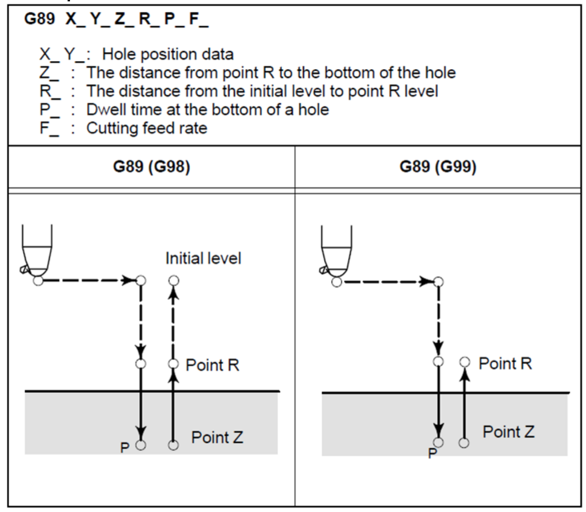
- **G86** This cycle is used to bore a hole. After positioning along X and Y axes, rapid traverse is performed to point R, then boring is performed to point Z (feedrate=F). Then the spindle is stopped at the bottom of the hole and the tool is retracted in rapid traverse. Before specifying G86, use M3/M4 to rotate the spindle.



- **G88** This cycle is used to bore a hole with dwell. After positioning along the X and Y axes, rapid traverse is performed to point R, then boring is performed to point Z (feedrate=F). When boring is completed, a dwell is performed (time=P), then the spindle is stopped. The tool is manually retracted from the bottom of the hole. At point R, the spindle rotation is started again and rapid traverse is performed to the initial level. Before specifying G88, use M3/M4 to rotate the spindle.



- **G89** This cycle is used to bore a hole. After positioning along X and Y axes, rapid traverse is performed to point R, then boring is performed to point Z (feedrate=F). When boring is completed, a dwell is performed (time=P). The tool is then retracted in feedrate (without stopping the spindle). It is almost the same as G85, except for the dwell function. Before specifying G89, use M3/M4 to rotate the spindle.



Most Frequently Used M-commands

M-commands (or M-codes) are also called **miscellaneous commands**. Most of these deal with overall Machine functions, such as tool changing, part clamping, coolant flow control, spindle control, input/output handling, etc. A few M-codes manage program execution (stopping, ending). Different machine tools may use the same code to perform different functions. Just like the G-codes, depending on the particular machine's implementation, any number of M-codes may be implemented in C&M by simply creating the proper Function Block Diagram.

Program Stopping/Ending: A running program can be stopped temporarily or permanently. A temporarily stopped program can be restarted by the user (e.g. external restart button). In this case, the program proceeds at the next line. If a program stop is programmed within a continuous motion path, then it closes the path (the motion stops at the end of the latest segment). The next motion is the beginning of a new path.

- **M00** stops a running program temporarily
- **M01** stops a running program temporarily, but only if the optional stop switch is on

- **M02** ends the program and restores the default settings of modal groups and default value of A-Z parameters
- **M30** is the same as M02, but extended with pallet change
- **M60** is the same as M00, but extended with pallet change

Spindle Control: In machine tools, a spindle is a rotating axis of the machine. For example, a shaft to which the tool (such as a drill bit or milling cutter) is attached, or shaft of the headstock on a lathe. Spindle speed is specified by **S** address, which takes effect with the following codes:

- **M3** starts the spindle turning clockwise
- **M4** starts the spindle turning counter-clockwise
- **M5** stops the spindle from turning.

Tool-change: The identifier of the selected tool can be specified by **T** address. The tool-change process can be triggered by the M06 command. If the spindle is turning, M06 stops it. If tool-change is inserted within a continuous motion path, then M06 closes this path (the motion stops at the end of the latest segment). The next motion is the beginning of a new path.

Coolant Control: Sometimes the manufacturing process needs cooling, to prevent the workpiece and the tool from overheating. It can be managed by the following commands.

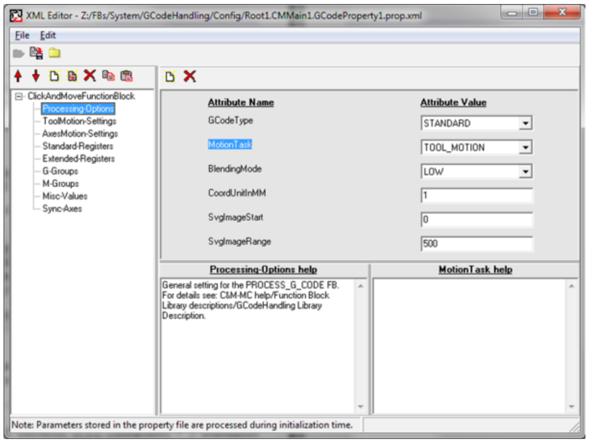
- **M7** turns mist coolant on
- **M8** turns flood coolant on
- **M9** turns all coolant off

Digital IO Control: Digital outputs can be programmed by these commands. Each output bit can be set/reset individually. The index of the selected bit can be specified by the **P** address. The digital output appears in **DigOut** column of MotionTask as a hexadecimal value.

- **M62** sets the selected bit to 1
- **M63** sets the selected bit to 0

G-code Processing Properties

Since Click&Move supports G-codes (as a motion description and input for the user projects), G-code processing is performed by a Click&Move Function Block (called PROCESS_G_CODE). This function block can be configured by **GCodeProperty** (Project Menu / Open XML property file editor).



<u>Processing-Options:</u> The main settings can be found here.

- **GCodeType** option allows selection of the type of the G-code file (considered as the input of the processing). STANDARD and EXCELLON can be selected.

- **MotionTask** option allows selection of the type of the MotionTask database (considered as the output of the processing). The following 2 options can be chosen:

  (1) **AXES_MOTION:** The StartPoint and EndPoint columns of the MotionTask database (Prog.xdb file) contain the pure axes-positions (not geometrical x-y-z coordinates!). The StartPoint and EndPoint contain as many elements as the number of the axes. The Axes can be selected in the **AxesMotion-Settings** Menu.

(2) **TOOL_MOTION:** The StartPoint and EndPoint columns of the MotionTask database contain geometrical x-y-z coordinates. In this case MotionTask database consists of 2 files (Prog.xdb and Tool.xdb):

- The **Prog.xdb** describes the programmed path (the geometrical shape of the workpiece). The StartPoint/EndPoint always contains 3 elements (x,y,z coordinates)

- The **Tool.xdb** describes the motion of the tool (including Tool-orientation, Tool- Length compensation, Tool-Radius compensation, Transition curves in the corners ...etc). The StartPoint/EndPoint always contains 6 elements (x,y,z coordinates + 3 orientation angles). Further settings that affect Tool.xdb, can be found in the **ToolMotion-Settings** Menu.

- **BlendingMode** is the method of velocity setting in joints of motion segments. It is used in continuous path mode (G64). The velocity in the joint of 2 motion segments equals to:

  (1) **OFF**: zero (no blending: the motion stops at the end of each segment)
  (2) **PREV**: velocity of the previous segment
  (3) **NEXT**: velocity of the next segment
  (4) **LOW**: the lower velocity from the previous and next
  (5) **HIGH**: the higher velocity from the previous and next

- **Coord Unit In mm** specifies the length unit (measured in mm) in the output database. For example: if 25.4 is set, then x-y-z coordinates in MotionTask database will appear in inches.

ToolMotion-Settings: Detailed settings of TOOL_MOTION mode can be found here

- **OrientationMode** allows selection from various methods of tool-orientation calculations. Tool-orientation is the orientation of the coordinate-system attached to the tool. It is called the Tool Coordinate System (TCS). The origin of TCS is attached to the Tool Center Point (TCP). The Z-axis of TCS is attached to the longitudinal axis of the tool. The direction of feed motion usually designates the X axis of the TCS. The orientation of TCS can be described by orientation angles (rotations around Z, Y and X in this order).

  (1) **FIXED**: orientation angles are constant zero (no corner curves are generated).
  (2) **PLANAR**: Z-axis of TCS equals to the normal vector of the base-plane
  (3) **SURFACE**: Z-axis of TCS equals to the normal vector of the surface
  (4) **SPATIAL**: The motion is not bound to a base plane or a surface. But 2 connecting motion segments define a local plane in each corner, which designates Z-axis of TCS.

- **CirclePrepare** allows conversion of all type of circles to center-point type
- **ConvertToMCS** allows calculation of all coordinates to Machine Coordinate System

AxisMotion-Settings: The axes for AXIS_MOTION mode can be selected here. The direct axes can be sorted into 2 groups: main axes and auxiliary axes. While main axis positions are changing, the motion segments create a continuous path. If only auxiliary axis positions are changing, then the previous motion stops (the path has been closed) before auxiliary axes start to move. Groups of main axes can be designated by the MainAxes option. For example MainAxes=2 means, that Axis1 and Axis2 will act as main axes, the others (Axis3, Axis4 ...etc) are auxiliary axes.

Standard-Parameters: Default values of standard (A-Z) addresses can be specified here.

Extended-Parameters: Default values of C&M specific parameters can be specified here.

G-groups: Default values of G-groups can be specified here.

M-groups: Default values of M-groups can be specified here.

Misc-Values: Misc columns of MotionTask table can be specified here (the current value of any address or modal group can be displayed in a Misc column).

Sync-Axes: The axes that are intended to move synchronously to the main axis motion can be selected here. The positions of these axes will appear in the SyncAxes column of MotionTask database.

Standard Parameters (Letter-codes)

G-code Address	Description
A	Absolute or incremental position of A axis (rotational axis around X axis)
B	Absolute or incremental position of B axis (rotational axis around Y axis)
C	Absolute or incremental position of C axis (rotational axis around Z axis)
D	Index in tool-table, which defines diameter or radial offset for G41 and G42 commands. Also used for depth of cut on lathes.
F	Defines feed rate (the velocity at which the tool is advanced along the workpiece, CoordUnit/minute)
G	One-shot G-commands (not to be confused with modal G-commands). Most of G- commands cause the machine to change from one mode to another, and the mode stays active until some other command changes it. These G commands are arranged in sets called "modal groups". Detailed description can be found below in the modal-group table. The values of G-commands are only loaded into the G address if they are not recognized as modal. For example: G04, G09, G28, G30, G52, G53, G92.
H	Index in tool-table, which defines tool length offset for G43 and G44 commands.
I	Defines arc center in X axis for G02/G03 arc commands. Also used as transformation parameters for G51 and G68.
J	Defines arc center in Y axis for G02/G03 arc commands. Also used as transformation parameters for G51 and G68.
K	Defines arc center in Z axis for G02/G03 arc commands. Also used as transformation parameters for G51 and G68.
M	One-shot M-commands (not to be confused with modal M-commands). Most of M- commands cause the machine to change from one mode to another, and the mode stays active until some other command changes it. These M-commands are arranged in sets called "modal groups". Detailed description can be found below in the modal-group table.
P	Serves as parameter for various G and M codes. Used as dwell time for G04 or some canned cycles. Used as scaling transformation parameter for G51. Also used in the calling and termination of subprograms (with M98, it specifies which subprogram to call; with M99, it specifies which block number of the main program to return to).
R	Defines arc`s radius (G02/G03). Define retract level in canned cycles (G73-G84). Also used as rotation angle for G68.
S	Defines speed, either spindle speed or surface speed depending on mode (not to be confused with feedrate)
T	Tool selection. This address contains the tool number for the tool change command (M06)
U	Secondary linear axis, parallel to X axis.
V	Secondary linear axis, parallel to Y axis.
W	Secondary linear axis, parallel to Z axis.
X	Absolute or incremental position of X axis. Primary linear axis. Also used as transformation parameters for G51, G52, G68 and offset for G92.
Y	Absolute or incremental position of Y axis. Primary linear axis. Also used as transformation parameters for G51, G52, G68 and offset for G92.
Z	Absolute or incremental position of Z axis. Primary linear axis. Also used as transformation parameters for G51, G52, G68 and offset for G92.

Extended Parameters (C&M Specific)

Keyword	Data Type	Description
Cycle	UInt32	Repeat number for program cycle. The cyclic program block begins with "{" and closed by "}"
Roll	Float64	Tool rotation around longitudinal axis
Pitch	Float64	Tool rotation around lateral axis
Yaw	Float64	Tool rotation around vertical axis
Radius	Float64	Tool radius value for tool radius compensation (G41, G42)
Length	Float64	Tool length value for tool length compensation (G43, G44)
TransMode	Float64	Transition Mode selection (See description of transition modes above)
TransParam	Float64	Main parameter of transition (See description of transition modes above)
TransAccel	Float64	Maximal allowed centripetal acceleration in the transition curve. Not used if TransMode = 0 or 1
Accel	Float64	Acceleration along the path
Decel	Float64	Deceleration along the path
Jerk	Float64	Jerk along the path

Modal Groups of G-commands

Name of Group	Value	Description	Affected Params
Motion Mode	0	Rapid (Point-to-point) positioning. Only the end point (X, Y, Z) of the motion is specified, the path is irrelevant. The axes can all move independently, their motions are	X,Y,Z

Name of Group	Value	Description	Affected Params
		not coordinated; typically each axis moves at its maximum velocity.	
	1	Linear interpolated motion. The most common workhorse code for feeding during a cut. The end point (X, Y, Z) of the motion is specified and the control automatically calculates (interpolates) the intermediate points to pass through that will yield a straight line (hence "linear").	X,Y,Z
	1.2*	Second degree interpolation (not yet implemented)	X,Y,Z
	1.3*	Third degree interpolation (not yet implemented)	X,Y,Z
	2	Circular interpolated motion (Clockwise direction). In addition to the arc`s end-point (X, Y, Z) either the arc`s center-point (I, J, K) or arc`s radius (R) and plane (G17/G18/G19) must be specified. The motions of the involved axes are coordinated so that a perfectly circular path is described for the arc. Center point coordinates are relative to the start point!	X,Y,Z, I,J,K,R
	3	Circular interpolated motion (Counter-Clockwise direction). In addition to the arc`s end-point (X, Y, Z) either the arc`s center-point (I, J, K) or arc`s radius (R) and plane (G17/G18/G19) must be specified. The motions of the involved axes are coordinated so that a perfectly circular path is described for the arc. Center point coordinates are relative to the start point!	X,Y,Z, I,J,K,R
	2.1	Circular interpolated motion with radius vector (Clockwise direction). The user defines the end point (X, Y, Z) and the perpendicular vector of the circle plane (I, J, K) according to the rule of right thumb. The length of this vector equals to the arc`s radius	X,Y,Z, I,J,K
	3.1	Circular interpolated motion with radius vector (Counter-clockwise direction). The user defines the end point (X, Y, Z) and the perpendicular vector of the circle plane (I, J, K) according to the rule of right thumb. The length of this vector equals to the arc`s radius	X,Y,Z, I,J,K
	2.2	Circular interpolation with Border Point (Clockwise direction). The user defines the arc`s end point (X, Y, Z) and a border point (I, J, K) on the arc. The motions of the involved axes are coordinated so that a perfectly circular path is described for the arc. Border point coordinates are relative to the start point!	X,Y,Z, I,J,K
	3.2	Circular interpolation with Border Point (Counter-clockwise direction). The user defines the arc`s end point (X, Y, Z) and a border point (I, J, K) on the arc. The motions of the involved axes are coordinated so that a perfectly circular path is described for the arc. Border point coordinates are relative to the start point!	X,Y,Z, I,J,K
Lathe Mode	7*	Diameter mode	-
	8*	Radius mode	-
Polar Coords	15*	Polar coordinates OFF	-
	16*	Polar coordinates ON	-
Plane	17	XY plane selection (used with G02/G03 and for canned cycles)	-
	18	ZX plane selection (used with G02/G03 and for canned cycles).	-
	19	YZ plane selection. (used with G02/G03 and for canned cycles)	-
Coord Unit	20	inch	-
	21	mm	-
Working Area	22*	Working area limitation, protection zone ON	-
	23*	Working area limitation, protection zone OFF	-
Tool Radius Compen-sation	40	Tool radius compensation OFF (Cancels G41 or G42)	-
	41	Tool radius compensation to left of contour. The value of compensation can be loaded from ToolRadius database by reference (D) or can be specified directly (Radius)	D, Radius
	42	Tool radius compensation to right of contour. The value of compensation can be loaded from ToolRadius database by reference (D) or can be specified directly (Radius)	D, Radius
Tool Length Compen-sation	43	Tool length offset compensation positive. The value of compensation can be loaded from ToolLength database by reference (H) or can be specified directly (Length). G43 is the commonly used version (vs. G44).	H, Length
	44	Tool length offset compensation negative. The value of compensation can be loaded from ToolLength database by reference (H) or can be specified directly (Length). G44 is the seldom-used version (vs. G43)	H, Length

Name of Group	Value	Description	Affected Params
	49	Tool length compensation OFF. (Cancels G43 or G44)	-
Scaling	50	Scaling OFF	-
	51	Scaling ON. Either axial scaling factors (I, J, K) or one general scaling factor (P) must be specified. The center point of the scaling can be shifted (U, V, W). The scaling axes can be rotated (A, B, C). Scaling results mirroring, if the scaling factor is -1.	I,J,K,P, U,V,W A,B,C
Coord System	54	Product Coordinate System #1	-
	55	Product Coordinate System #2	-
	56	Product Coordinate System #3	-
	57	Product Coordinate System #4	-
	58	Product Coordinate System #5	-
	59	Product Coordinate System #6	-
Path Control	61	Exact path mode (may temporarily stop at corners)	-
	62*	Automatic corner override (feedrate reduction at corners). It can only be used effectively in conjunction with Radius Compensation (G41, G42) (not yet implemented)	-
	63*	Tapping mode (feedrate override is not allowed, the tool does not decelerate at corners) (not yet implemented)	-
	64	Continuous path mode (constant feedrate)	-
Macro Call	66*	Modal macro call (not yet implemented)	-
	67*	Modal macro call cancel (not yet implemented)	-
Rotation	68	Rotation around arbitrary axis. Rotation axis is defined by pivot point (U, V, W) and rotation vector (I, J, K). The angle of rotation is specified by R.	U,V,W,I, J,K,R
	69	Rotation OFF	-
Canned Cycle	73*	Deep hole drilling with chip breaking (High-speed peck drilling, no full retraction from pecks. PeckDepth=Q).	X,Y,Z,R, F,Q
	74	Counter-clockwise tapping (left-hand tapping, DwellTime=P).	X,Y,Z,R, F,P
	76*	Fine drilling (DwellTime=P, ToolShift=Q)	X,Y,Z,R, F,P,Q
	80	Canned cycle cancel	-
	81	Simple drilling	X,Y,Z,R,F
	82	Simple drilling with dwell (DwellTime=P)	X,Y,Z,R, F,P
	83*	Deep hole drilling with swarf removal (Peck drilling, full retraction from pecks. PeckDepth=Q).	X,Y,Z,R, F,Q
	84	Clockwise tapping (right-hand tapping. DwellTime=P).	X,Y,Z,R, F,P
	85	Boring, feed out (retraction with G01)	X,Y,Z,R,F
	86	Boring, spindle stop, rapid out (retraction with G00)	X,Y,Z,R,F
	87*	Back boring (reverse countersinking, DwellTime=P, ToolShift=Q).	X,Y,Z,R, F,P,Q
	88	Boring with dwell, spindle stop, manual out. (DwellTime=P)	X,Y,Z,R, F,P
	89	Boring with dwell, feed out (retraction with G01, DwellTime=P).	X,Y,Z,R, F,P
Distance Mode	90	Absolute programming. Positioning defined with reference to part zero. New transformations (G51, G68) recreate the actual transformation matrix (previous transformations are deleted)	-
	91	Incremental programming. Positioning defined with reference to previous position. New transformations (G51, G68) are added to actual transformation matrix (previous transformations are kept)	-
Feedrate Mode	93*	Inverse-time feedrate (rpm) (not yet implemented)	-
	94*	Feed per minute (mm/min, inch/min) (not yet implemented)	-
	95*	Feed per rotation (mm/rev, inch/rev) (not yet implemented)	-
Speed Control	96*	Constant surface speed control ON (not yet implemented)	-
	97*	Constant surface speed control OFF (not yet implemented)	-
Return Mode	98	Return to initial point in canned cycles	-
	99	Return to retract point (R) in canned cycles	-

*Recognized by C&M but not yet implemented

One-shot G-code Commands

Command	Description
M06	Tool Change (tool is selected by T)
G04	Dwell (stopping and waiting). Waiting time is specified by P

G09	Exact Stop (no parameters)
G52	Set local coordinate system (translation: X,Y,Z and rotation: A,B,C)
G53	Selecting MCS (Machine Coordinate System)
G92	Set Coordinate Offsets (X, Y, Z, U, V, W … etc)

Modal Groups of M-commands

All M modal groups are user-defined but some of them (e.g. the first three groups) are usually used for specific tasks described below.

Name of Group	Description	Affected Params
M0_M1_M2_M30	**M00:** Compulsory Stop	-
	M01: Optional Stop	-
	M02: End of program	-
	M30: End of program and return to program top	-
M3_M4_M5_M19	**M03:** Start the spindle clockwise at the specified speed (S)	S
	M04: Start the spindle counter-clockwise at the specified speed (S)	S
	M05: Stop the spindle	-
	M19: Set the spindle to the specified orientation (R)	P, Q, R
M7_M8_M9	**M07:** Turn mist coolant ON	-
	M08: Turn flood coolant ON	-
	M09: Turn all coolants OFF	-
M10-19	**M10:** Pallet clamp ON	-
	M11: Pallet clamp OFF	-
M20-29	-	-
M30-39	-	-
M40-49	**M48:** Spindle speed and feedrate override ENABLED	-
	M49: Spindle speed and feedrate override DISABLED	-
M50-59	-	-
M60-69	**M62-M68:** Input-Output control	P, Q, E
M70-79	-	-
M80-89	-	-
M90-99	**M97:** Local sub-routine CALL	P
	M98: Sub-program CALL	P, Q, L
	M99: Sub-program RETURN	P, L
M100-199	-	-
M200-299	-	-
M300-399	-	-
M400-499	-	-
M500-599	-	-
M600-699	-	-
M700-799	-	-
M800-899	-	-
M900-999	-	-