TOMATECH 9 Series

CNC Programming Manual

Basic Information

This Manual is written by TOMATECH (SHENZHEN) TECHNOLOGY CO.,LTD.

This Manual is mainly written by Tang Xiaobing, Yao Lei and Xu Yuwen.

This manual was typeseted on July 21, 2018, and the project number is applicable for all the machine control projects of CNC9 series.

TOMATECH (SHENZHEN) TECHNOLOGY CO.,LTD.

Email: info@tomatech-cnc.com

Whats app/ Mob.: +86 18617165968

Precautions and Notes

※Transport and storage

Do not exceed six layers for products packing cases piling

Forbid to climb, stand, or place heavy items on products packing cases

Do not use the cable connecting with the product to drag or move products

Non-collision, non-scratching on the panel and display screen

Prevent the moisture, exposure and rain affected packing cases

※Open Package Inspection

Confirm products after opening package

Check whether damages exist during the transportation

Confirm whether parts are complete or have damages by comparison with list

Contact us promptly if products models are inconsistent, parts are missed, or damages during shipping are found etc.

※Wiring

Professional personnel with corresponding capability is must for participation in wiring and inspection

Reliable grounding is must for products, with less than 4Ω ground resistance, and neutral wire (neutral wire) is not allowed to substitute grounding wire

Wiring should be correct and secure, to avoid the consequences of product failure or unexpected outcome

Surge absorption diode connecting with product should be linked upon the stipulated direction; otherwise the product may be damaged

The power supply of the product should be cut before plug-in & plug-out or opening the cabinet.

※Overhauling

Power off prior to overhauling or replacement of components

Check defects when short circuit or overload occurs, and restart it after troubleshooting

Do not connect power frequently and at least 1 minute interval after power-off for re-connection of power.

※Miscellaneous

Do not open the cabinet without permission

Disconnect the power when long term stand-by

Avoid dust and iron powder from getting into controller

If non-solid-state relay is used as output relay, freewheel diode should be connected in parallel on the relay coil.

Check whether connected power satisfies the requirement, in order to avoid burning the controller

The life span of controller is associated with environmental temperature; install cooling fan in the over-heat processing field.

The working temperature range is between 0℃~60℃

Avoid using it in the environments with high temperature, humidity, dust, or corrosive gas.

Provide rubber rails for buffering in the place with strong vibration.

※Maintenance

The following items can be conducted for daily and regular inspection, under the general usage conditions (environmental conditions: daily average temperature: 30℃, load-carry duty: 80%, and operational rate: 12 hours per day)

Daily inspection	Daily	Confirm environmental temperature, humidity, dust and foreign matter; Check whether there are abnormal vibration and sounds Check whether vents are blocked by yarns
Regular inspection	One year	Check whether solid components are loose Check whether terminal blocks are damaged

Contents

1. Overview

1.1 Specification

Pulse equivalent: (electronic gear ratio: 1:1) 0.001MM

Linkage/control axis: CNC9640 4-axis, CNC9650 6-axis, CNC9960 6-axis, CNC9810 6-axis, CNC9810E 6-axis/supporting two channels.

Program capacity: The electronic disk capacity is 4GB, and is divided into 2 zones: 2G each for Disk D and C.

RAM: 512M

Display: CNC9640, CNC9650 7” LCD 800 * 480 pixels; CNC9810, CNC9810E 8” LCD 800 * 600 pixels CNC9960 10.4” LCD 800*600 pixels

1.1.1 Basic functions

Name	Specification
Data input method	(1) NC keyboard input
	(2) U disk import
	(3) Network and serial port download and upload
Edit	(1) New program
	(2) Teach program
	(3) Save file
	(4) Programmed search
	(5)Search, search row, copy row, paste row, delete row, copy segment, delete segment
	(7) Replace
File management	(1) Browse
	(2)Copy
	(3)Paste
	(4)Cut
	(5)Delete
Authority management	(1) Superuser
	(2) Operator
	(3) Guest
System data management	(1) Parameter backup
	(2) Parameter recovery

Name	Specification
	(3) Factory reset of parameters
Auxiliary function control	(1)M_FUNC.NC M code control macro program
	(2)T_FUNC.NC T code control macro program
User configurable items	(1) Axis (number of axis, characteristic linear rotation, return-to-zero sequence)
	(2) IO port configuration
	(3) Variable name customize "SYSTABLE.csv" import
Language	(1) Simplified Chinese
	(2) Traditional Chinese
	(3) English: users can translate it into other languages through the "ZIDIAN.ZD" file.
Diagnosis	(1) Input point status
	(2) Output point status, and manual control
	(3)Alarm information
	(4) Auxiliary channel operation information
	(5)System information

1.1.2 Auxiliary functions

Name	Specification
Common functions	(1) M03,M04,M05
	(2) M08and M09 coolant switches
	(3)M10and M11 chuck control
	(4)M06 tool change command
Special auxiliary function	(1)GRUN4, GRUN5, GRUN6, GUR7
	(2)M2000, M2201 ……
	(3)M10003 ……Axis dynamic switching and synchronization
	(4) Input point triggered segments M2001 IN1 M2002 IN2

1.1.3 Spindle functions

Name	Specification
M03	(1) Spindle forward

Name	Specification
M04	(2) Spindle reversal
M05	(3)Spindle stop
S	(1) Spindle rotating speed

1.1.4 Tool functions

Name	Specification
M06	(1)M06 Txx, two-digit tool number

1.2 G codes list

G code	Group	Function
*G00	01	Positioning (rapid traverse)
G01		Linear interpolation (cutting feeding)
G02		Arc interpolation CW (clockwise)
G03		Arc interpolation CCW(counterclockwise)
G04	00	Pause, accurate stop
*G17	02	XY plane selection
G18		ZX plane selection
G19		YZ plane selection
G20	06	Imperial data entry
*G21		Metric data entry
G27	00	Return to and check reference point
G28		Return to reference point
G29		Return from reference point
*G40	07	Tool radius compensation cancel
G41		Left tool radius compensation
G42		Right tool radius compensation
G43	08	Positive tool length offset
G44		Negative tool length offset
*G49		Tool length offset cancel
G52	00	Local coordinate system setting
G53		Select machine tool coordinate system
*G54	05	Workpiece coordinate system 1
G55		Workpiece coordinate system 2

G code	Group	Function
G56		Workpiece coordinate system 3
G57		Workpiece coordinate system 4
G58		Workpiece coordinate system 5
G59		Workpiece coordinate system 6
G591		Extended workpiece coordinate system 7
G592		Extended workpiece coordinate system 8
G593		Extended workpiece coordinate system 9
G594		Extended workpiece coordinate system 10
G595		Extended workpiece coordinate system 11
G596		Extended workpiece coordinate system 12
G597		Extended workpiece coordinate system 13
G598		Extended workpiece coordinate system 14
G599		Extended workpiece coordinate system 15
G65	00	Macro program command
G73	09	Deep hole drilling fixed cycle
G74		Reverse threading fixed cycle
G76		Boring fixed cycle
*G80		Cancel fixed cycle
G81		Drilling fixed cycle
G82		Drilling fixed cycle
G83		Deep hole drilling fixed cycle
G84		Taping fixed cycle
G85		Boring fixed cycle
G86		Boring fixed cycle
G87		Reverse boring fixed cycle
G88		Boring fixed cycle
G89		Boring fixed cycle
*G90	03	Absolute value programming
G91		Increment value programming
G92	01	Programmable workpiece coordinate system setting
*G98	10	Return to initial plane in fixed cycle
G99		Return to point R plane in fixed cycle

Notice:

The items marked with * are the default modal values of G codes of the system;

1.3 Program structure

1.3.1 Program composition

CNC processing program consists of the following parts:

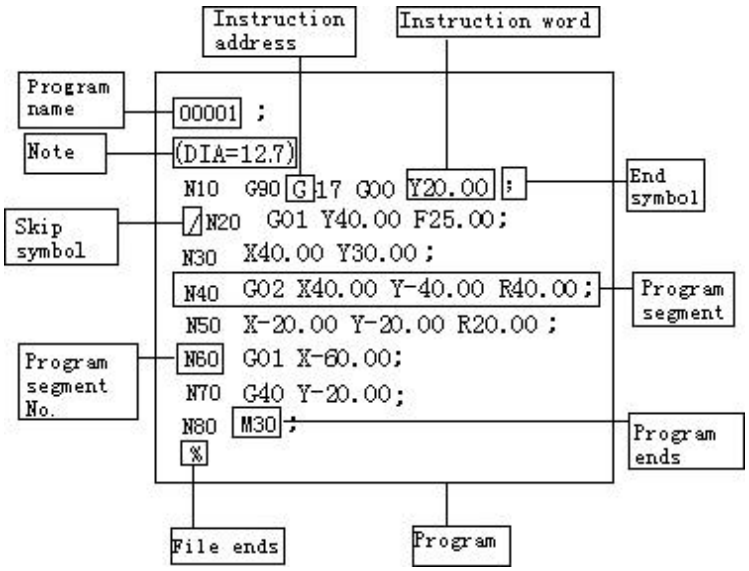


Fig. 1.3.1 CNC Program Structure Diagram

Program name:

Used to mark different programs, and consists of O and four digits.

- If the start of the program doesn't have program name, the program segment No. of the program start will be considered as the program name by default;
- If the program segment No. contains five digits, the latter four digits will be used as the program name;
- If the latter four digits are 0, add 1 automatically to use as the program name;
- N0 can't be used as program name;
- When saving the program, if both program name and program segment No. don't exist, it is necessary to make a program name through MDI panel.

Note:

The content in the parentheses, in which the user can specify notes, guide, etc.:

- The note doesn't have limit on length; if the program has a long note, the axis motion will pause for a while; therefore, if a long note is required, please put it at the place that motion pauses or without motion;
- If there is only one ")" without "(", ")" will be ignored;
- The note may have multiple lines and are separated with space;
- During processing, the note can't be executed.

Instruction address:

One English letter in the text of the processing program ("Address" hereinafter)

Instruction word:

Adding a number after the instruction address will constitute an instruction word.

Program segment No.:

Consist of letter N and number (≤5 digits), and can be randomly arranged.

- The sequence of executing program segments only related to the storage position rather than program segment No.;
- If program segment N20 appears before program segment N10, N20 shall be executed first.

Program segment:

A program segment consists of one or several instruction word and ends with ";";

N_ G_ X_Z_ F_ S_ T_ M_ ;
Program segment No. Preparation Size definition Feeding speed Spindle rotation Tool change Auxiliary function

Skip symbol:

If the first character of a program segment is "/", this program segment is conditional, i.e. skip switch. In upper position, this program segment isn't executed; when the skip switch is in lower position, this program segment is executed.

Program end:

Generally, the following codes are used when program ends:

Code	Action
M30	End main program
M99	End subroutine

✍ **Note:**

After M30 is executed, CNC stops executing and returns to program start;

After M99 is executed, CNC returns to the program that calls this subroutine and continues executing.

File end:

If the program end doesn't have %, CNC is reset.

Instruction word is the basic unit of program segment. Every address has unique meaning, and the following values also have different formats and ranges, as in the Table below:

Table 4.1 Instruction Address and Range of Command Value

Function	Address	Range	Meaning
Program name	O	1~9999	Program No.
Program segment No.	N	1~9999	Sequence No.
Preparation function	G	00~99	Specify motion mode (linear, arc…)
Size definition	X, Y, Z	±99999.999 mm	Coordinate position value
	R	±99999.999 mm	Arc radius, corner radius
	I, J, K	±9999.9999 mm	Arc center coordinate position value
Feeding rate	F	1~100,000 mm/min	Feeding rate
Spindle rotation	S	1~4000 rpm	Spindle rotation
Select tool	T	0~99	Tool No.
Auxiliary function	M	0~99	Auxiliary function M code No.
Tool offset No.	H, D	1~200	Specify tool offset No.
Pause time	P, X	0~65 sec	Pause time (ms)
Specify subroutine No.	P	1~9999	To call subroutine
Repeat times	P, L	1~999	To call subroutine
Parameter	P, Q, R	P is 0~99999.999 Q is ±99999.999 mm R is ±99999.999	Fixed cycle parameters

1.3.2 Main program and subroutine

The processing programs include main programs and subroutines. Generally, NC executes the instructions of main program; however, NC will turn to execute subroutine when executes a subroutine calling instruction, and will return to the main program when executes the return instruction in subroutine.

When the processing program needs to run same track for several times, edit this track into the subroutine and save in the program memory of the machine tool, and this subroutine can be called when this track should be executed in the program.

When the main program calls a subroutine, this subroutine can call another subroutine, which is called double nesting. Generally, the machine tool allows up to quadruple subroutine nesting. In calling subroutine instruction, the subroutine can be repeated for 999 times.

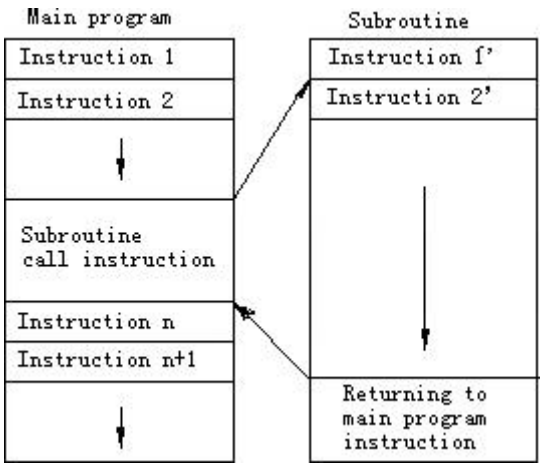


Fig. 1.3.2 Main Program and Subroutine

Subroutine format:

OXXXX	; Subroutine name
……	;
……	; Subroutine content
……	;
M99	; Subroutine ends, and returns to previous program

✍ **Example: X100.0 Y100.0 M99;**

☞ **Note:**

Program start should have a subroutine name specified by address O

M99 doesn't need to appear in a program segment separately.

Subroutine call format:

M98P XXX XXXX

☞ **Note:**

In the number following address P, the latter four digits are used to specify the program No. of called subroutine, and the former three digits are used to specify the repeat times of calling.

✍ **Example:**

M98 P41005; call subroutine 1005, repeat four times

G90 G00 X-75. Y50. Z53. M98 P40035; this program segment specifies the X, Y, Z axis to fast locate the instruction position, and then call subroutine 0035 for four times.

☞ **Note:**

- If the calling time isn't specified, the subroutine will be called only once;
- M98 doesn't need to appear in a program segment separately;
- Different from other M codes, M98 and M99 won't send signal to the machine tool when executing;
- NC gives an alarm if can't find the program No. specified by address P;
- Subroutine call instruction M98 can't be executed in MDI mode; to execute a subroutine separately, please edit the following program in the editing mode, and execute in automatic running mode.

O×××; M98 P××××; M30;

1.3.3 Modal and non-modal function

G code determines the function of the command and can be classified into two types:

Non-modal G code:

G code is only valid in defined program segment

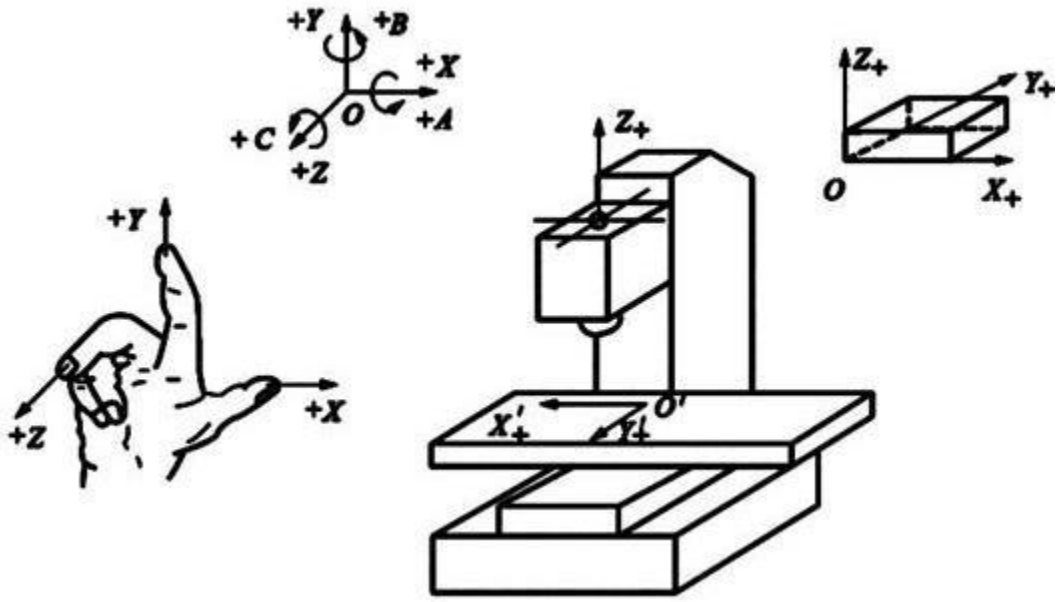
Modal G code:

G code is always valid, until next G code of same group appears.

✍ **Example: G01 and G00 are modal G codes**

G01X_;
Y_;
Z_;
} G01 is valid in this range
G00X_;

1.4 Motion direction naming of control axes



This system can control the rapid traverse, feeding and interpolation of four axes. The axis direction is defined in Cartesian coordinate system, as shown below (facing to the machine tool):

Z axis: The up and down movement of the tool relative to the workpiece is Z axis motion, with the upward movement the positive motion and the downward movement the negative motion.

X axis: The left and right movement of the tool relative to the workpiece is X axis motion, with the rightward movement the positive motion and the leftward movement the negative motion.

Y axis: The forward and backward movement of the tool relative to the workpiece is Y axis motion, with the forward movement the positive motion and the backward movement the negative motion.

Spindle:

Look down to the workpiece, the clockwise rotation is spindle positive rotation and the counterclockwise rotation is negative rotation.

A, B, C axes:

The positive directions of rotation axes correspond to the positive directions of X, Y, Z axis, which are determined according to the forward direction of right hand screw.

Notice:

The X, Y, Z, A, B, C axis motion described in this manual is the tool's motion relative to the workpiece, i.e. it is assumed that the workpiece coordinate system has been set.

Coordinate systems of machine tool and workpiece

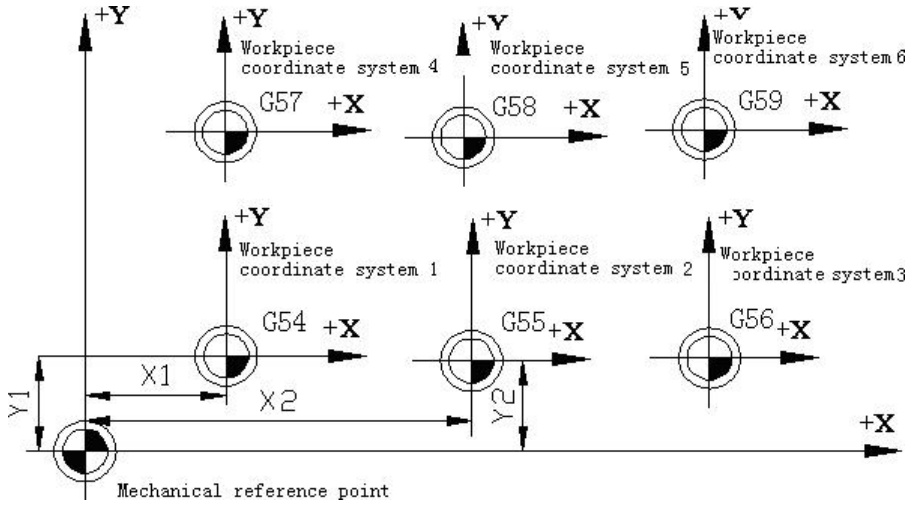
Machine tool coordinate system:

The coordinate system fixed on the machine tool is created through returning to reference point after NC is electrified every time. To select machine tool coordinate system, use G53 instruction.

Workpiece coordinate system:

When start programming, the programmer doesn't know the position of the workpiece on the machine tool, and usually uses a point on the workpiece as the reference point to write processing program. The coordinate system created with this reference point is the workpiece coordinate system. When the workpiece is fixed on the worktable of the machine tool, move the tool to specified workpiece reference point and set the coordinate value of this point as the origin of workpiece coordinate system, and the tool will use this workpiece coordinate system as the reference system and process according to program instruction when the system executes the machining program. Therefore, the origin offset function of coordinate system is very important to CNC machine tool.

This system can preset six workpiece coordinate systems (nine extended coordinate systems G591-G599 are added in new version). Set the offset of every workpiece coordinate system origin relative to machine tool coordinate system origin, and then use G5X (5X is the specific workpiece coordinate system number, the same below) instruction to select. G5X are nodal instructions, corresponding to 1#~6# preset workpiece coordinate system respectively.



Workpiece Coordinate System Diagram

2. System programming

2.1 Preparation functions (G function)

2.1.1. G90 G91 absolute and relative programming

Function:

Tool motion instructions include absolute value instruction and increment value instruction. In absolute value instruction mode, the coordinate value of the motion end in current coordinate system is specified; in increment value instruction, the distance of every coordinate axis relative to the start point motion is specified.

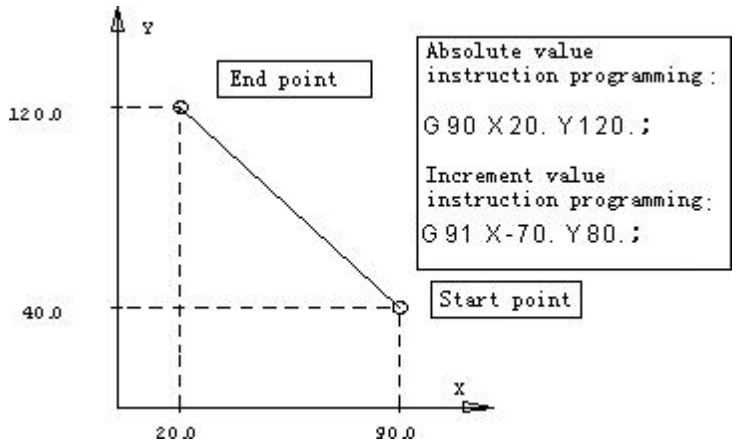
Format:

G90 X_ Y_ Z_ α_;
G91 X_ Y_ Z_ α_;
G90…… absolute value instruction
G91…… increment value instruction
α…… additional axis

Details:

In absolute value instruction mode, the tool motion is unrelated to current position, and moves according to the position of specified workpiece coordinate system;

In increment value instruction, the current position is the start point;



Graphic Description Text

For the instructions from workpiece coordinate system home, absolute value or increment value coordinate instructions are same;

G90 and G91 are modal instructions, and are always valid until next new setting of G90 and G91.

2.1.2. Rapid positioning (G00)

Function:

Every axis moves to specified position at specified rapid traverse speed respectively; in absolute coordinate system, the specified motion end is the coordinate value in current coordinate system; in increment coordinate system, the motion distance of every coordinate axis relative to start point is specified.

Format:

G00 X_ Y_ Z_α_; (α is additional axis) X Y Z α is coordinate value; absolute or increment programming mode is determined according to G90 or G91 state specified by the program.

Details:

This instruction changes other G functions; G00 is always valid until the G01, G02 and G03 instructions of same group (01) appears; when G00 mode is valid, the latter instructions only need to specify coordinate X, Y, Z.

In G00 mode, the tool always accelerates at the start point and decelerates at the end point of every path. It will execute next path only after the in-place state is confirmed.

When every motion axis reaches the end point, CNC considers that this program segment has ended and turns to next program segment.

When G00 instruction is valid, the G code function of group 09 (G73-G89) turns into cancellation state (G80).

The motions among different axes are disrelated, i.e. tool path is straight line or broken line (confirmed by selected parameters), but the positioning time doesn't change.

Straight line path: same as linear interpolation (G01) mode, the speed is limited by the fast feeding speed of every axis.

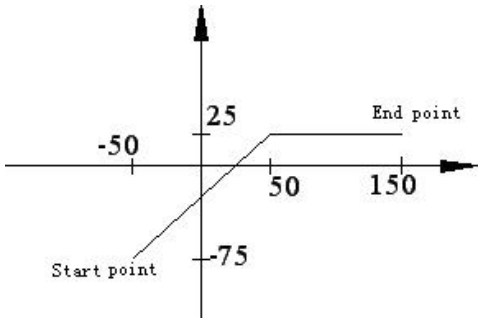
Broken line path: every axis is independent and moves for positioning at the maximum speed.

Notice:

If there is no following number, G will be treated as G00.

Example:

The position of start point is X-50, Y-75; instruction G00 X150. Y25; the tool will have the track shown in the figure below.



G00 Programming Diagram

2.1.3. Linear interpolation (G01)

Function:

G01 changes current interpolation state into linear interpolation, tool moves to specified position from current position, and the track is a straight line from start point to end point.

Format:

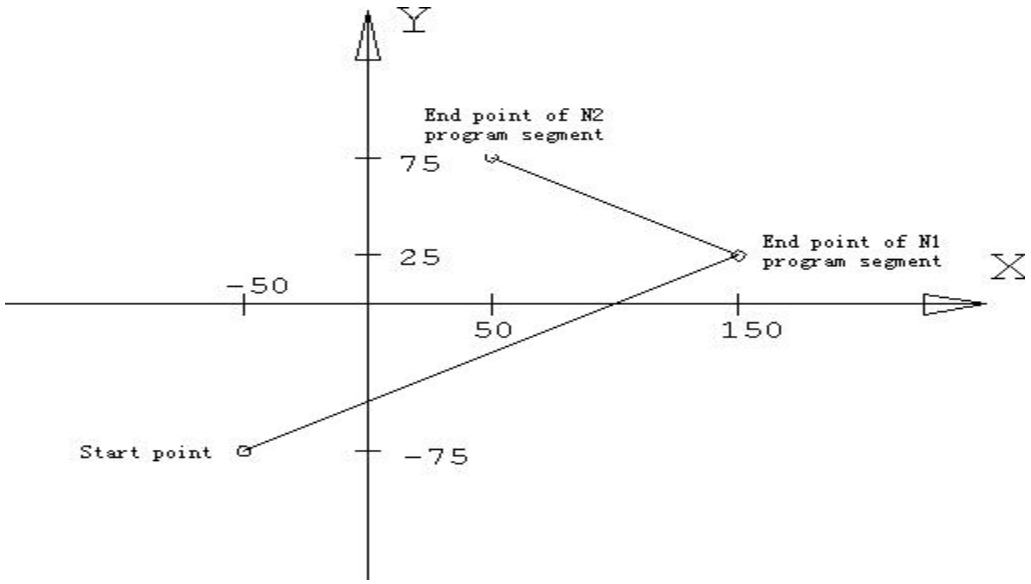
G01 X_ Y_ Z_ α_ F_; (α is additional axis) X Y Z α is coordinate value; absolute or increment programming mode is determined according to G90 or G91 state specified by the program. F indicates the speed of linear motion (unit: mm/min)

Details:

This instruction changes other G functions, and G01 is always valid until G00, G02 or G03 instruction of same group (01) appears. If the next instruction is still G01 and the feeding speed is same, G01 can be ignored. If the program segment in which G01 instruction appears for the first time doesn't have F instruction, there will be error.

Example:

The feeding speed of rotation axis is expressed in °/min. (F300=300°/min) Suppose that the current point of the tool is X-50. Y-75., the following program segment N1 G01 X150. Y25. F100 ; N2 X50. Y75.; will make the tool have the track shown in the figure below.



G01 Programming Diagram

2.1.4. Arc interpolation (G02, G03)

Function:

Used to move the tool in arc track

Format:

On X—Y plane
G17 { G02 / G03 } X__Y__{ (I__J__) / R__} F__;
On X--Z plane
G18 { G02 / G03 } X__Z__{ (I__K__) / R__} F__;
On Y--Z plane
G19 { G02 / G03 } Y__Z__{ (J__K__) / R__} F__;

Arc Interpolation Command Format Description

S/N	Data content		Instruction	Meaning
1	Plane selection		Plane selection	Specify the arc interpolation on X-Y plane
				Specify the arc interpolation on Z-X plane
				Specify the arc interpolation on Y-Z plane
2	Arc direction		Arc direction	Arc interpolation in clockwise direction CW
				Arc interpolation in counterclockwise direction CCW
3	End point	G90 mode	Two axes instruction in X, Y, Z	The coordinate value of the end point position in current workpiece coordinate system
		G91 mode	Two axes instruction in X, Y, Z	Distance from start point to end point (directional)
4	Distance from start point to circle center		Distance from start point to circle center	Two axes instruction in I, J, K
	Arc radius		Arc radius	Arc radius
5	Feeding rate		Feeding rate	The speed of arc motion

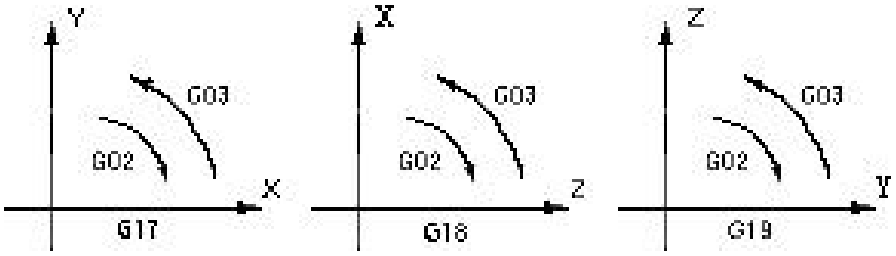
Details:

G02 (G03) is modal instruction.

The arc crossing multiple quadrants can be specified in one program segment.

Note:

Arc direction
X-Y plane: look to negative direction from Z axis
X-Z plane: look to negative direction from Y axis
Y-Z plane: look to negative direction from X axis



Arc Interpolation Plane Definition Diagram

The end point of the arc is determined by address X, Y and Z. In G90 mode, i.e. absolute value mode, address X, Y and Z specify the coordinate value of arc end in current coordinate system; in G91 mode, i.e. increment value mode, address X, Y and Z specify the distance from the point of current tool to the end point in the direction of every axis.

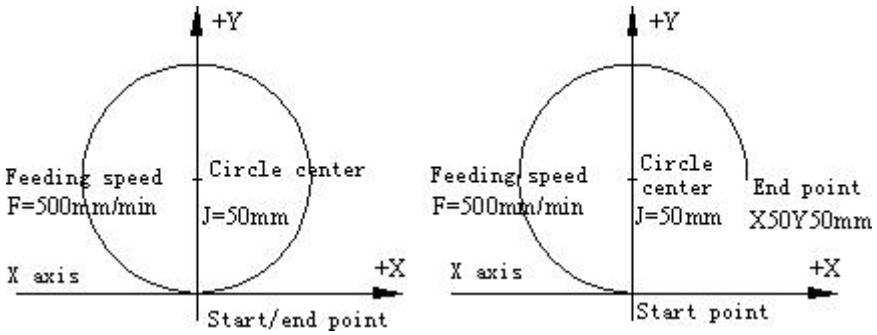
In X, Y and Z direction, the distance from the point of current point to the circle center is specified by address I, J and K respectively, the symbols of which are determined by their motion directions.

The coordinate value of arc end can be either in absolute value or increment value, while the coordinate value of arc center must be increment instruction from the start point.

When X, Y and Z are ignored (the start point coincides with the end point), I, J and K define the circle center, and the track will be a full circle.

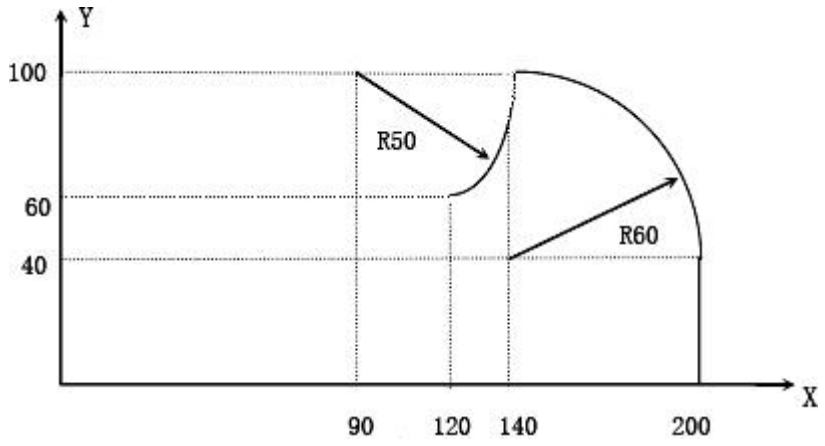
Example:

G02 J50 F500;
G91 G02 X50 Y50 J50 F500;
The processing tracks are shown in the figures below (full circle and 3/4 arc)



Instruction Diagram of Processing Full Circle

To program a segment of arc, in addition to specifying end point and circle center position, it is also possible by specifying radius and end point position. If the radius is specified with address R, the value of R can be either positive or negative; a positive R value can be used to determine an arc smaller than 180°, and a negative value can be used to determine an arc larger than 180°. Programming a full circle is only possible by specifying circle center.



Absolute/Increment Programming Diagram

Above tracks are programmed in absolute value and increment value mode as follows:

- Absolute value mode

```
G00 X200.0 Y40.0 Z0    ;
G90 G03 X140.0 Y100.0 I-60.0 F300.0    ;
G02 X120.0 Y60.0 I-50.0    ;
or:

G00 X200.0 Y40.0 Z0    ;
G90 G03 X140.0 Y100.0 R60.0 F300.0    ;
G02 X120.0 Y60.0 R50.0    ;
```

- Increment mode

```
G91 G03 X-60.0 Y60.0 I-60.0 F300.0    ;
G02 X-20.0 Y-40.0 I-50.0    ;
or:
G91 G03 X-60.0 Y60.0 R60.0 F300.0    ;
G02 X-20.0 Y-40.0 R50.0    ;
```

The feeding speed of arc interpolation is specified with F, which is the speed of tool in arc tangent direction.

2.1.5. Pause instruction (G04)

Function:

Pause for a period of time between two program segments.

Format:

G04 P_ or G04 X_
Address P specifies the pause time, and the minimum unit of its instruction is 0.001 second if there is no radix point.
Address X specifies the pause time, and the minimum unit of its instruction is 1 second if there is no radix point.

Example:

G04 P 1000	: pause for 1000ms, equal to 1sec
G04 X 1	: pause for 1sec

2.1.6. Plane selection (G17-G19)

Function:

This group of instruction is used to select the plane of arc interpolation and tool radius compensation.

Format:

G17……select XY plane
G18……select ZX plane
G19……select YZ plane
X, Y, Z indicate the coordinate axes or parallel axes

Details:

When the system is electrified, plane XY is selected by default.

In the program segment without instruction G17, G18 or G19, the plane doesn't have any change.

Example:

```
G18 X_ Z_   ;ZX plane
X_ Y_   ; plane doesn't change (ZX plane)
```

Motion instruction is disrelated to plane selection.

Example:

Under the following instruction,

```
G17 Z_   ;
Z axis doesn't exist on XY plane, and Z axis motion is disrelated to XY plane.
```

About the instructions related to plane selection, please refer to the content related to arc interpolation and tool compensation instructions.

2.1.7. Machine tool coordinate system (G53)

Machine tool coordinate system:

The coordinate system fixed on the machine tool is created through returning to reference point after NC is electrified every time. To select machine tool coordinate system, use G53 instruction.

Format (machine tool coordinate system):

```
G53   X_Y_Z_;
X_Y_Z_; The coordinate absolute value of every axis
```

Details:

When the machine tool is electrified, it must be reset in auto or manual mode, and the coordinate system is created basing on reset reference origin.

The machine tool coordinate system won't change before the power supply is cut off after created.

The machine tool coordinate system won't be changed due to G92 instruction.

G53 instruction only can be used in absolute value mode (G90).

G53 is non-modal instruction, and is only valid in current program segment.

If G53 instruction and G28 instruction appear in the same program segment at the same time, the latter instruction is valid.

When G53 instruction is created, cancel tool radius compensation and tool offset.

All G53 instructions move in quick feeding mode.

The distance between machine tool coordinate system home and machine tool reference point is determined by the parameters; unless otherwise specified, the reference point of every axis coincides with machine tool coordinate system home.

Workpiece coordinate system

Workpiece coordinate system:

When start programming, the programmer doesn't know the position of the workpiece on the machine tool, and usually uses a point on the workpiece as the reference point to write processing program. The coordinate system created with this reference point is the workpiece coordinate system. When the workpiece is fixed on the worktable of the machine tool, move the tool to specified workpiece reference point and set the coordinate value of this point as the origin of workpiece coordinate system, and the tool will use this workpiece coordinate system as the reference system and process according to program instruction when the system executes the machining program. Therefore, the origin offset function of coordinate system is very important to CNC machine tool.

2.1.8. Programmable workpiece coordinate system (G92)

Function:

This instruction creates a new workpiece coordinate system, so that the coordinate value of the point where current tool locate is the value of IP_ instruction in this workpiece coordinate system. (As shown in Fig. 8.1)

Format:

(G90) G92 X_Y_Z_;
X_Y_Z_; The coordinate absolute value of every axis

Details:

G92 instruction is a non-modal instruction, but the workpiece coordinate system created with this instruction is modal.

Actually, this instruction also specifies an offset, which is specified indirectly. It is the coordinate value of new workpiece coordinate system origin in original workpiece coordinate system; seen from G92 function, this offset is the difference between the coordinate value of the tool in original workpiece coordinate system and IP_ instruction value. (As shown in Fig. 8.1)

If G92 instruction is used for several times, the offset specified by G92 instruction will superpose. For every preset workpiece coordinate system (G54-G59), the superposed offset is valid.

New coordinate system of the part is set in above instruction, e.g. the coordinate value of tool tip is IP_. Once the coordinates are confirmed, the position of the absolute value instruction is the coordinates in this coordinate system.

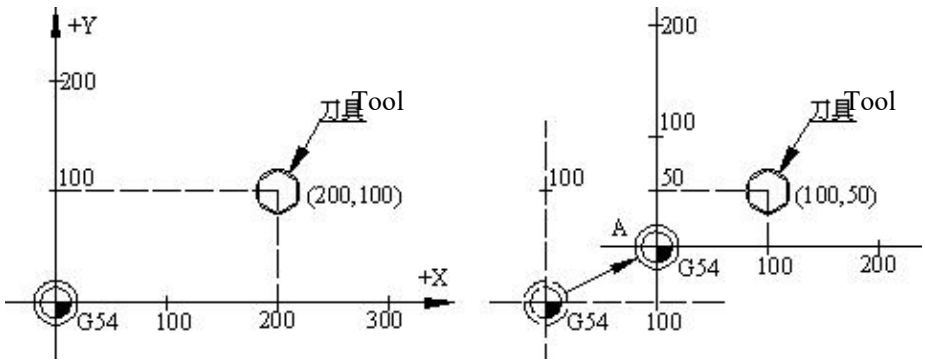
Example:

The coordinates of the tool in original coordinate system are (200, 100), after executing (G92 X100 Y50):

The origin of new coordinate system offsets to the position A in the lower right figure;

The offset of coordinate system is (100, 50), (the difference between the coordinates of the tool in original coordinate system and IP_ instruction value).

The coordinates of the tool in new coordinate system are (100, 50).



G92 Instruction Function Diagram

2.2 Gfunction related to reference point

The machine tool coordinate system is created through returning to reference point after NC is electrified every time. The reference point is a fixed point on the machine tool, and its position is determined by the installation position of stopper switch of every axis and the home position of the servo motor of every axis. When this machine tool returns to the reference point, the coordinates of the reference point in the machine tool coordinate system is X0, Y0, Z0.

2.2.1. Auto return to reference point (G28)

Function:

This instruction makes the axis return to reference point of the machine tool through the center point specified by IP at the feeding speed of quick positioning.

Format:

G28 X_ Y_ Z_α_; (α is additional axis)

X Y Z α indicate the coordinates of center point.

Details:

The center point may be specified either in absolute value mode or increment value mode, which depends on current mode.

Generally, this instruction is used to move the workpiece out of the processing area when the entire processing program ends, so as to unload processed parts and load the parts to be processed.

When execute G28 instruction before returning to reference point manually, the motion of every started from center point is same as returning to reference point manually, and the motion direction started from the center point is positive.

The coordinates in G28 instruction is saved as center point by NC; on another hand, if an axis isn't contained in G28 instruction, the coordinates of the center pointed saved by NC will use the value G28 instruction specified previously.

✍ **Example:**

```
N0010  X20.0   Y54.0;
N0020  G28   X-40.0   Y-25.0;        coordinates of center point (-40.0,-25.0)
N0030  G28   Z31.0;                  coordinates of center point (-40.0,-25.0,31.0
```

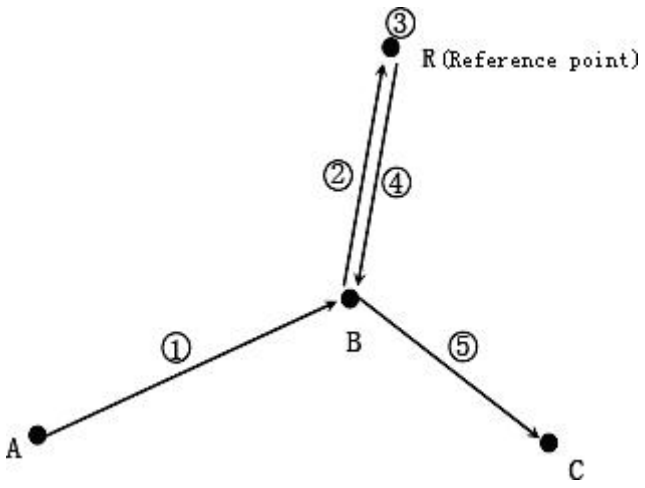


Diagram of Automatically Returning to Reference Point

💣 **Note:**

The coordinates of this center point are mainly used by G29 instruction.

In tool offset mode, tool offset is also valid for G27; for safety reasons, tool offset should be disabled before executing G28 instruction (radius offset and length offset).

2.2.2. Auto return from reference point (G29)

Function:

This instruction makes the axis move from reference point to instruction position through center point at the feeding speed of quick positioning; the position of center point is confirmed by previous G28 instruction.

Format:

G29 X_ Y_ Z_α_; (α is additional axis)
X YZ α indicate the coordinates of end point of the tool motion.

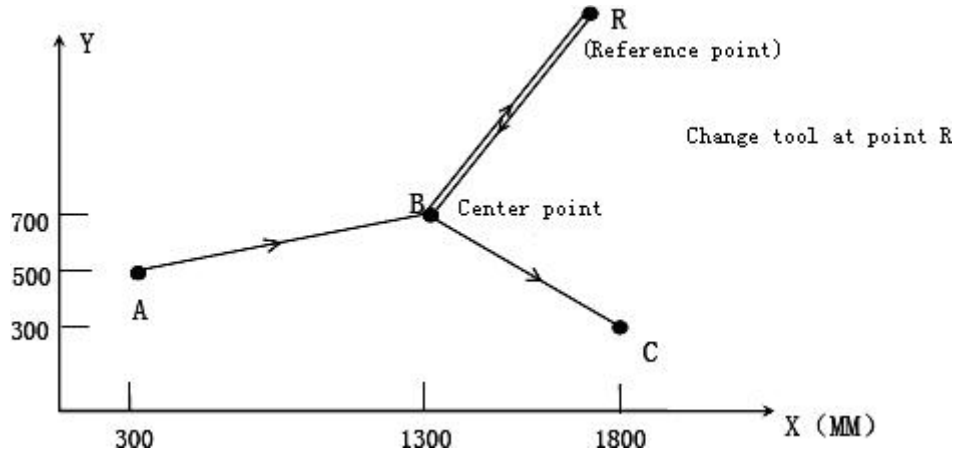
Details:

Generally, after this instruction is used for G28, the instructed axis is on reference point or second reference point.

In increment value mode, the instruction value is the distance from center point to end point (instruction position).

In program, the specific movement amount from center point to reference point doesn't need to be calculated.

G28, G29 example:



G28, G29 Usage Diagram

G28 X1300.0 Y700.0;	(A→B program)
……	
G29 X1800.0 Y300.0;	(B→C program)

Note:

When change part coordinate system after moving to reference point through center point with G28 instruction, the center point also moves to new coordinate system; when instruct G29 later, positioning at instructed position through center point in new coordinate system.

2.2.3. Reference point return checking (G27)

Function:

This instruction makes the axis move to the position of IP instruction at the feeding speed of quick positioning, and then checks whether this point is reference point; if yes, sends the finishing signal that this axis returns to reference point (reference point arriving indicator of this axis is lighted); if not, gives an alarm and interrupts the running program.

Format:

G27 X_ Y_ Z_ P_;
X YZ indicates that reference point returns to control axis.
P reference point returns number (the first reference point by default)

Details:

The axes of simultaneous reference point return check are same to simultaneously controlled axes.

If the reference point isn't reached after instruction is executed, the program alarms.

Coordinate system setting function (G52-G59, G591-G599, G92)

Using preset workpiece coordinate system (G54~G59, G591~G599)

According to the loading position of the workpiece in the machine tool, this system can preset six coordinate systems (nine extended in new version); through the operation on LCD panel, set the offset of the origin of every workpiece coordinate system relative to the origin of machine tool coordinate system, and then use G54~G59, G591~G599 to select, which are modal instructions, corresponding to 1#~15# preset workpiece coordinate systems respectively.

Example:

Preset 1# workpiece coordinate system offset: X-150.000, Y-210.000, Z-90.000

Preset 4# workpiece coordinate system offset: X-430.000, Y-330.000, Z-120.000

Program segment content	Coordinates of end point in machine tool coordinate system	Note
N1 G90 G54 G00 X50. Y50.;	X-100, Y-160	Select 1# coordinate system, quick positioning

N2 Z-70.;	Z-160	
N3 G01 Z-72.5 F100;	Z-160.5	Linear interpolation, F value is 100
N4 X37.4;	X-112.6	(Linear interpolation)
N5 G00 Z0;	Z-90	Quick positioning
N6 X0 Y0 A0;	X-150, Y-210	
N7 G53 X0 Y0 Z0;	X0, Y0, Z0	Select to use machine tool coordinate system
N8 G57 X50. Y50. ;	X-380, Y-280	Select 4# coordinate system
N9 Z-70.;	Z-190	
N10 G01 Z-72.5;	Z-192.5	Linear interpolation, F value is 100 (modal value)
N11 X37.4;	X392.6	
N12 G00 Z0;	Z-120	
N13 G00 X0 Y0 ;	X-430, Y-330	

Seen from above samples, the function of G54~G59 instruction is to move the coordinate origin used by NC to the point that the coordinates in machine tool coordinate system are preset value; please refer to the operation section in this manual for the method of presetting.

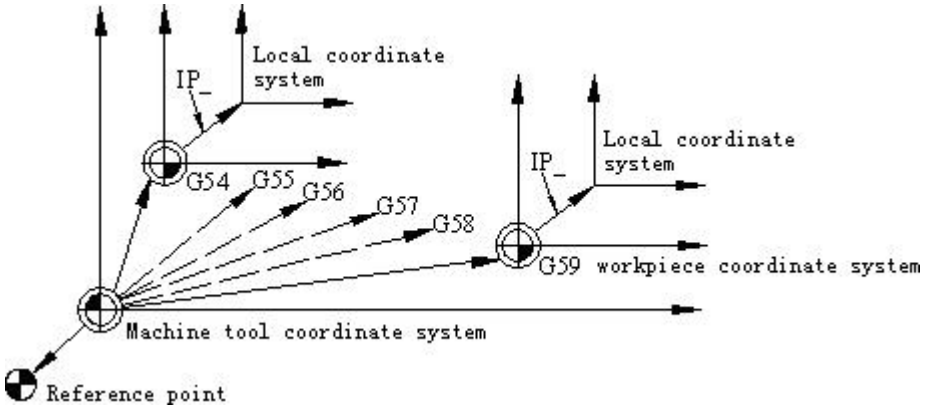
After returning to the home of machine tool, coordinate systems 1~6 of the workpiece are created. G54 is the initial mode after electrified. The absolute position of the position screen is the coordinates in current coordinate system.

In CNC programming of machine tool, unless otherwise specified, the IP of interpolation instruction and other instructions related to coordinates are the coordinate position in current coordinate system (the coordinate system used when the instruction is executed). In most cases, the current coordinate system is one of G54~G59, and machine tool coordinate system are seldom used directly.

2.2.4. Local coordinate system (G52)

Function:

G52 can create a local coordinate system, which is a sub-coordinate system equivalent to G54~G59.



Local Coordinate System Diagram

Format:

G52 X_Y_Z_; X_Y_Z_; Equivalent to the offset of current G54~G59 coordinate systems,

Details:

In this instruction, IP_specifies the offset equivalent to current G54~G59 coordinate systems, i.e. IP_specifies the position coordinates of local coordinate system origin in current G54~G59 coordinate system.

G52 instruction is always valid after specified until next G52 instruction is specified.

G52 instruction can set the processing coordinate system without changing the workpiece coordinate system.

G52 IP0 (G52 X0 Y0 Z 0 α0) can be used to cancel local coordinate system.

The setting of local coordinate system doesn't change the machine tool coordinate and workpiece coordinate system.

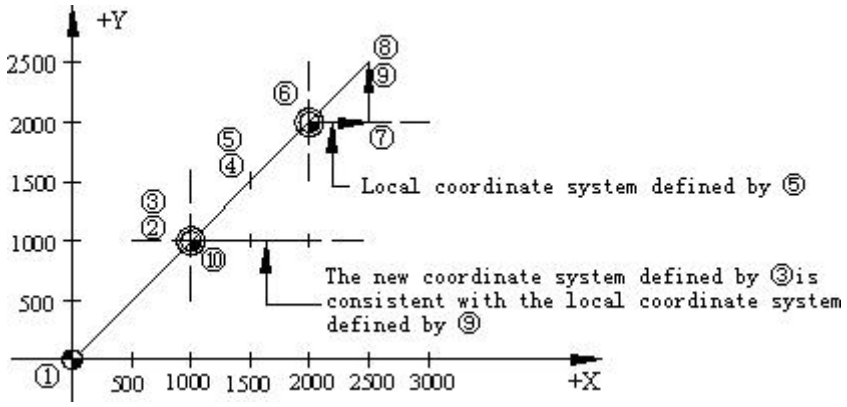
G52 instruction can replace G92 instruction to specify the offset between the origin of processing program and workpiece origin.

Example:

Local coordinate system in absolute value mode

①G28 X0 Y0; ②G00 G90 X1000 Y1000; ③G92 X0 Y0; define workpiece coordinate system ④G00 X500 Y500;

⑤G52 X1000 Y1000;	define local coordinate system
⑥G00 X0 Y0;	
⑦G01 X500 F100;	
⑧Y500;	
⑨G52 X0 Y0;	cancel local coordinate system
⑩G00 X0 Y0;	



Local Coordinate System Usage Diagram in Absolute Value Mode

2.3 Tool compensation G function

CNC programming is considered as the motion track of a point; however, the tool has certain length or radius, and therefore the motion track of tool point during part contour machining isn't the actual contour of the part; they have the difference of a tool length or radius; to make the motion track of tool point coincide with the actual contour, it must offset a distance, which is called tool compensation.

Tool compensation consists of length compensation and radius compensation. The tool length is different or wears due to long time cutting, and thus the length compensation is required. Radius compensation is required because the actual processing tool always has certain tool radius or tip arc radius, and therefore there is a difference of tool radius between tool point motion track and the actual contour of the part during part contour processing. To make the motion track of tool point coincide with the actual contour, it is necessary to offset a tool radius, which is tool radius compensation.

2.3.1. Tool length compensation (G43, G44, G49)

Function:

Assume the difference between tool length and actual tool length when correct the programming.

Format:

G43 Z_ H_;	positive offset
G44 Z_ H_;	negative offset
G49 Z_; (or H00) tool length compensation cancel	

Move the end point position of Z axis instruction for an offset according to above instruction, and preset the difference between tool length and the tool length of actual processing assumed during programming in offset memory, and therefore the operator only needs to change the tool compensation to process parts with tools of different lengths without changing the program.

Details:

In either absolute value or increment value mode, for G43, add the offset specified by H code (set in offset memory) to Z axis motion instruction end point coordinates in the program; for G44, subtract the offset specified by H code, and use the calculated coordinates as the end point coordinates.

When Z axis motion is omitted, if the offset is positive, G43 instruction will move an offset in positive direction and G44 will move an offset in negative direction. If the offset is negative, it moves to reverse direction

G43 and G44 are modal G codes, which are always valid before the G codes of same group appear.

Specifying offset:

H code specifies the offset No., the corresponding offset will add or subtract Z axis motion instruction when the program is running, and thus creates new motion instruction of Z axis. Offset No. can be specified between H00 and H18, while the offset corresponding to H00 can't be set to static 0.

Enter tool compensation menu, and preset the offset to corresponding offset No. in the offset memory.

mm inch		
Offset	0-±999.999	0-±99.9999

Cancel tool length compensation:

Cancel tool length compensation with G49 or H00.

Example:

Tool compensation processing (hole #1, #2, #3)

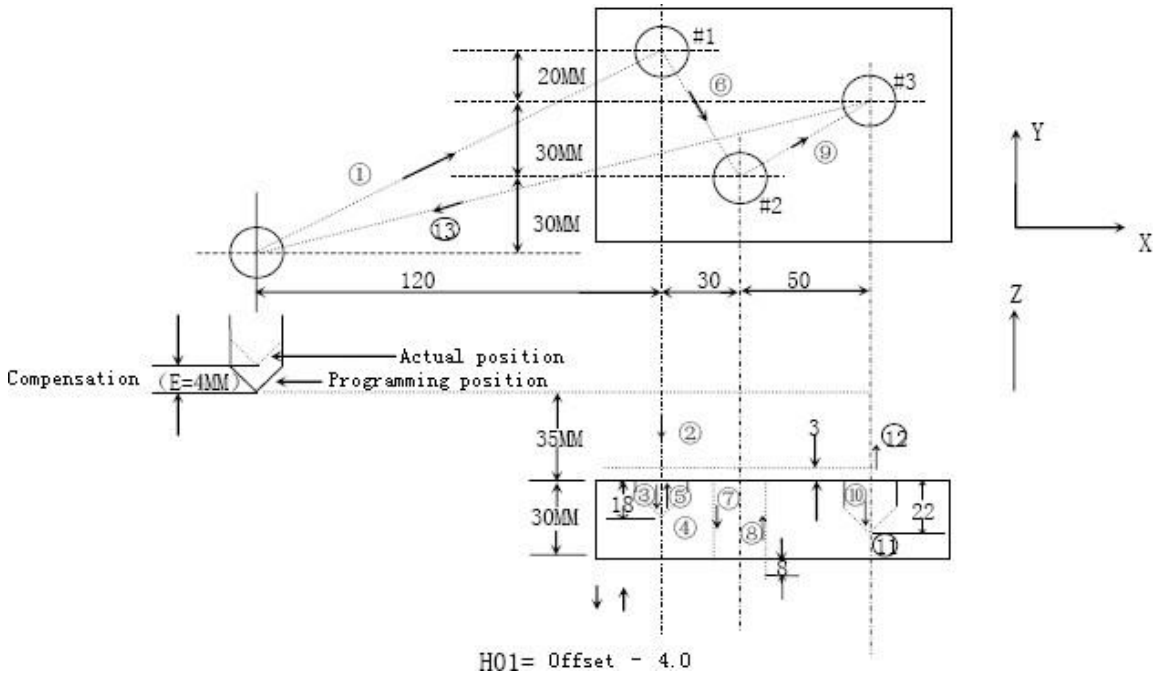


Fig. 9.1 Tool Compensation Processing Hole Example

```
N1 G91 G00 X120.0 Y80.0;.....(1)
N2 G43 Z-32.0 H01;.....(2)
N3 G01 Z-21.0;.....(3)
N4 G04 P2000;.....(4)
N5 G00 Z21.0;.....(5)
N6 X30.0 Y-50.0;.....(6)
N7 G01 Z-41.0;.....(7)
N8 G00 Z41.0;.....(8)
N9 X50.0 Y30.0;.....(9)
N10 G01 Z-25.0;.....(10)
N11 G04 P2000;.....(11)
N12 G00 Z57.0 H00;.....(12)
N13 X-200.0 Y-60.0;.....(13)
N14 M30;
```

Note:

When the offset No. is changed, it only changes to new offset, rather than adding the new offset to the old offset.

H01.....offset 20.0
H02.....offset 30.0
G90 G43 Z100 0 H01….....Z moves to 120.0
G90 G43 Z100 0 H02….....Z moves to 130.0

2.3.2. Tool radius compensation (G40, G41, G42)

Tools radius compensation function:

Tool radius compensation is expressed with G instruction (G40-G42) and D instruction, and the radius of selected tool can be compensated in any vector direction.

Format:

Cancel or carry through tool radius compensation vector with G40, G41 and G42 instruction. They combine with G00, G01, G02 and G03 instructions, define a mode and confirm the value of compensation vector, direction and tool motion direction.

G code	Function
G40 X_ Y_ ;	Tool radius compensation cancel
G41 X_ Y_ ;	Tool radius left compensation
G42 X_ Y_ ;	Tool radius right compensation

Details:

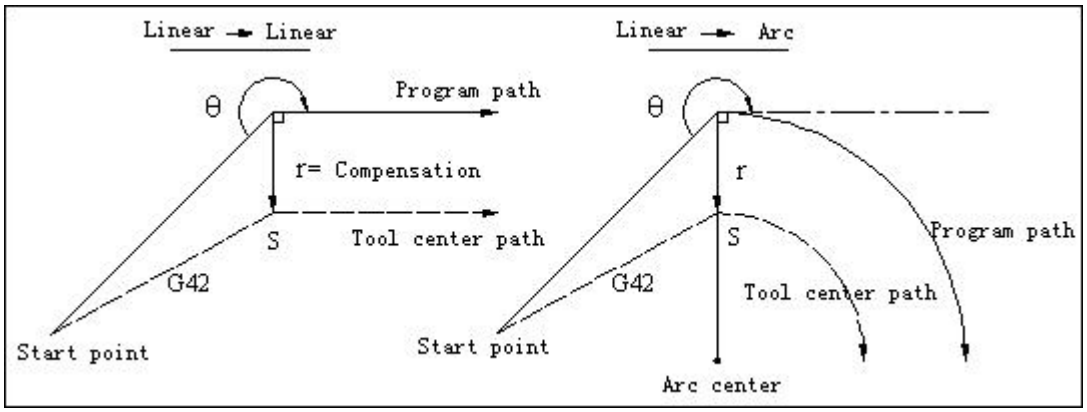
Tool radius compensation is specified by D instruction, and H instruction is invalid.

The plane selection of tool radius compensation can be compensated according to D instruction or in the plane specified by two axes; the axis instructions out of selected plane won't be compensated; for the usage of G instruction plane selection, please refer to the instructions of plane selection.

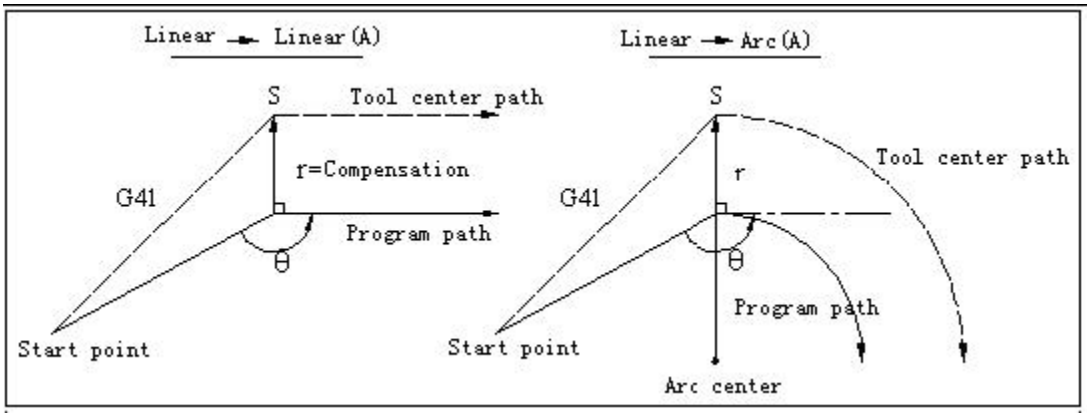
Tool radius compensation action

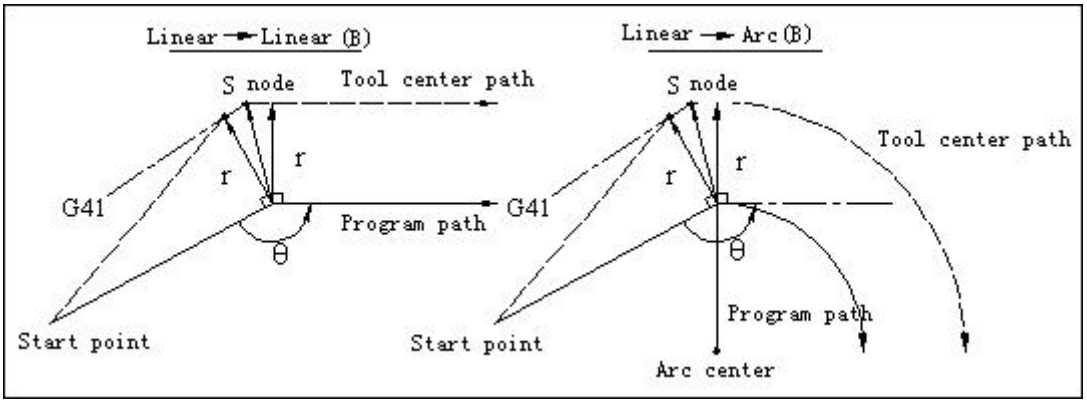
Start action of tool radius compensation

(1) Casions inside of the corner

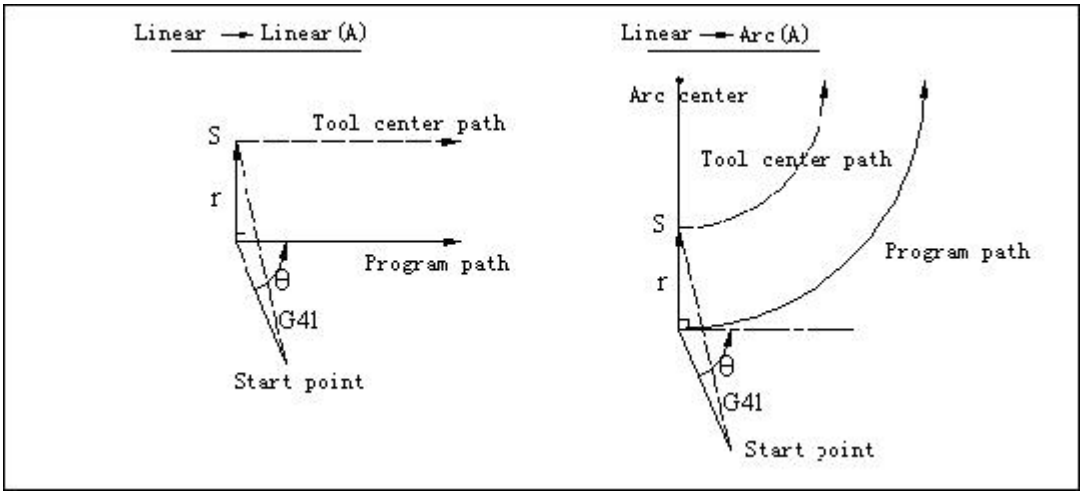


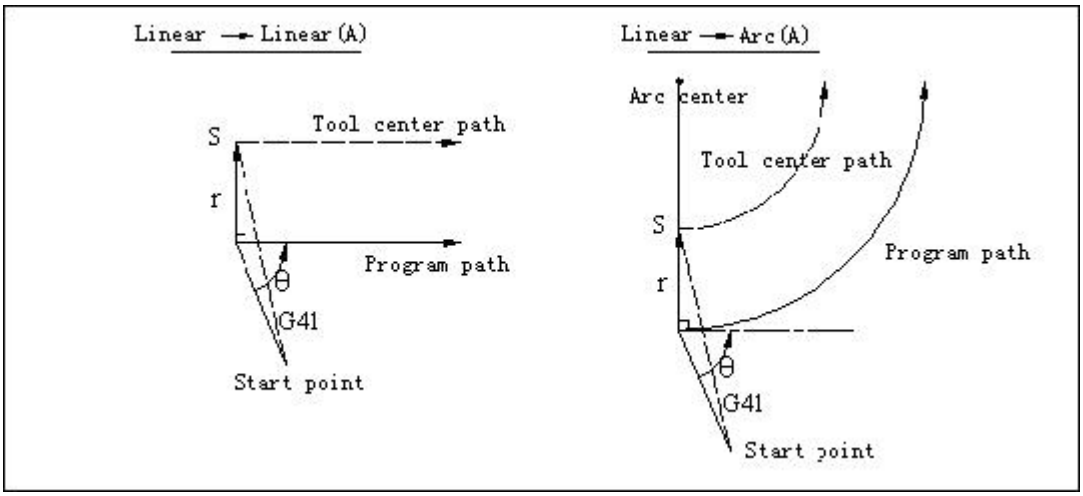
(2) Oasions out of the corner (obtuse angle) [$90^{o} \leq \theta < 180^{o}$]





(3) Occasions out of the corner (acute angle) [$\theta < 90^{o}$]



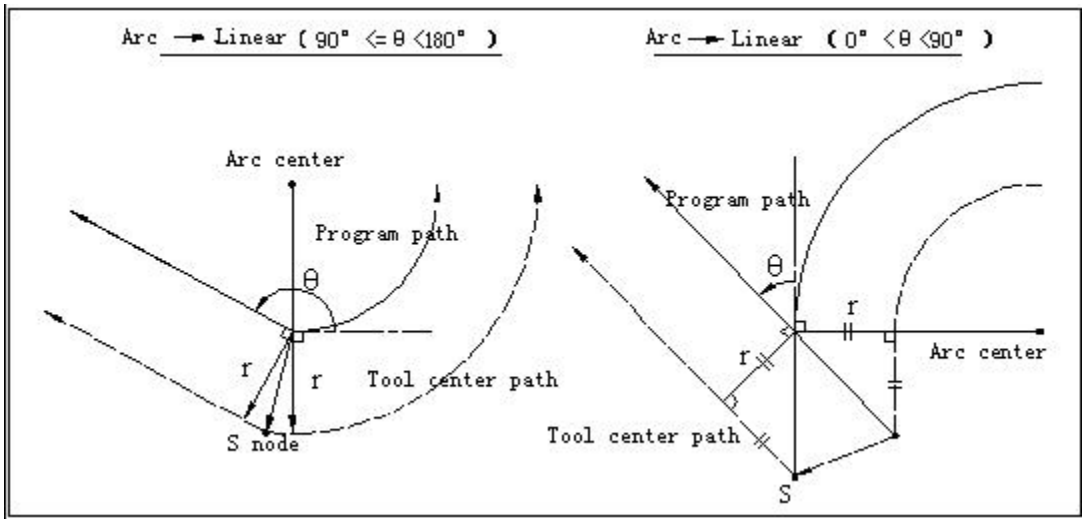


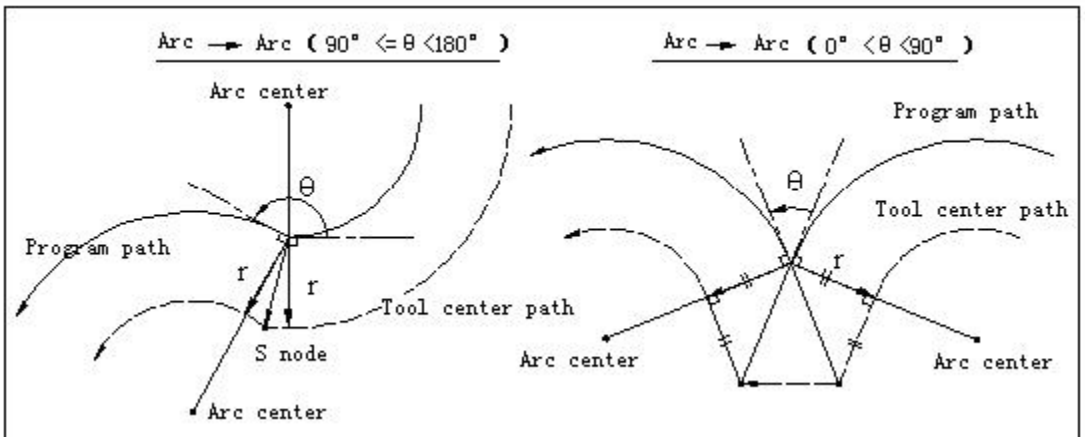
Note: In the program segment that compensation starts, there shouldn't be arc instruction G02, G03, else it will alarm (P/S69).

Action in compensation mode

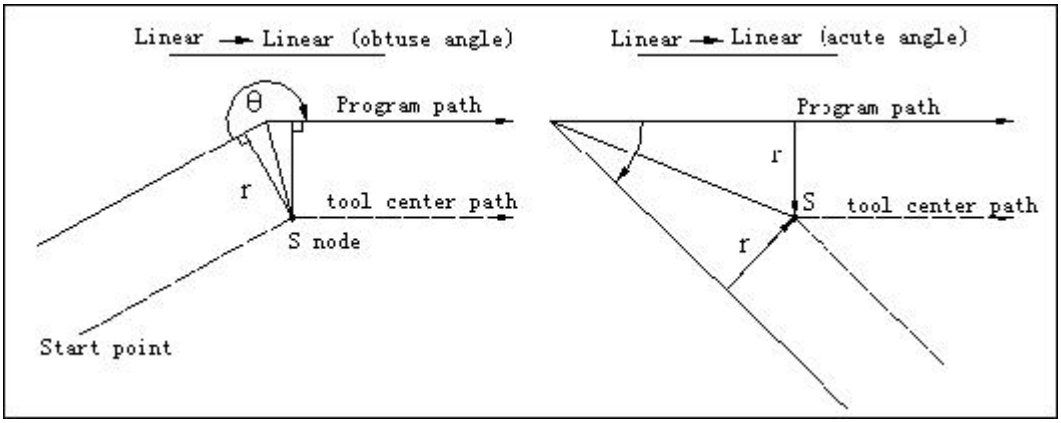
In compensation mode, the same compensation instructions (G41/G42) do not require new setting; over cutting or insufficient may occur if four or more continuous segments do not have motion instructions.

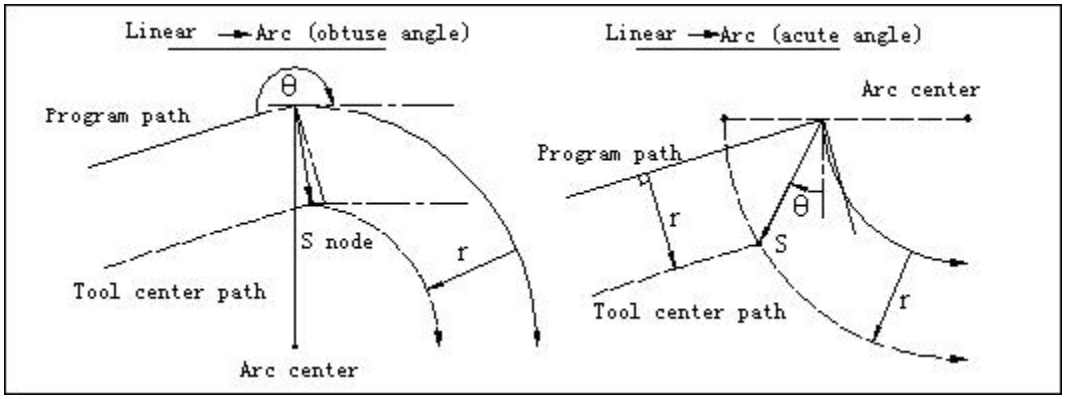
(1) Occasions that outer corner rotates

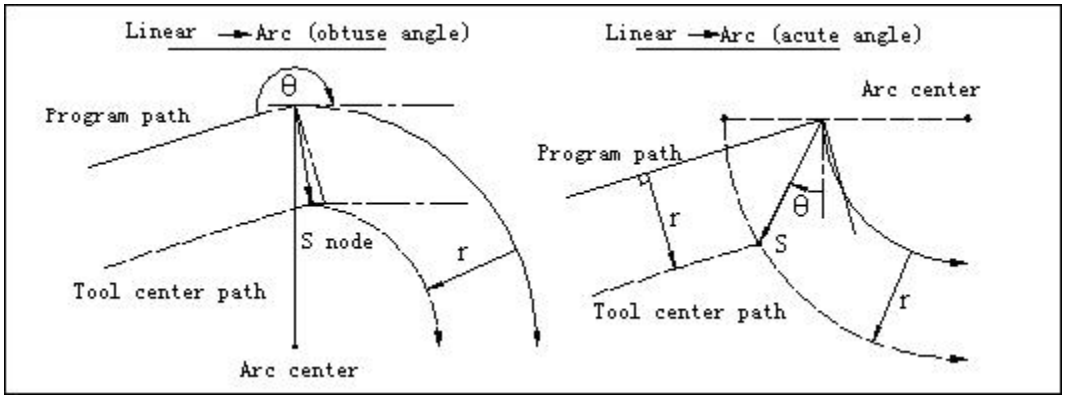


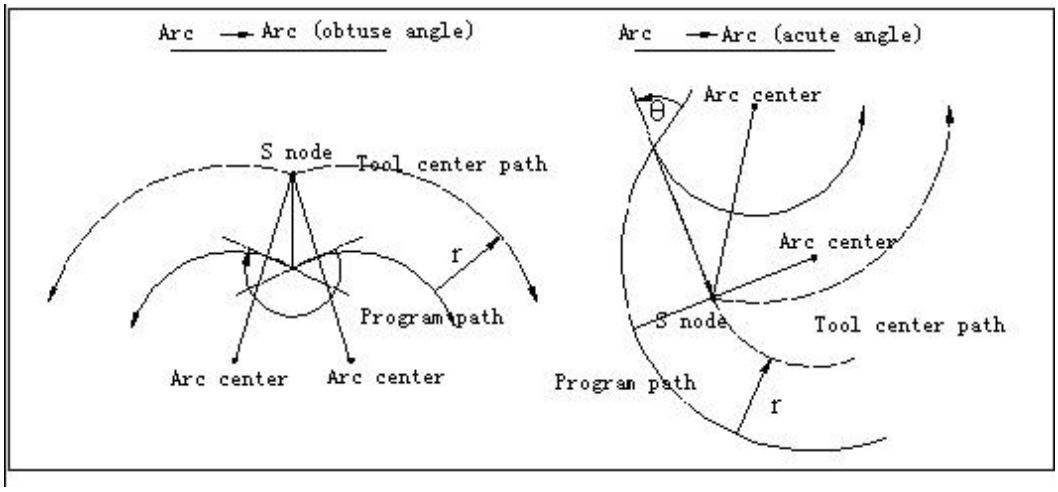


(2) Occasions that inner corner rotates



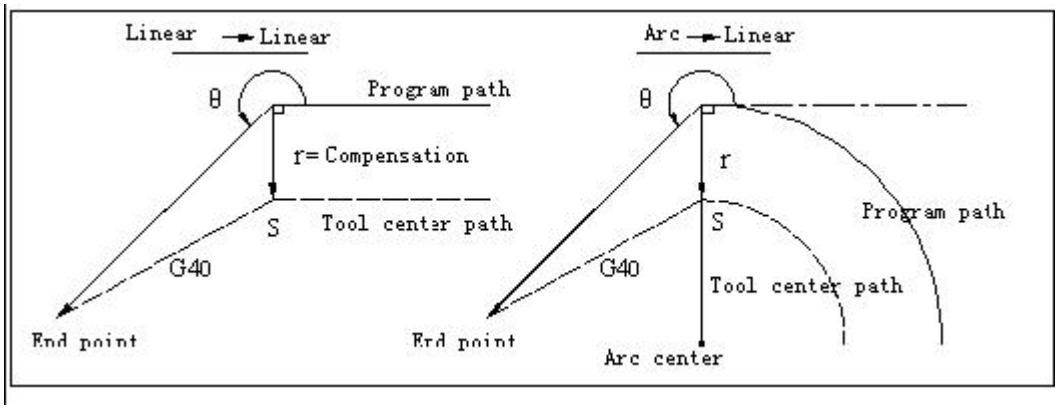

Linear → Arc (obtuse angle)
Linear → Arc (acute angle)
θ
Program path
S node
r
Tool center path
Arc center
Arc center
Program path
r
θ
S
Tool center path
r


Linear → Arc (obtuse angle)
Linear → Arc (acute angle)
θ
Program path
S node
r
Tool center path
Arc center
Arc center
Program path
r
θ
S
Tool center path
r

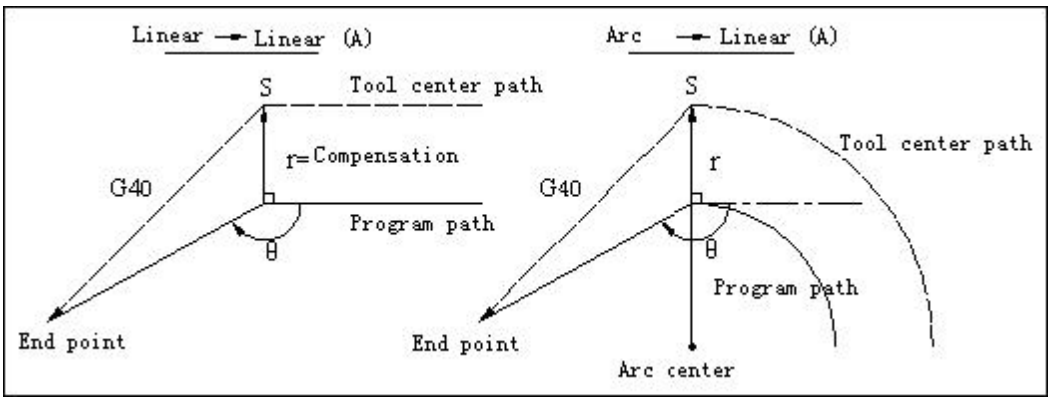

Arc → Arc (obtuse angle)
Arc → Arc (acute angle)
S node
Tool center path
r
Program path
Arc center
Arc center
θ
Arc center
Arc center
S node
Tool center path
Program path
r

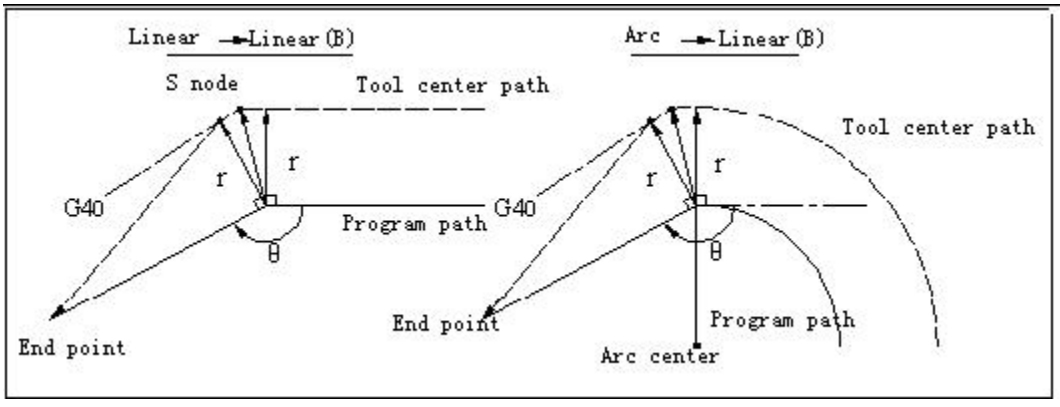
Cancelling tool radius compensation

(1) Occasions inside the corner

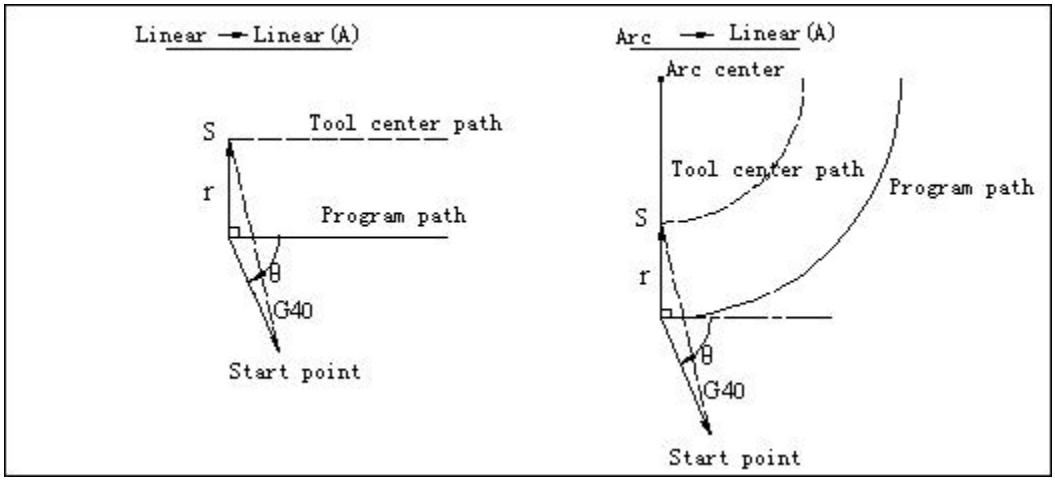


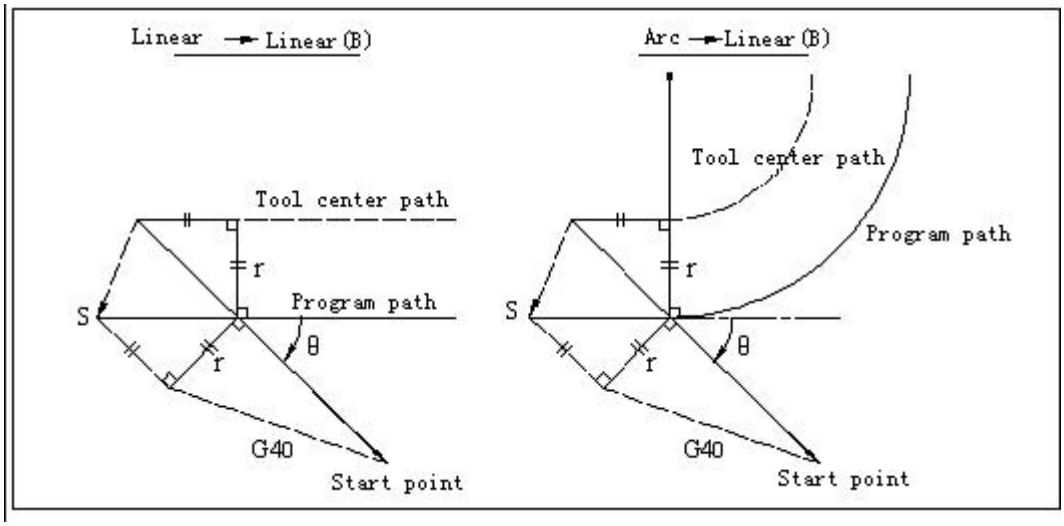
(2) Occasions out of corner (obtuse angle)





(3) Occasions out of corner (acute angle)



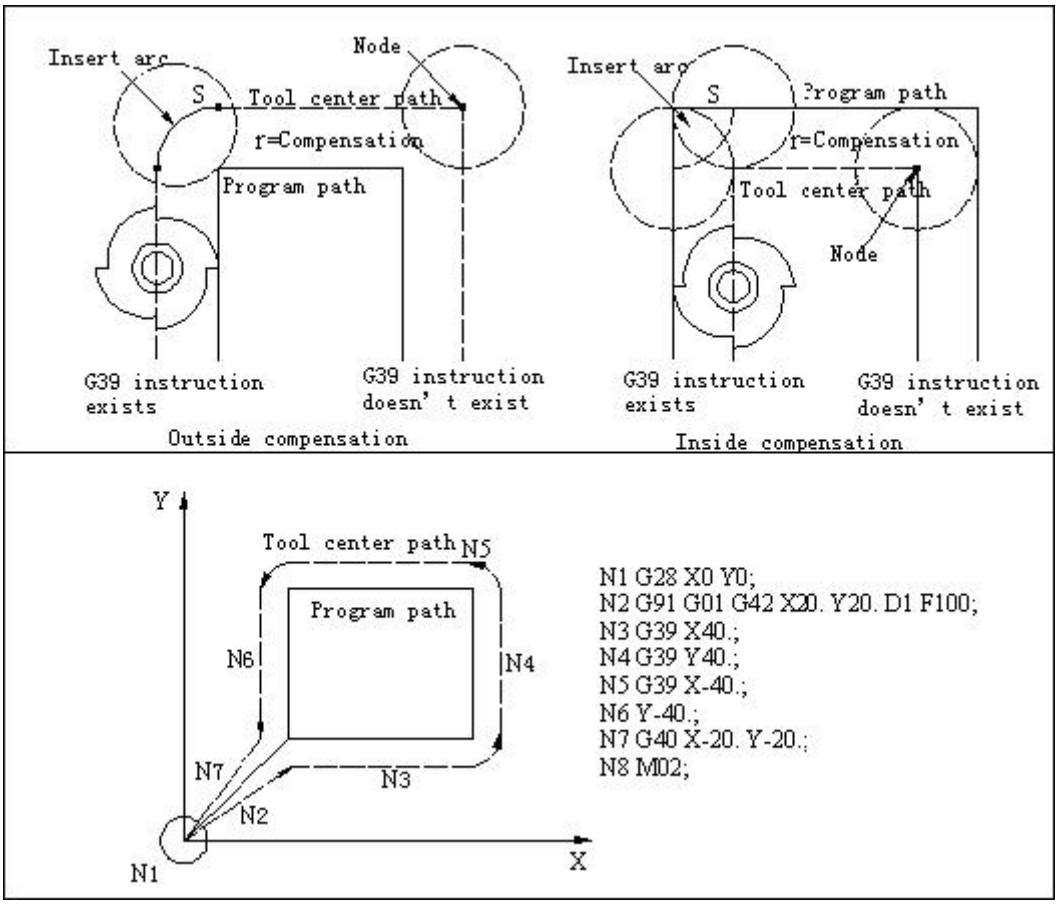


Note: In the program segment that cancelling compensation starts, there shouldn't be arc instruction G02, G03, or else it will alarm (P/S70).

Other instructions and actions during tool radius compensation

Inserting corner arc

When G39 (corner arc) instruction is specified, the node at the workpiece corner calculates compensation and inserts automatically.

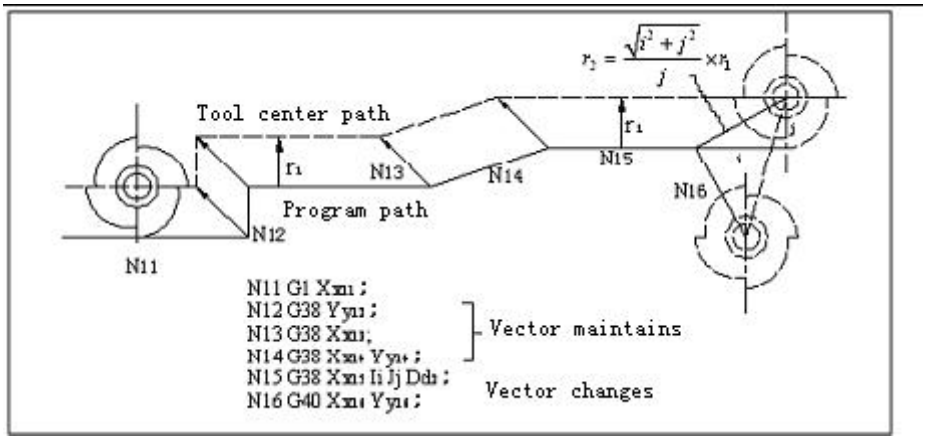


Corner vector changes/maintains

According to G38 instruction, the compensation vector in tool radius compensation can be changed or maintained.

(1) Maintain vector: when G38 instruction is moving single segment instruction, the end point of this single segment isn't calculated as the node, and maintains the vector same to migration segment.

(2) Change vector: the new compensation vector direction is specified by I, J and K, and the compensation is specified by D.

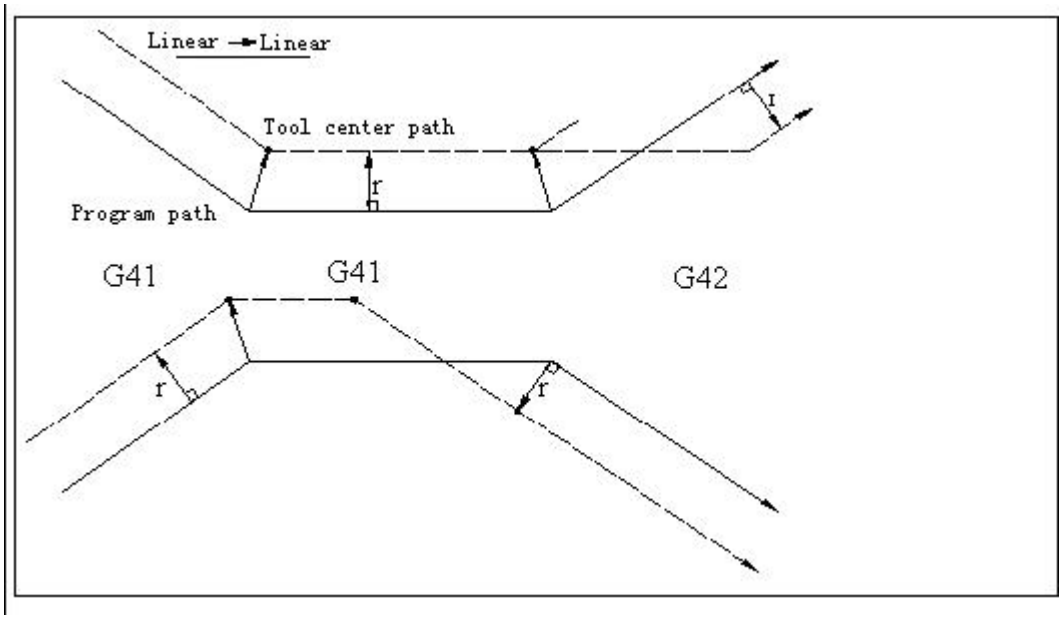


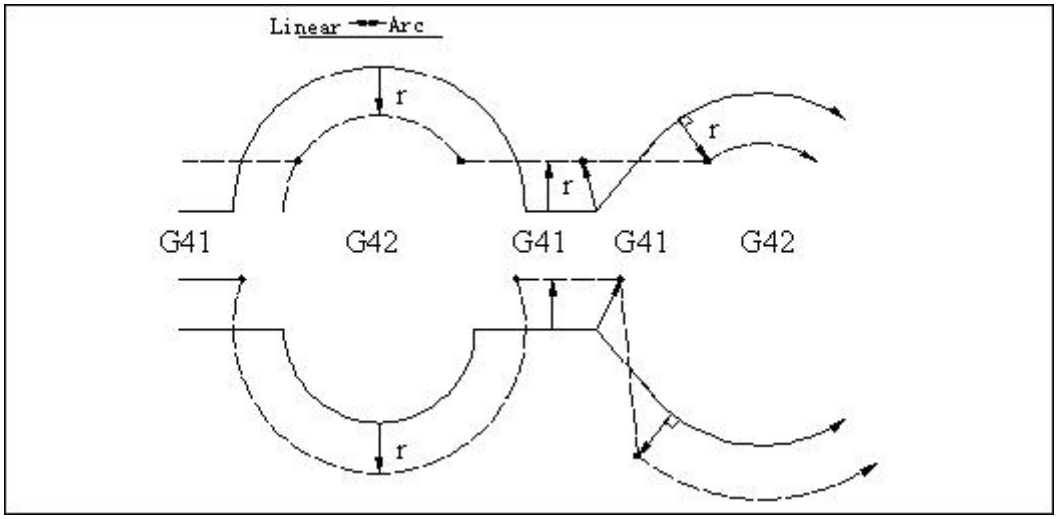
Changing compensation direction in tool radius compensation

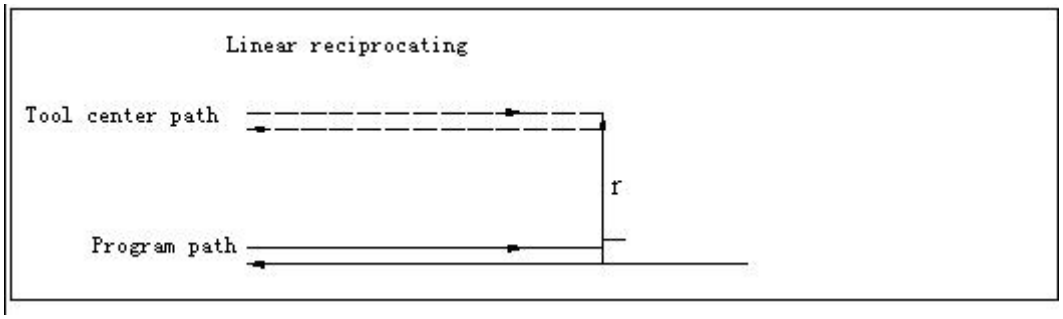
The compensation direction follows the tool radius compensation instruction (G41, G42) and compensation symbol.

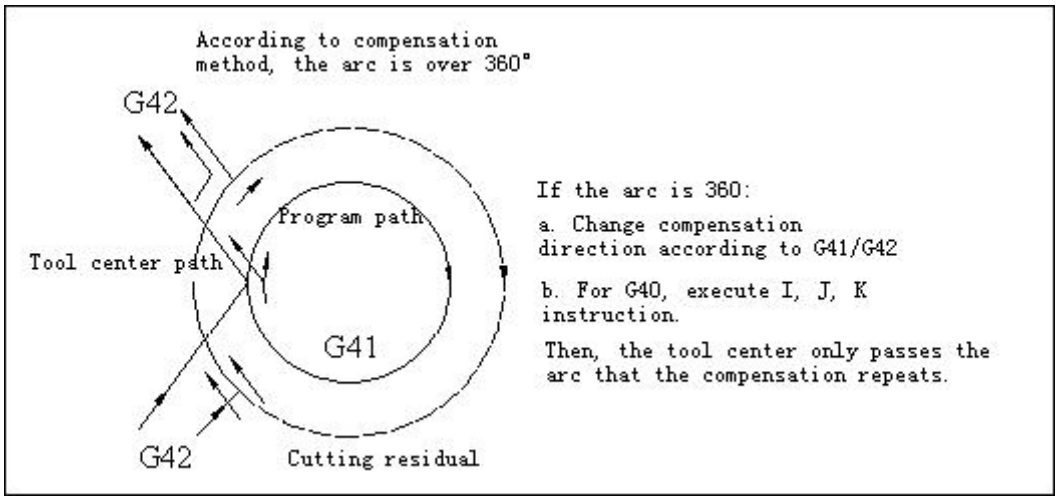
In compensation mode, the compensation instruction and direction can be changed without compensating cancellation instruction. However, the compensation start segment and next segment can't be changed.

When compensation direction is changed, and there is no intersection





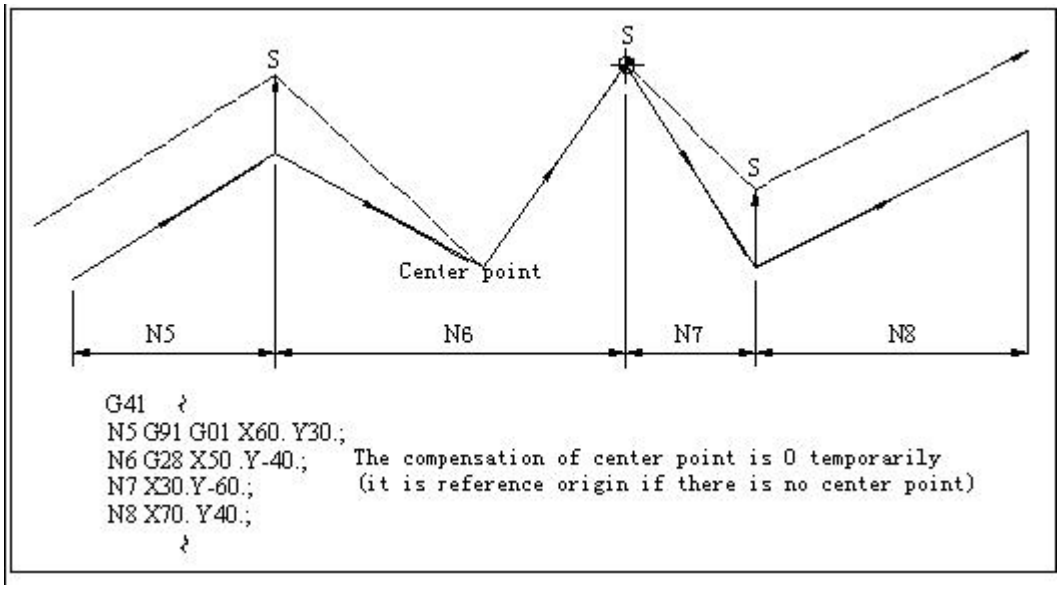




Instruction of canceling compensation vector temporarily

If the following instructions are used in compensation mode, the compensation vector will be invalid temporarily. Later, the compensation mode will resume automatically. In this case, the compensation cancellation action is invalid, the tool moves from intersection to the instruction point of compensation vector directly, i.e. moving to program instruction point; when compensation mode resumes, the tool moves to the intersection directly.

(1) Instruction of returning to reference point



(2) If G53 instruction is used, basic mechanical coordinate system selection will become temporary compensation vector.

When the coordinate system sets (G92) instruction, the compensation vector doesn't change.

Details

In the following segments, the tool doesn't have motion

M03;….....M instruction

S12;….....S instruction

T45;….....T instruction

G04X500;….....Pause

G22X200 Y150 Z100;….....Restricted processing area setting

G10 L10 P01 R50;….....Compensation setting

G92 X600 Y400 Z500;….....Coordinate system setting

(G17)Z40;….....Compensation the motion out of the plane

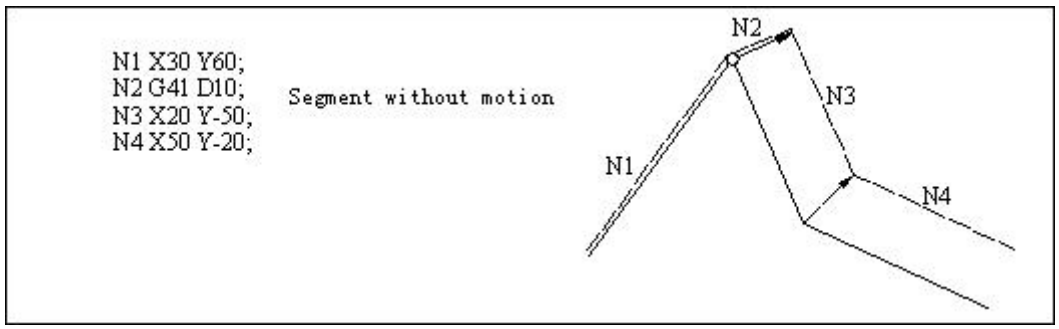
G90;….....G instruction only

G91 X0;….....0 is moved

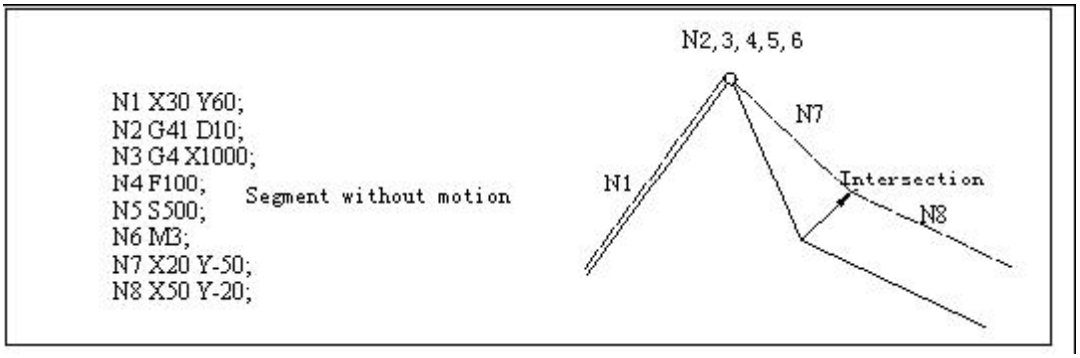
M00, M01, M02, M03 stop M instruction

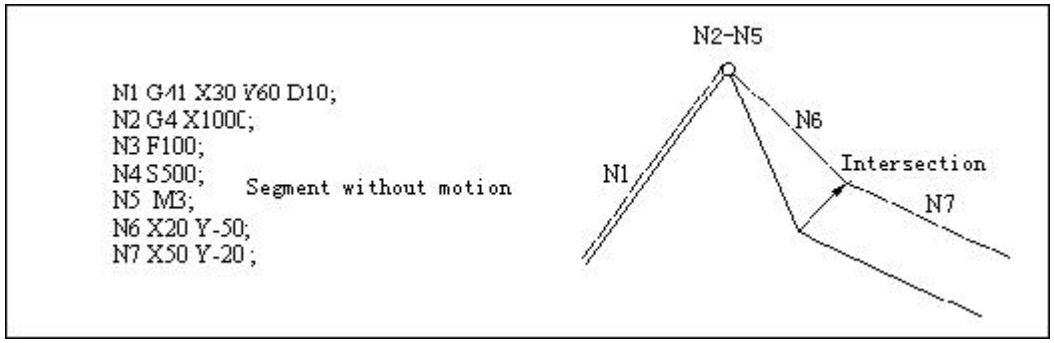
(1) Instructions when compensation starts

Then, move the segment to compensate in vertical direction.



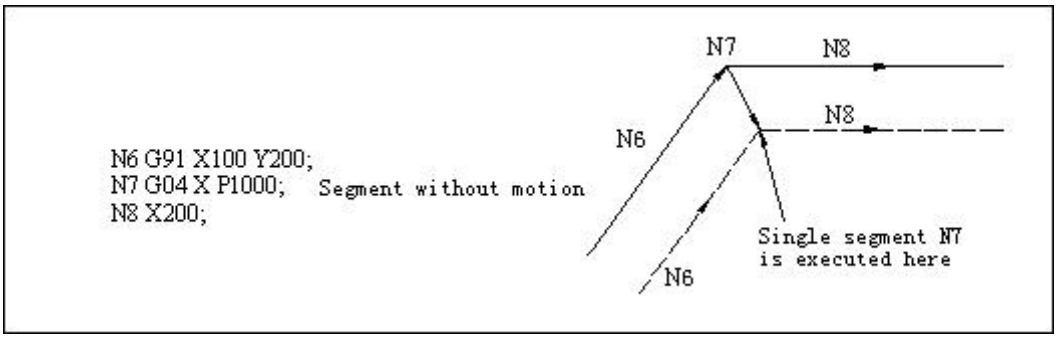
If four segments without motion are specified consecutively, the compensation vector can't be accomplished.



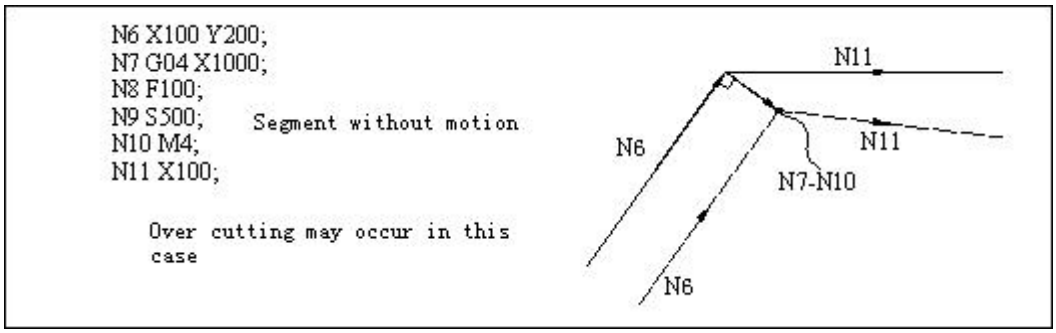


(2) In compensation mode, the occasions specified by instruction

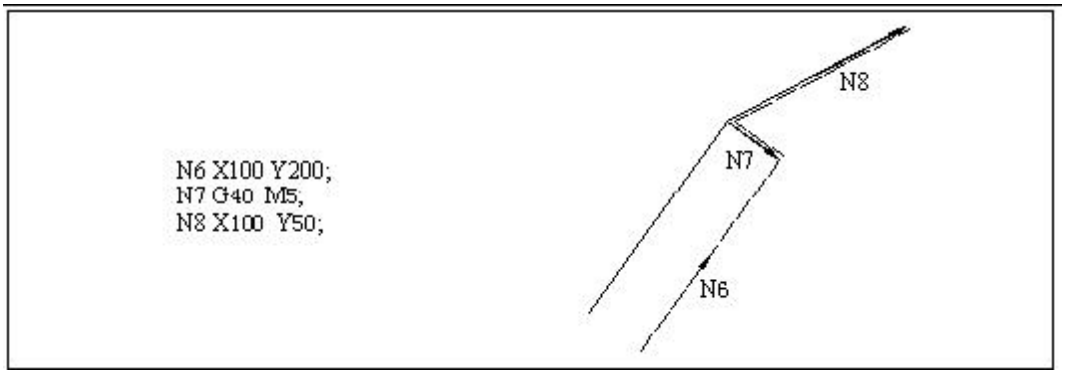
In compensation mode, if the segments without motion aren't specified consecutively for four and M instruction isn't restricted in advance, the intersection vector of usual path can be calculated.



If four segments without motion are specified consecutively and M instruction is restricted in advance, the compensation vector is made in the vertical direction of the end point of previous segment.

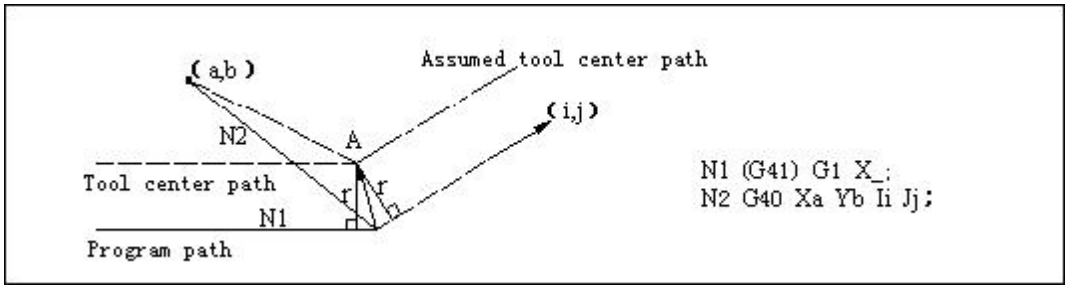


(3) Occasions that have instructions same to compensation cancellation instruction

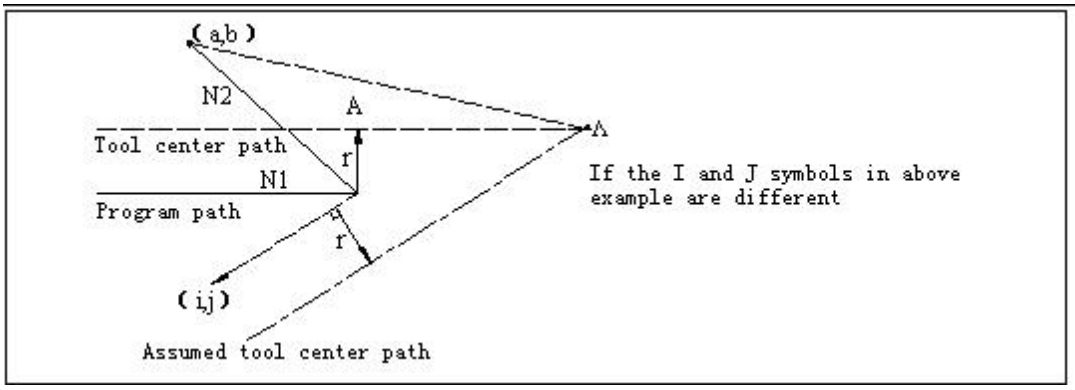


Occasions specified by I, J, K in G40

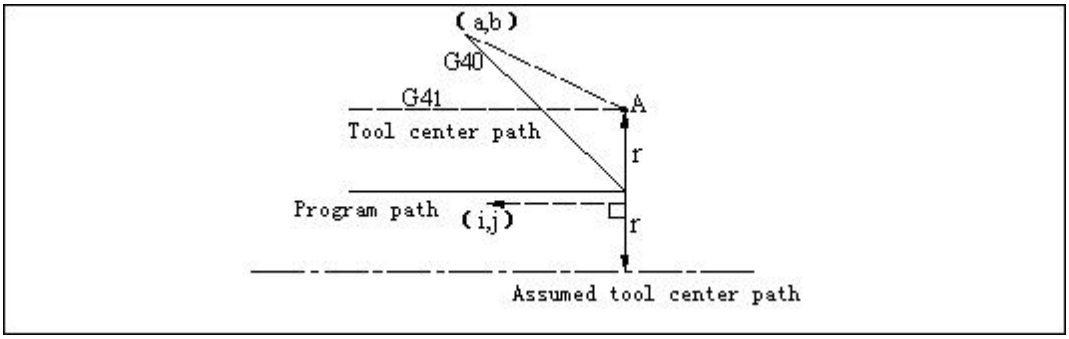
(1) In the four segments before G40 segment, if the last motion instruction segment is in G41 or G42 mode, the compensation cancels and the compensation direction doesn't change after the compensating from the last motion instruction end point to the intersection of tool center path of assumed motion instruction in I, J, K direction.



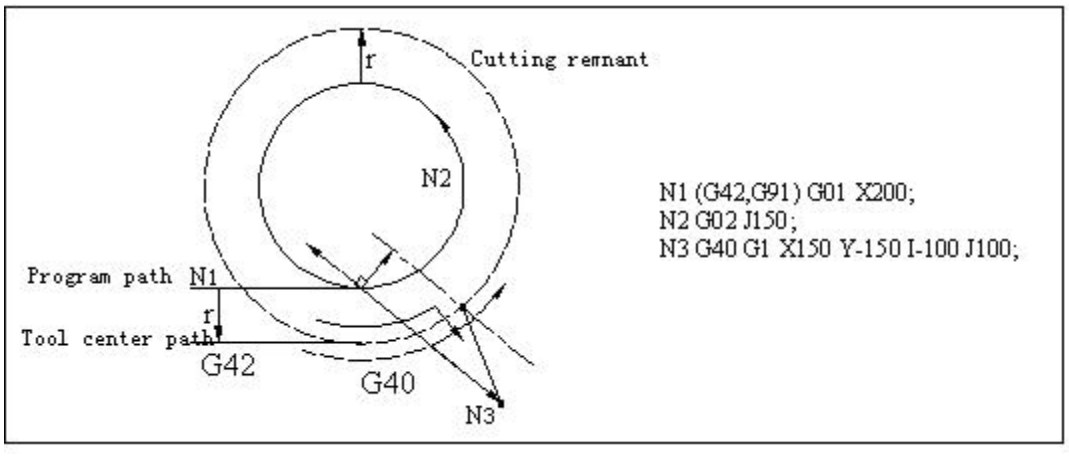
In this case, the compensation direction is shown in the figure below; although the compensation direction is different from the instruction direction, the intersection still can be calculated, and therefore attention is required.



Secondly, if the compensation of intersection calculation is high, vertical vector occurs in the program before G40.



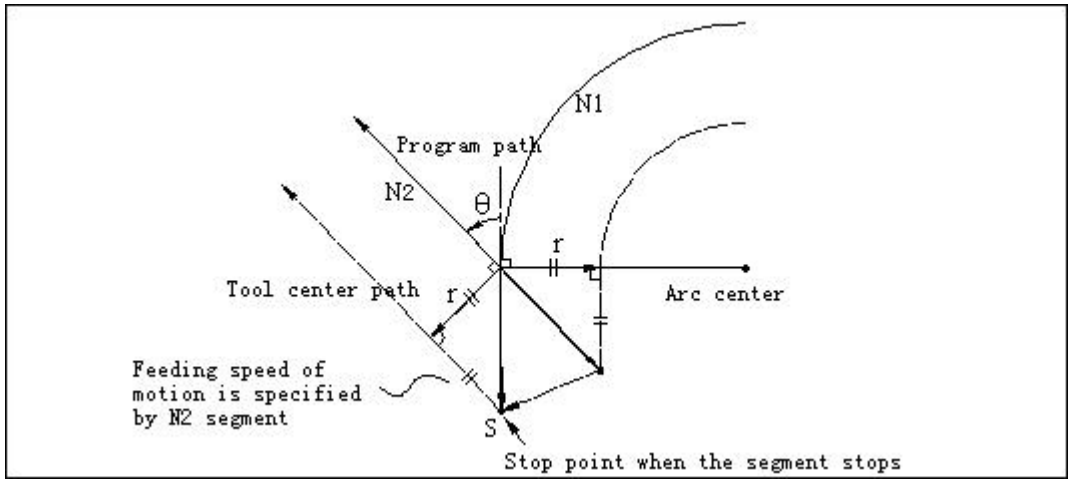
⑵ After the arc instruction, according to I, J, K vector of G40, if the arc path exceeds 360°, the uncut part occurs, and attention is required.



Corner motion

When the connection between motion instruction segments has several compensation vectors, the tool will move on the linear direction of the vectors, and this motion is called as corner rotation.

If these vectors are inconsistent, to move the corner, the motion action is executed in subsegment; therefore, in single segment mode, it will execute previous segment + corner motion of previous segment and keep connection motion + the secondary segment executes the corner motion of the other half in following operation.



2.3.3. G41/G42instruction and I, J, K designation

Function and purpose

If G41/G42 and I, J, K are specified in same segment, the compensation direction can be changed.

Format

G17 (XY plane)G41/G42 X_Y_I_J_;
G18 (ZX plane)G41/G42 X_Z_I_K_;

G19 (YZ plane)G41/G42 Y_Z_J_K_;

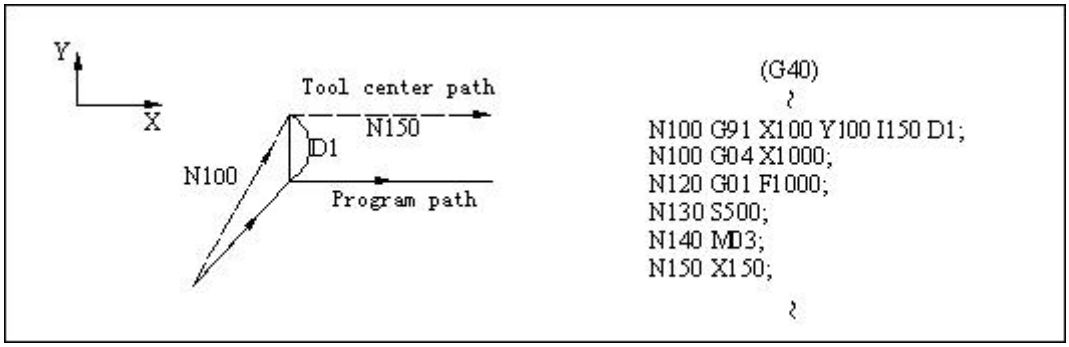
Then, the motion mode is used as linear instruction.

I, J vector (G17XY plane selection)

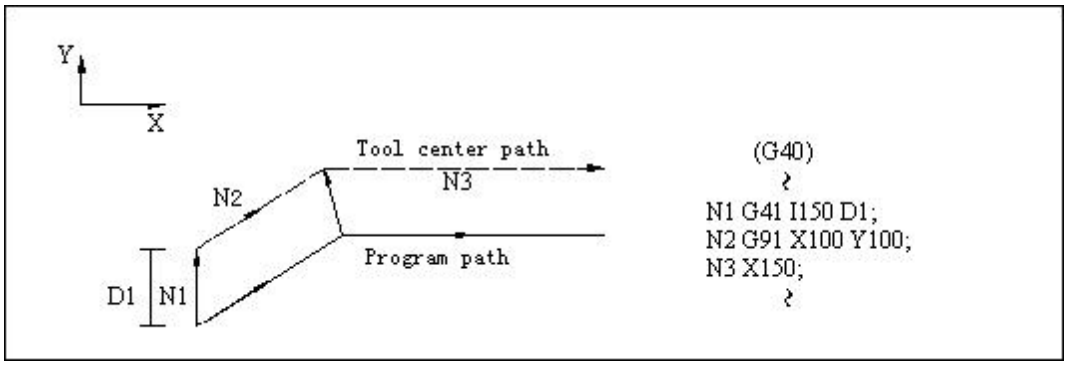
Now, using this instruction to generate new I, J vector (G17 plane) is described; similar description is also suitable for vector KI (G18 plane) and JK (G19 plane).

As shown in the figure below, I, J vector isn't related to the intersection calculation of program specified path, and only uses the vector in I, J specified direction and having same compensation. I, J vector can be specified when the compensation starts or in compensation mode.

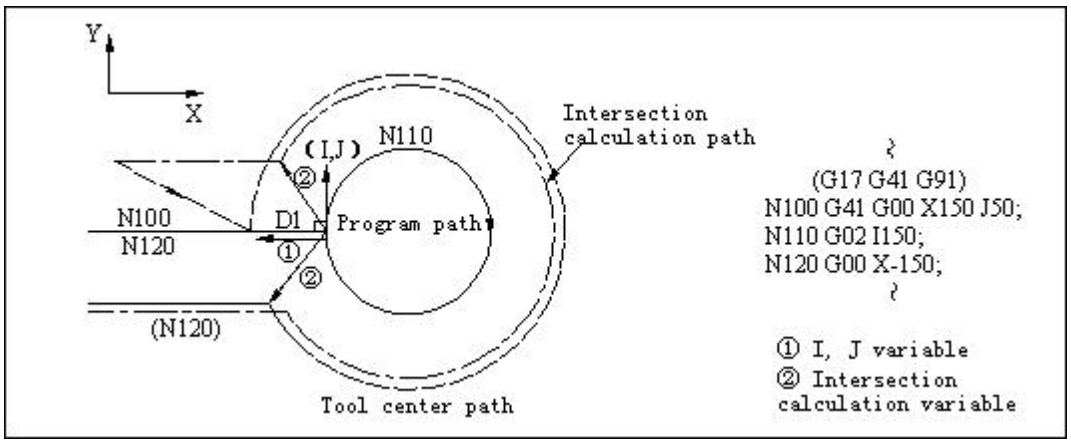
(1) I, J compensation specified occasion



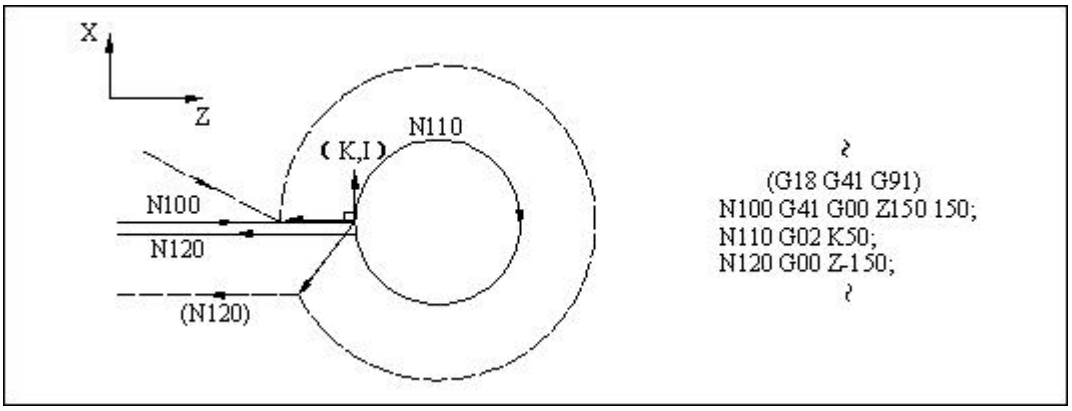
(2) Compensation without motion instruction



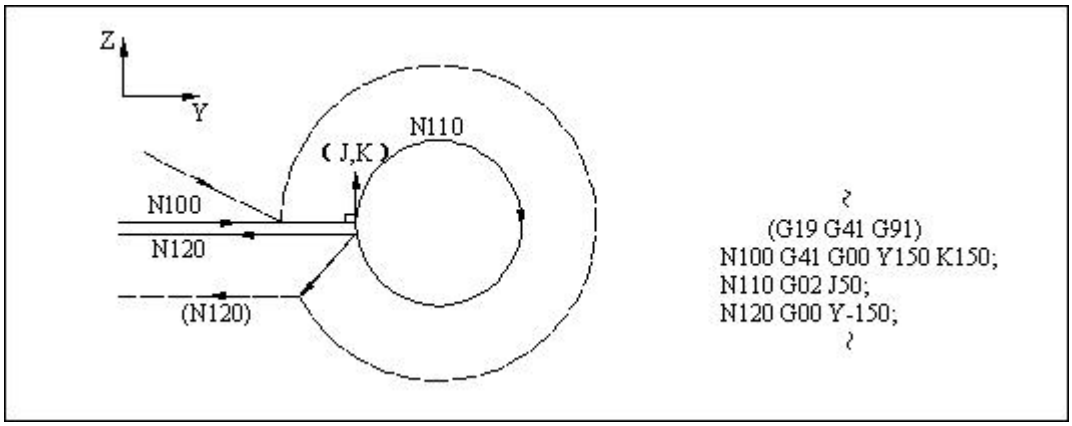
(3) I, J specified (G17) occasions in G41/G42 mode



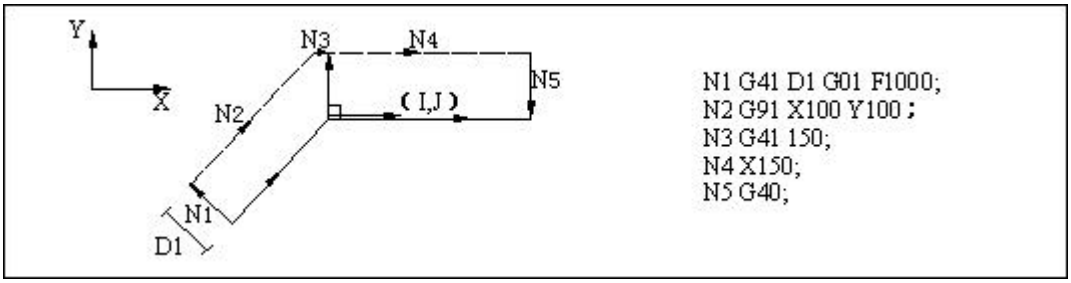
G18 plane



G19 plane



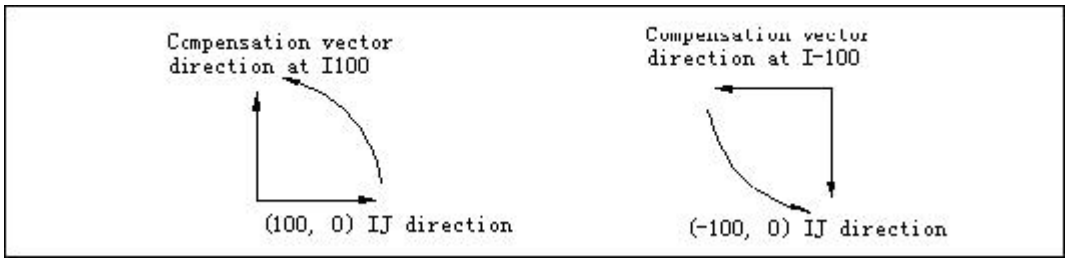
(4) If I, J is specified in the segment without motion



Direction of compensation vector

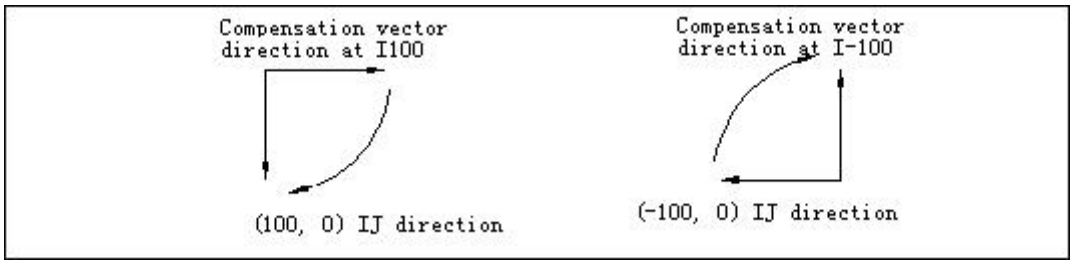
(1) In G41 mode

In the direction specified by I, J, rotate 90° to the left in the positive direction of Z axis.



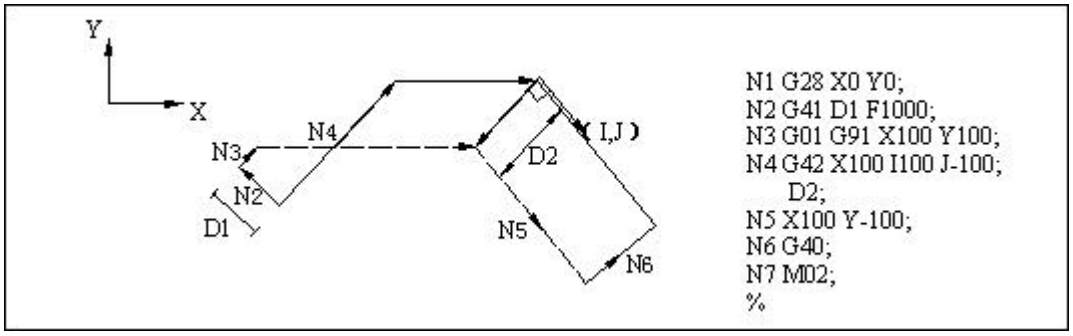
(2) In G42 mode

In the direction specified by I, J, rotate 90° to the right in the positive direction of Z axis.



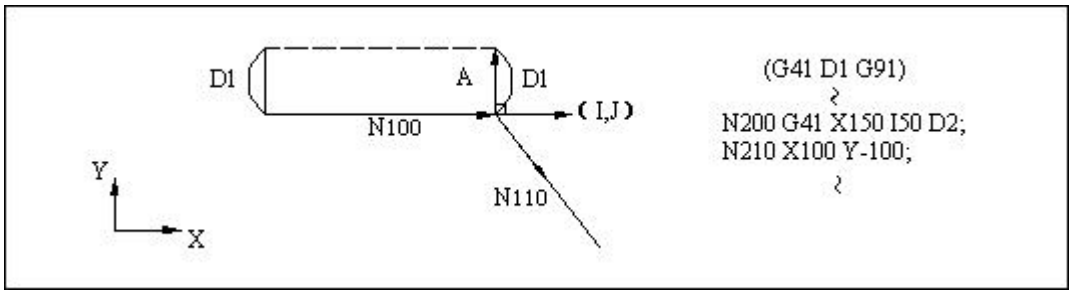
Switching compensation mode

In compensation mode, G41/G42 mode can be switched at any moment.

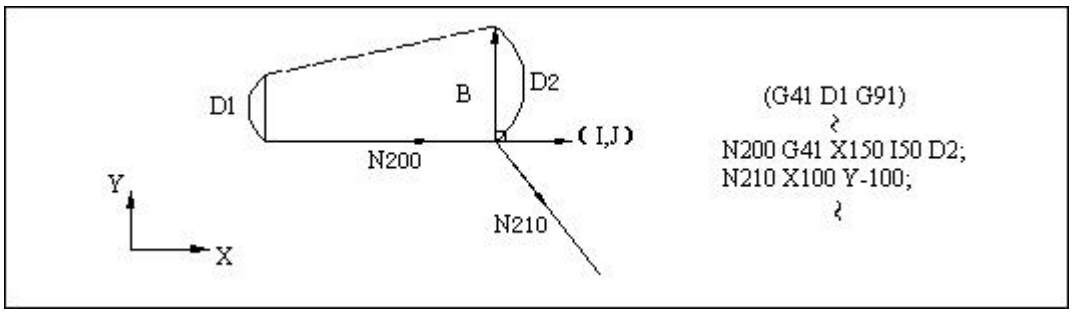


Compensation value of compensation vector

The compensation value is determined by I, J specified segment compensation No. (or mode).



The compensation value of vector O equals to the value recorded on compensation No. mode D1 of N100 segment.

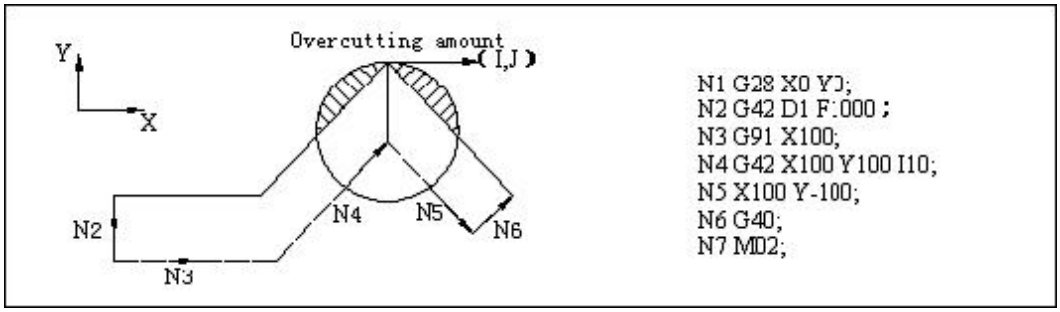


The compensation value of vector P equals to the value recorded on compensation No. mode D2 of N200 segment.

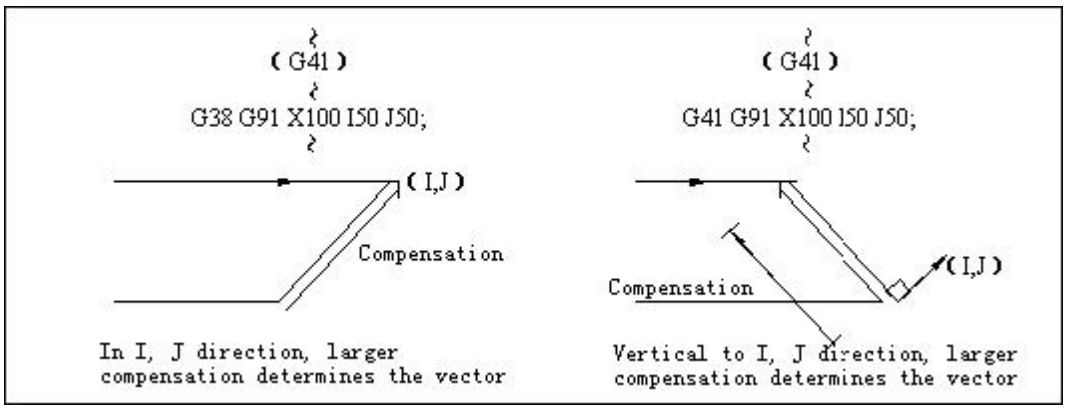
Other precautions

(1) If I, J vector is used, the compensation starts in linear mode (G00, G01). In arc mode, the program will alarm. In compensation mode, the IJ instruction in arc mode is the arc center.

(2) After I, J vector is made, the vector won't disappear even there is interference (no interference avoidance). Therefore, over cutting may occur sometimes.

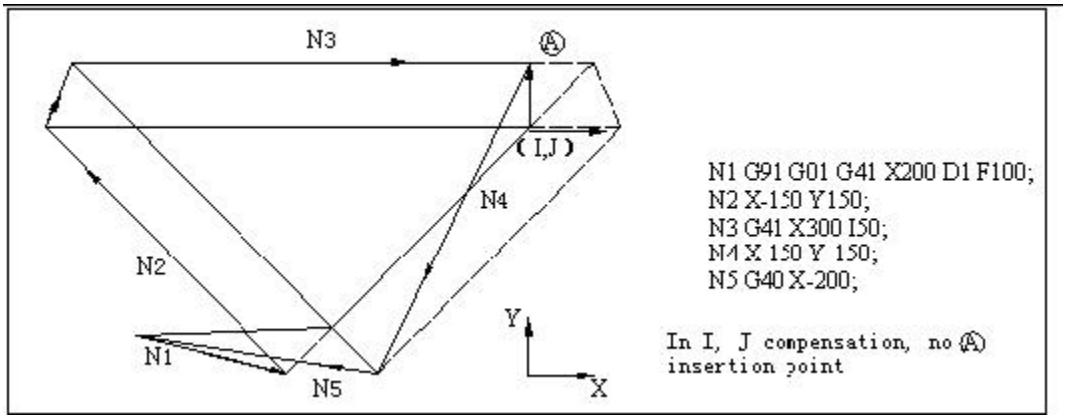


(3) G38 I_J_(K_) instruction and G41/G42 I_J_(K_) instruction specified different vectors.



(4) According to the combination of G41/G42 and I, J, K instructions, the compensation method follows:

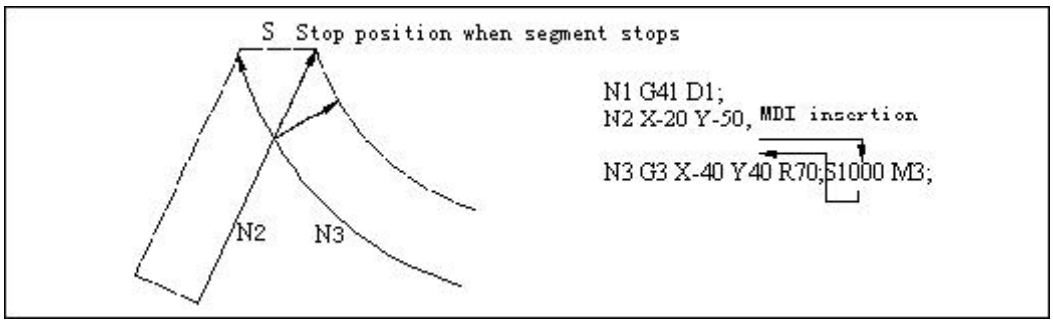
G41/G42	I,J,K	Compensation method
No	No	Intersection caculstion vector
No	Yes	Intersection caculstion vector
Yes	No	Intersection caculstion vector
Yes	Yes	I, J vector, no segment inserted



Insertion treatment during tool radius compensation

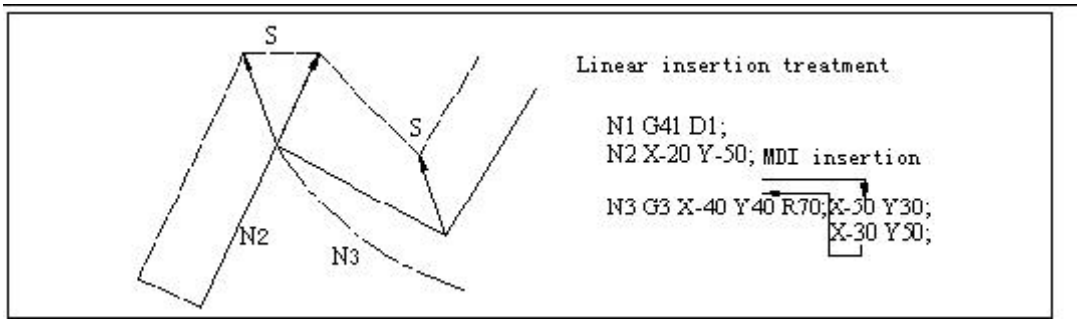
MDI insertion

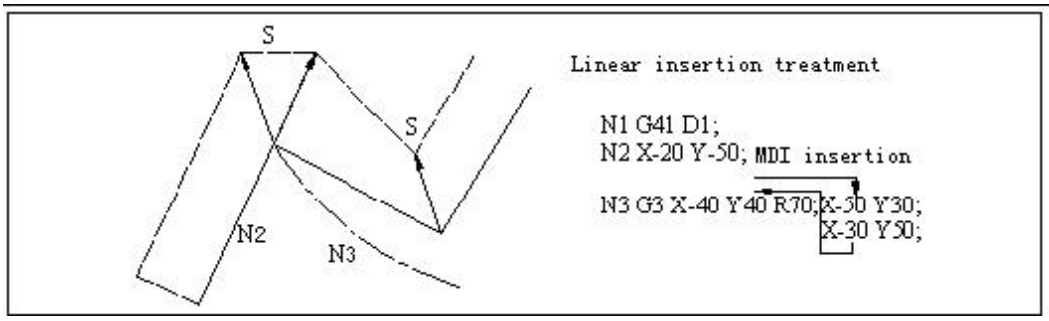
(1) Insertion treatment when there is no motion (tool track doesn't change)



(2) Insertion treatment when there is motion

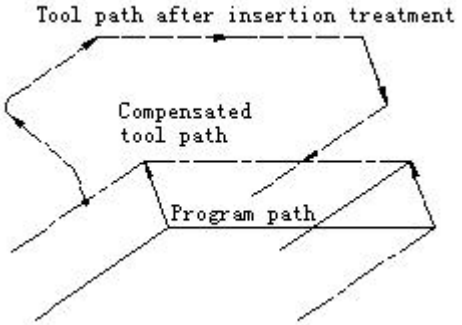
Insert the treated motion segment, and then the compensation vector calculates automatically.



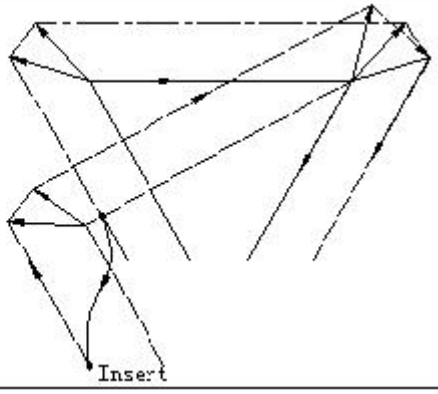


Manual insertion

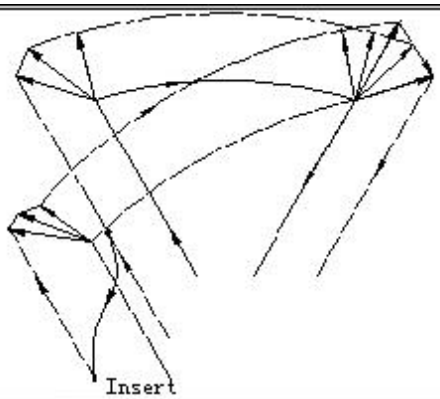
When manual absolute value is off, the insertion tool track is shifted for an offset that equals to the insertion



When manual absolute value is on, the insertion is in increment mode, and the action has the same result when manual absolute value is off.



In absolute value mode, the end point tool in inserted segment returns to its original path, as shown in the figure on the right.



2.3.4. Notes for tool radius compensation

(1) Specifying the compensation

The compensation is specified by D instruction and compensation No. Once D instruction is specified, this instruction is always valid until new D instruction is specified. P170 error will occur if specified with H instruction.

In addition to specifying the compensation of tool radius compensation, D instruction also can be used as the compensation value of tool position compensation.

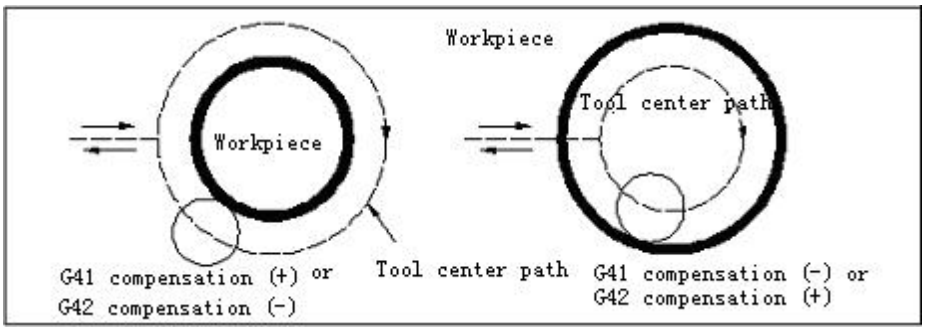
(2) Changing compensation

The compensation is usually changed after radius compensation mode is canceled and another tool is selected; in compensation mode, when the compensation is changed, the vector of segment end point is calculated according to the compensation specified by the segment.

(3) Compensation symbol and tool center path

If the compensation is negative (-), it is same to G41 and G42 switched circles; but the rotation outside of workpiece turns into inside rotation, and the inside rotation turns into outside rotation.

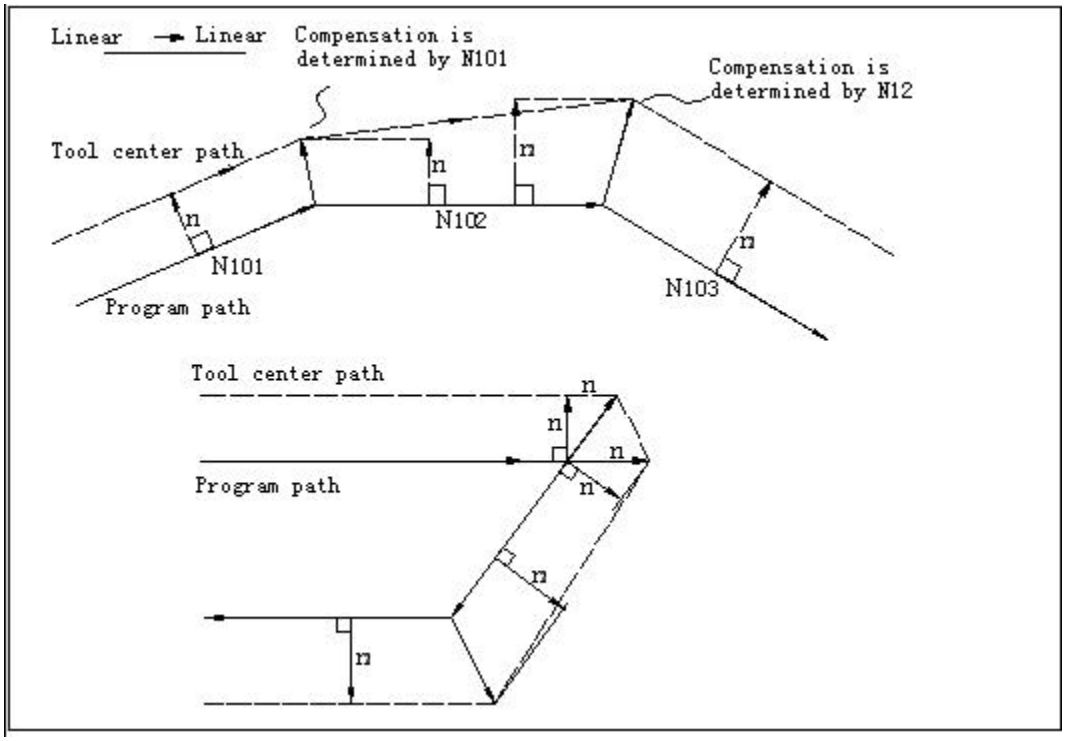
Generally, the compensation is made into program with positive (+) symbol. In the figure below, the tool center path in the left will be as in the right if the compensation turns to negative. Therefore, the processing shown in the figure below only needs to select the tolerance of them, adds in appropriate compensation, and then cut into two shapes with one program.

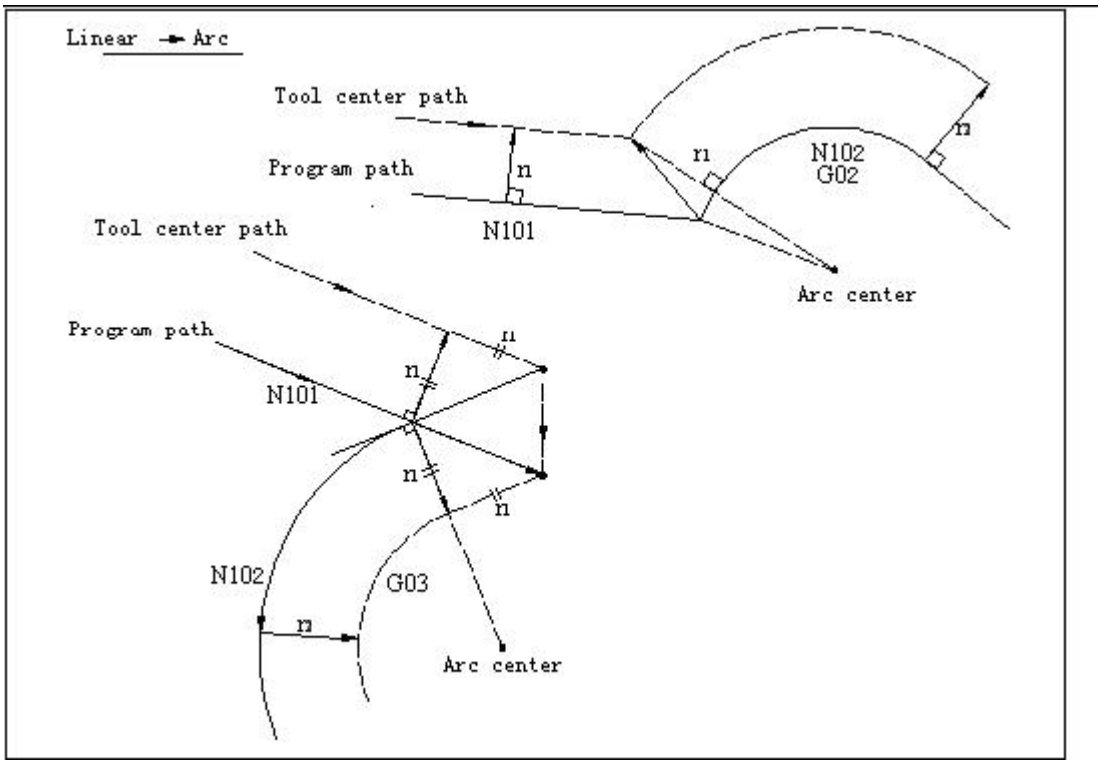


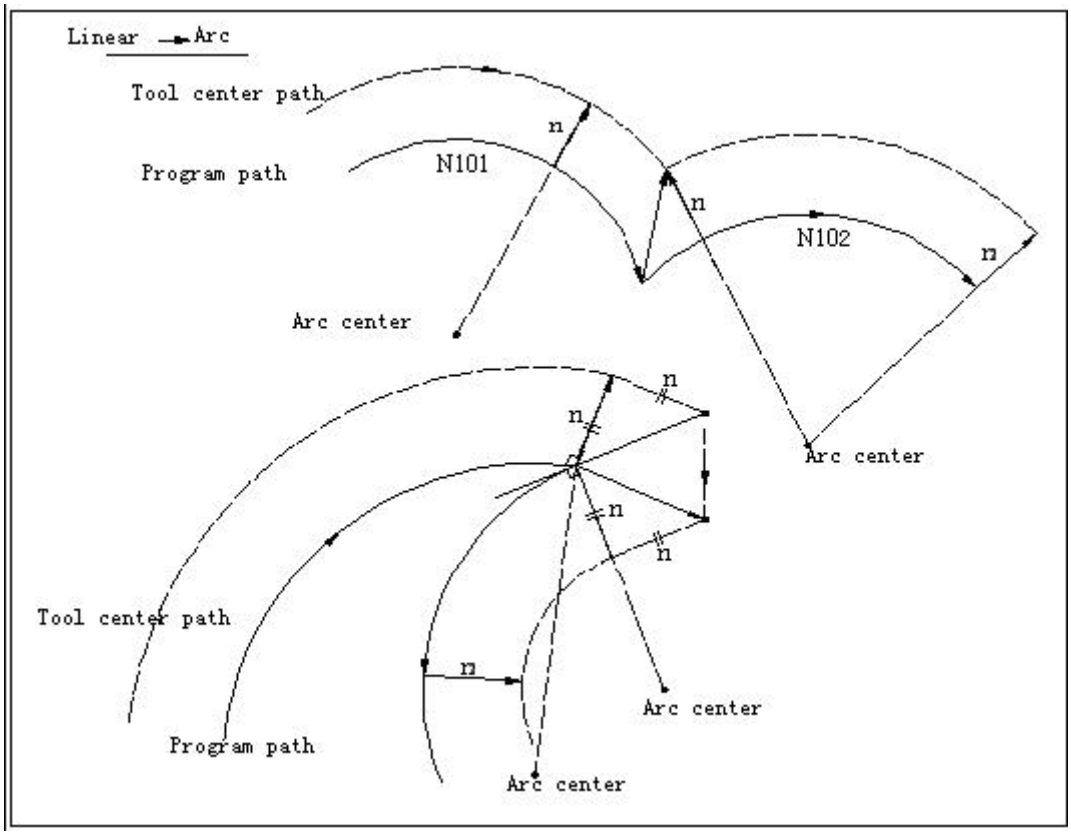
Compensation number change in compensation mode

In compensation mode, the compensation No. shouldn't be changed in principle. To change, the motion is shown in the figure below:

```
G41 G01….....Dr1;
α=0,1,2,3
N101 G00 α Xx1 Yy1;
N102 G00 α Xx2 Yy2 Dr2;.....compensation No. change
N103          Xx3 Yy3;
```







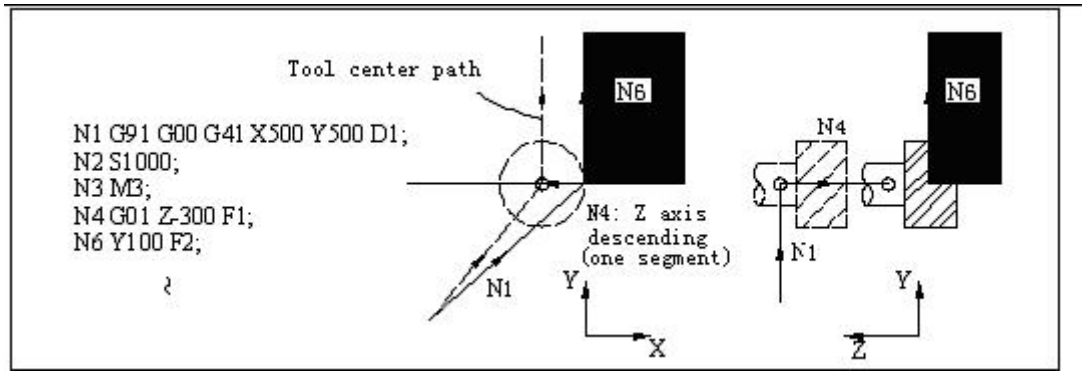
Tool radius compensation start and axis Z cut-in action

Function

Before cutting starts, make tool radius compensation (usually XY plane) action at the position before leaving the workpiece, and then Z axis can execute cutting; at this moment, Z axis motion can approach the workpiece quickly, and then executes cutting action, which contains two sections; please pay attention to the description below:

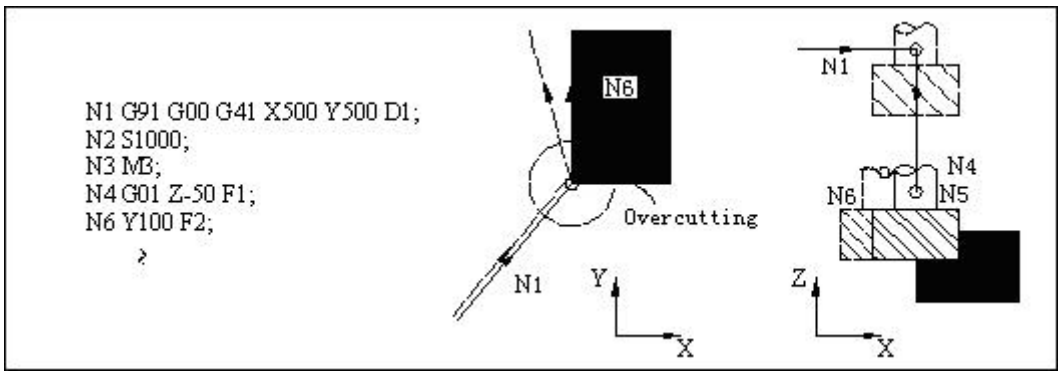
Example:

When programming as below



If above program, i.e. N1 compensation starts, pre-read to N6 segment, and then determine the relation between N1 and N6, and compensate appropriately as shown above.

Then, divide N4 segment into two in above program.

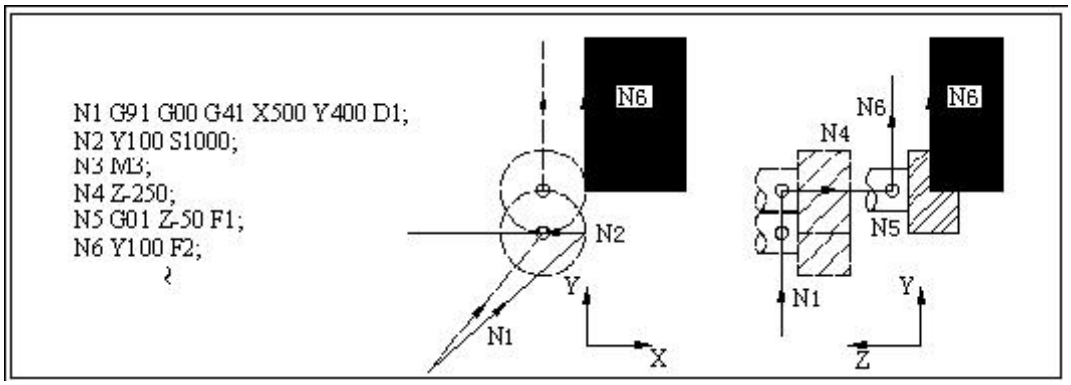


At this moment, there is no instruction segment of XY plane in the four continuous segments N2-N5, pre-reading isn't allowed from N1 to N6, and overcutting as above occurs.

Basic execution compensation is made with N1 only, but correct compensation vector can't be made, and thus overcutting occurs.

In this case, considering the calculation in NC, in the cutting direction after Z axis descends, before Z axis descends and cuts, and add the instruction of the same direction to prevent overcutting.

N2 and N6 have the same direction, thus the compensation can be executed properly.



N2 and N6 have same direction, and thus the compensation can be executed properly.

2.4 Hole processing function

Standard fixed cycle

With hole processing fixed cycle, the functions that require several segments in other method can be finished in one segment. Table 10.1 lists all hole processing fixed cycles.

Table 10.1: Hole Processing Fixed Cycle

G code	Processing motion (Z axis negative)	Hole bottom action	Return motion (Z axis positive)	Application
G73	Sub, cutting feeding	—	Quick positioning feeding	High speed deep hole drilling
G74	Cutting feeding	Pause-Spindle CW	Cutting feeding	Left-hand tapping cycle
G76	Cutting feeding	Spindle orientation stops	Rapid traverse	Fine boring cycle
G80	—	—	—	Cancel ficed cycle
G81	Cutting feeding	—	Rapid traverse	Common drilling cycle
G82	Cutting feeding	Pause	Rapid traverse	Drilling to rough boring
G83	Sub, cutting feeding	—	Rapid traverse	Deep hole drilling cycle

G84	Cutting feeding	Pause – spindle CCW	Cutting feeding	Right thread taping
G85	Cutting feeding	—	Cutting feeding	Boring cycle
G86	Cutting feeding	Spindle stop	Rapid traverse	Boring cycle
G87	Cutting feeding	Spindle CW	Rapid traverse	Boring cycle
G88	Cutting feeding	Pause- spindle stop	Manual	Boring cycle
G89	Cutting feeding	Pause	Cutting feeding	Boring cycle

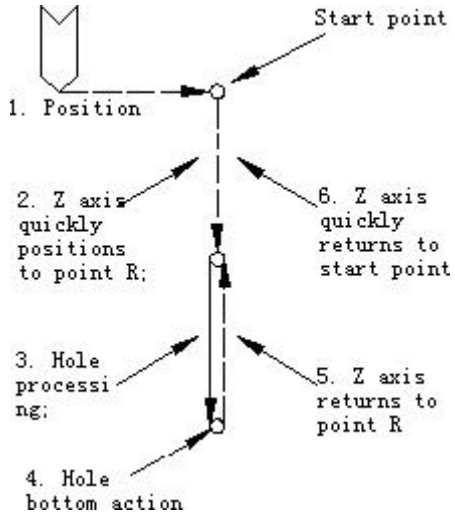
Format:

After G73/G74/G76/G81~G89, give hole processing parameters,
The format follows: (See table 10.2 for details)
G××X_ Y_ Z_ R_ Q_ P_ F_ K_ ;
G×× : hole processing method
X_ Y_ Z_ : position parameters of hole processed
R_ Q_ P_ F_ : hole processing parameters
K_ : repeat times

Details:

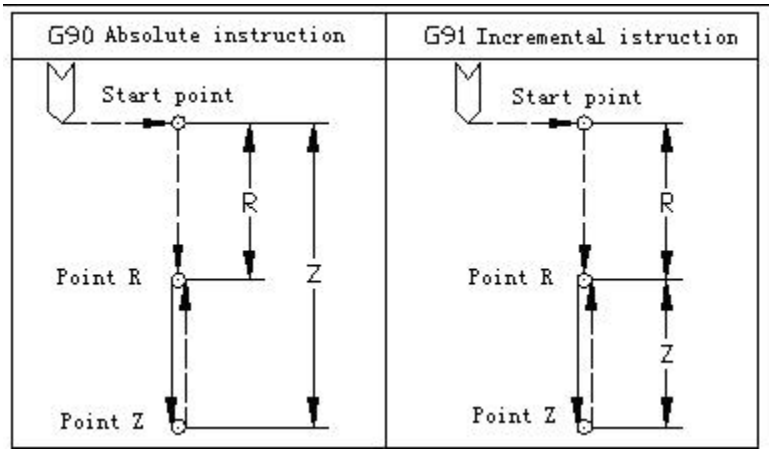
Generally, one hole processing fixed cycle completes the following six steps:

G73/G74/G76/G81~G89,
1. X, Y axis quick positioning. 2. Z axis quickly positions to point R. 3. Hole processing. 4. Hole bottom action. 5. Z axis returns to point R. 6. Z axis quickly returns to the start point.



Six Steps of Hole Processing Fixed Cycle

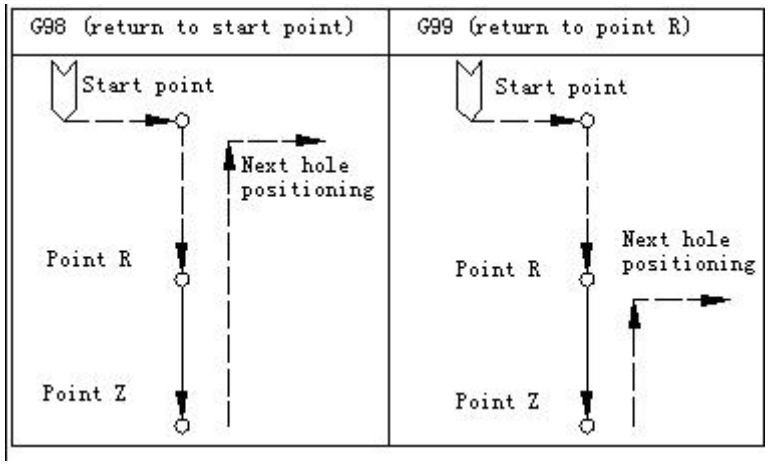
The instructions that have influence on the execution of hole processing fixed cycle instruction include G90/G91 and G98/G99. Fig. 10.2 shows the effect of G90/G91 on hole processing fixed cycle instruction.



Effect of G90/G91 on Hole Processing

G98/G99 determines fixed cycle returns to point R or the start point after hole processing; in G98 mode, Z axis returns to the start point after hole processing; in G99 mode, it returns to point R.

Generally, if the holes being processed are on a flat plane, we can use G99 instruction, because it will position next hole after returning to point R in G99 mode; in general programming, point R is close to workpiece surface, it will shorten part processing time; but if the workpiece surface has convex platform or tendon, the tool and workpiece may collide if G99 is used; at this moment, G98 should be used to return Z axis to the start point and then position next hole to ensure the safety. See the figure below.



Effect of G98/G99 on Hole Processing

Meaning of Every Address in Hole Processing Fixed Cycle

Address	Meaning
Position parameter X, Y of holes being processed	Specify the position of the hole being processing in increment or absolute mode; the track and speed of the tool moving to processed hole are same to G00
Position parameter Z of holes being processed	In absolute value mode, specify the position of hole bottom in Z axis direction; in increment value mode, specify the distance from point R to hole bottom
Hole processing parameter R	In absolute value mode, specify the position of point R in Z axis direction; in increment value mode, specify the distance from the start point to point R
Hole processing parameter Q	Used to specify the tool feeding of deep hole drilling cycle G73 and G83, and the offset of fine boring cycle G76 and reverse boring cycle G87 (always increment value instruction no matter G90 or G91 mode)
Hole processing parameter P	Used to specify the pause time (unit: sec) in the fixed cycle that hole bottom action has pause
Hole processing parameter F	Used to specify the cutting feeding speed in fixed cycle; in the fixed cycle, the motion from start point to point R and from point R to start point executes in the speed of quick feeding, the motion from point R to point Z executes in the cutting feeding speed specified by F, while the motion from point Z to point R executes either in the speed specified by F or quick feeding speed.

Address	Meaning
Repeat times K	Specify the repeat times of fixed cycle in current positioning point; if K isn't specified, NC considers that K=1; if K is specified as 0, the fixed cycle won't be executed at current point.

- The hole processing specified by Gxx is modular, and the fixed cycle can be canceled with G80 or 01 G instruction.
- Hole processing parameter is also modular, and will be retained before changed or fixed cycle is canceled, even hole processing mode is changed.
- A hole processing parameter can be specified or changed when specifying a fixed cycle or at any moment in the fixed cycle.
- Repeat times K isn't a modular value, and is only specified when required.
- Feeding speed F is a modular mode, and it will be retained even the fixed cycle is canceled.
- If NC system is reset while executing the fixed cycle, hole processing mode, hole processing parameter and repeat times K are canceled.

The following example describes above content better.

SN	Program content	Remark
1	S_ M03	Specify the rotation, and specify the spindle to rotate positively
2	G81X_Y_Z_R_F_K_	Locate specified X, Y point quickly, process with the hole processing parameter specified by Z, R, F and in the hole processing mode specified by G81, and repeat for K times; when the fixed cycle is started, Z, R, F are necessary hole processing parameters.
3	Y_	X axis doesn't move, Y axis quickly positions the instruction point and processes the hole; the hole processing parameter and hole processing mode retain the modular value in 2. The K value in 2 is invalid.
4	G82X_P_K_	Hole processing mode is changed; hole processing parameter Z, R, F retain the modular value, specify the value of hole processing parameter P and specify the repeat times K.
5	G80X_Y_	The fixed cycle is canceled, and all hole processing parameters except F are canceled.
6	G85X_Y_Z_R_P	Since the fixed cycle has been canceled when executing 5, all

	_	necessary hole processing parameters except F must be re-specified, even if these parameters aren't changed.
7	X_Z_	X axis positions the instruction point and processes the hole, and hole processing parameter Z is changed in this segment.
8	G89X_Y_	Position the XY instruction point and process the hole, and the hole processing mode is changed to G98. R, P are specified by 7, and Z is specified by 7.
9	G01X_Y_	The fixed cycle mode is canceled, and all hole processing parameters except F are canceled.

In the following diagrams, we use the modes below to indicate the feeding of every segment:

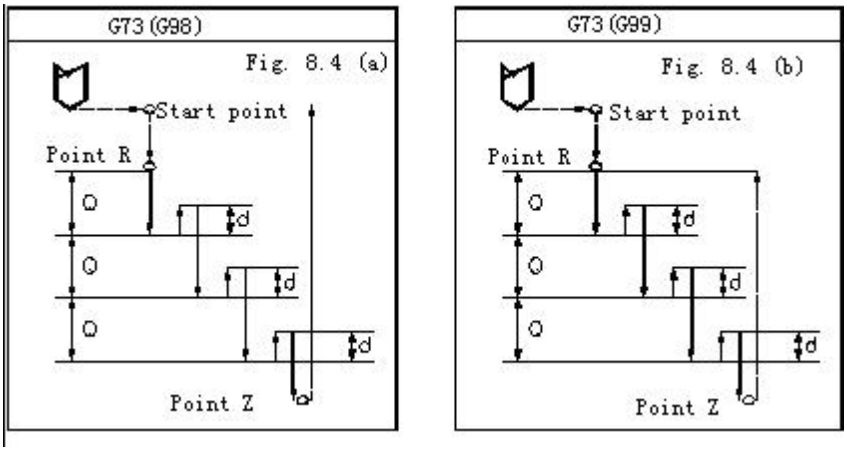
Indicate motion in quick feeding speed ·····→ Indicate motion in cutting feeding speed ——→ Indicate manual feeding - - - - - - - - →

2.4.1. High-speed deep-hole drilling cycle (G73)

Format:

Format: G73 X_ Y_ Z_ R_ Q_ F_

Details:



High Speed Deep Hole Drilling Cycle Diagram

The feeding from point R to point Z is finished in several segments; after cutting every segment, Z axis lifts for certain distance, and then executes cutting feeding for next segment.

The distance that Z axis lifts every time is d, which is specified by parameter P1.015,G73(M)circulating tool

retraction amount (mm), and the depth of every feeding is determined by hole processing parameter Q.

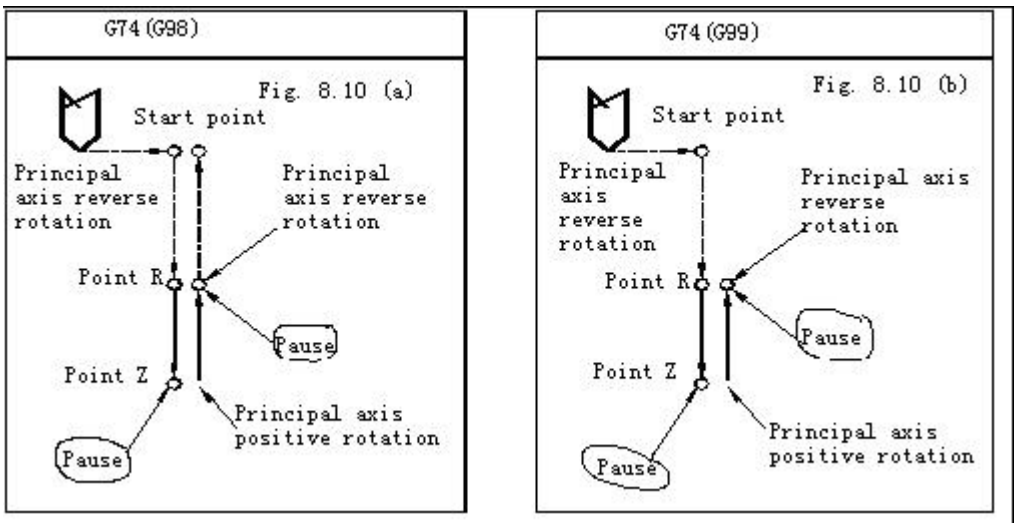
This fixed cycle is mainly used for processing holes with small diameter/depth ratio (e.g. Φ5, depth 70), and Z axis lift has the effect of chip breaking after cutting and feeding every segment.

2.4.2. Left-hand Thread tapping cycle (G74)

Format:

Format G74 X_ Y_ Z_ R_ F_(D_)
X_Y_ :thread position
Z_ :Threaded hole bottom position
R_ :point of tool feeding/ retreating
P_ :Dwell time at the bottom of hole
F_(D_) :convert to feeding speed according to screw distance, or specify the distance with D
K_ :repeat times (If necessary)

Details:



Left-hand Thread Taping Cycle Diagram

The sequence of actions is as follows:

Quickly locate it to the hole (X Y, but the tool maintains the original height)

Quickly locate it to point R

Tapping starts, and spindle reverses

Cut to the bottom position of the hole (Z) with the set cutting feedrate and spindle speed

Spindle stops; and if P is specified, it pauses at hole bottom (P) ms

The spindle rotates forward, and cut to the (R) point position with the set cutting federate and spindle speed.

Tapping finishes, and returns to initial point of tool under the G98 mode quickly, and returns to the (R) position under G99 mode.

If K (K>1) is specified, repeat the above actions 2~7 until the specified number of drill holes repeats is completed.

In the G94 (feed per minute) mode, the cutting feedrate is the spindle speed (S) * the pitch of the thread (PITCH). And, in the G95 (feed per revolution) mode, the cutting speed feedrate F is thread pitch (PITCH).

Notice:

In G74 and G84 cycle, the feeding rate switch and feeding retaining switch are ignored, i.e. feeding rate is retained at 100%, and can't be stopped before a fixed cycle completes; before cycle starts, the spindle should be specified to rotate in taping direction.

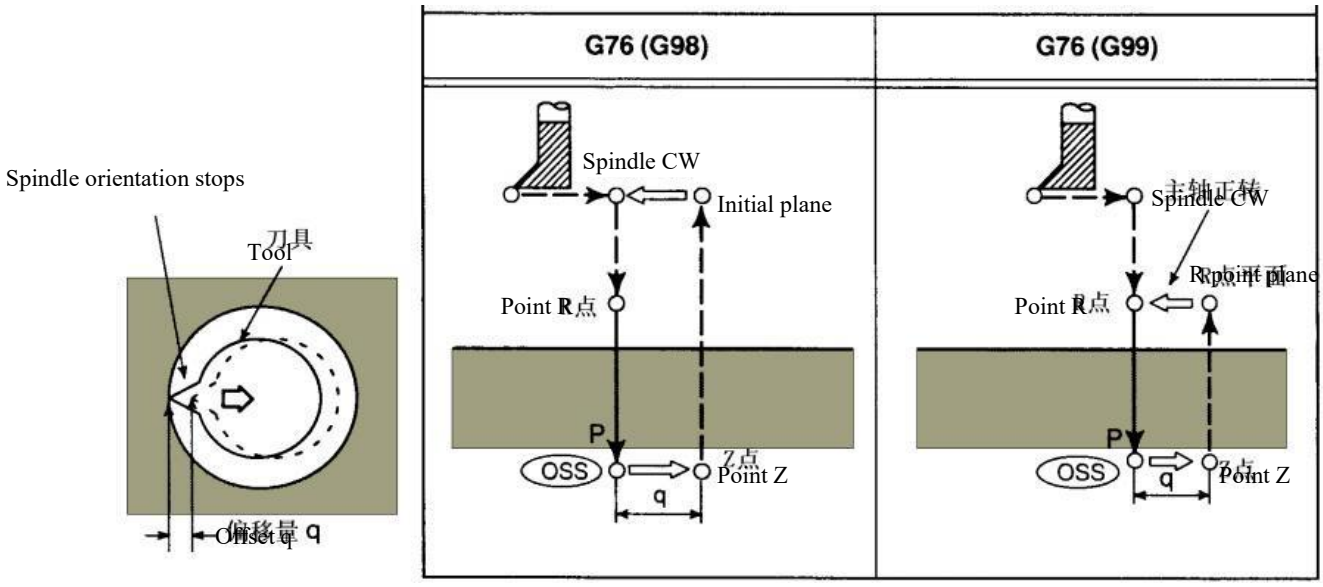
2.4.3. Fine boring cycle (G76)

Fine boring, the precision hole by boring

Format:

Format: G76 X_ Y_ Z_ R_ Q_ F_	
X_ Y_	Hole position
Z_	Threaded hole bottom position
R_	Point of borehole engaging/retracting
Q_	Offset of tool in X direction
P_	Dwell time at the bottom of hole
F_	Cutting feed rate
K_	Repeat times (if necessary)

Details:



(Note: The offset of Q at the bottom of hole is the modal value stored in the fixed cycle. It must be specified carefully, because this value can also be used as the cutting depth in the G73 and G83 commands. In addition, since this command requires the spindle to position the borehole, only the servo spindle can execute this command).

The sequence of actions can be as follows:

1. Perform G00 command to quickly locate to the positions of X and Y coordinates
2. Perform G00 to position R
3. Perform G01 borehole from point R to Z
4. Perform spindle positioning (Note: Perform servo spindle zeroing--by setting the spindle parameter [No. 15 spindle return-to-zero offset (Angle)] so that the boring tool tip is parallel to the X-axis negative direction)
5. The tool moves Q_ in the opposite direction of the tool tip using X-axis (note: this offset Q_ is a relative movement amount)
6. Return from the hole bottom position to the initial plane (G98 mode) or back to the R point (G99 mode) to perform G00 fast positioning.
7. The tool moves Q_ in the direction of the tool tip using X-axis (note: this offset Q_ is the relative movement amount)
8. Perform M03 spindle forward rotation

Cancel fixed cycle (G80)

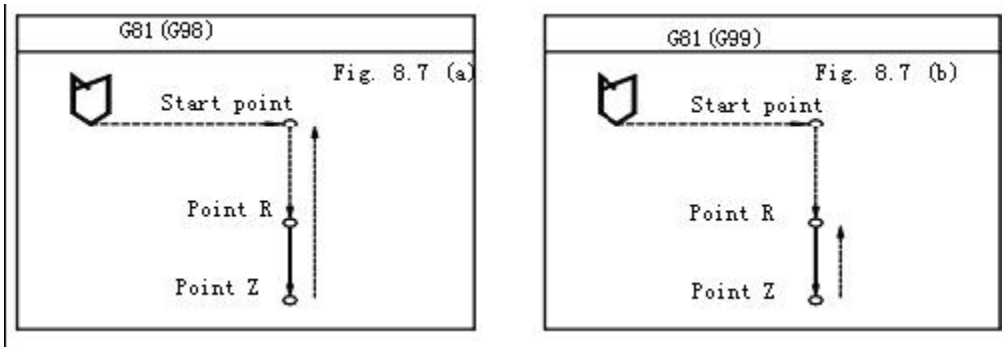
After G80 instruction is executed, the fixed cycle (G73, G74, G81~g89) instruction is canceled, point R and point Z parameter and all hole processing parameter except F are canceled. In addition, the G codes of group 01 also have the same effect.

2.4.4. Drilling cycle (G81)

Format:

Format G81 X_ Y_ Z_ R_ F_

Details:



G81 is the simplest fixed cycle, and its execution process follows:

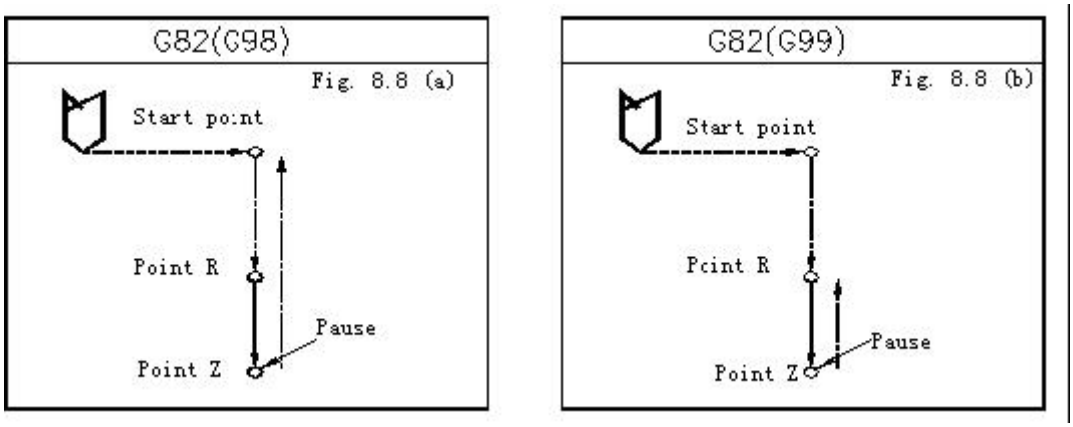
X, Y positioning, Z axis moves to point R quickly, and feeds to point Z at F speed, Quickly returns to the start point (G98) or point R (G99), No hole bottom action

2.4.5. Drilling cycle, rough boring cycle (G82)

Format:

G82 X_ Y_ Z_ R_ P_ F_

Details:



Drilling Cycle, Rough Boring Cycle Diagram

☞ **Note:**

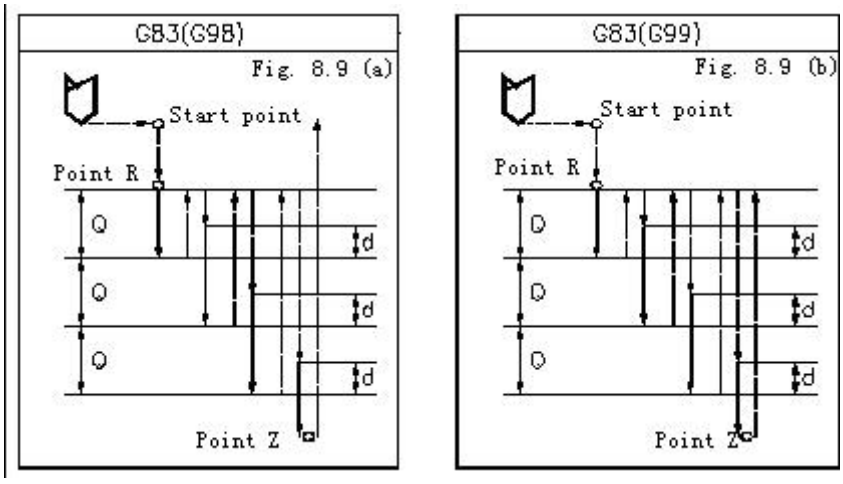
G82 fixed cycle has a pause action in the hole bottom, and others are same to G81. The pause of hole bottom can improve the precision of hole depth.

2.4.6. Deep-hole drilling cycle (G83)

Format:

G83 X_ Y_ Z_ R_ Q_ F_

Details:



Deep Hole Drilling Cycle (G83) Diagram

☞ **Note:**

Similar to G73 instruction, the feeding from point R to point Z under G83 instruction is also finished in two segments; different from G73 instruction, Z axis returns to point R after feeding of every segment, and then moves to position d above the start point at the quick feeding speed and starts the feeding of

next segment. The distance of every feeding is specified by hole processing parameter Q, which is always positive; the value of d is specified by 532# machine tool parameters.

2.4.7. Tapping cycle (G84)

Format:

G84 X_ Y_ Z_ R_ P_ F_(D_)

X_Y_ :thread position

Z_ :threaded hole bottom position

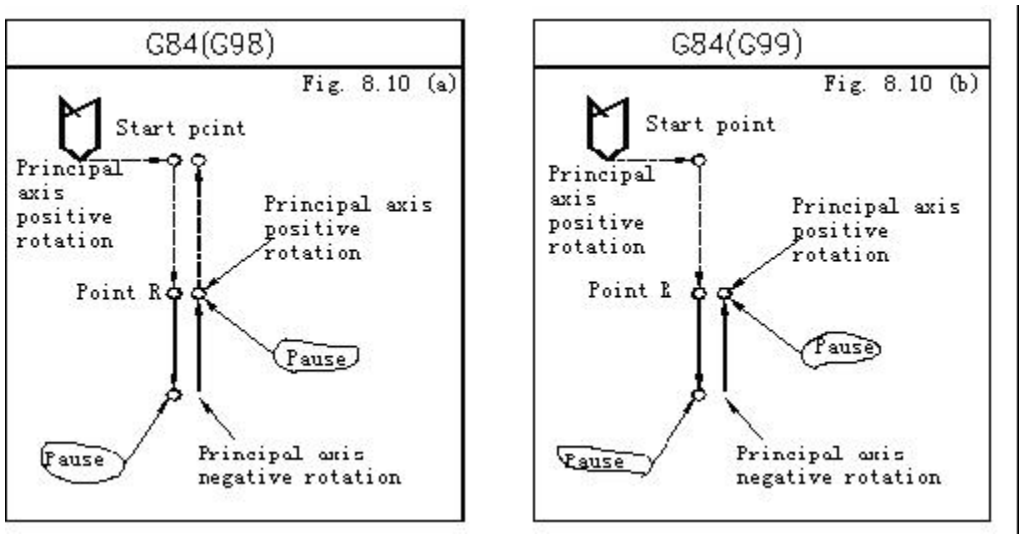
R_ :point of tool feeding /retreating

P_ :Dwell time at the bottom of hole

F_(D_) :convert to the feeding speed according to screw distance, or specify the screw distance with D_ directly

K_ :repeat times (if necessary)

Details:



Taping Cycle Diagram

The sequence of actions is as follows:

1. Quickly locate it to the hole (X Y, but the tool maintains the original height)
2. Quickly locate it to point R
3. Tapping starts, and spindle reverses
4. Cut to the bottom position of the hole (Z) with the set cutting feedrate and spindle speed
5. Spindle stops; if P is specified, it pauses at hole bottom (P) ms

6. The spindle rotates forward, and cut to the (R) point position with the set cutting federate and spindle speed.
7. Tapping finishes, and returns to initial point of tool under the G98 mode quickly, and returns to the (R) position under G99 mode.
8. If K (K>1) is specified, repeat the above actions 2~7 until the specified number of drill holes repeats is completed.
9. In the G94 (feed per minute) mode, the cutting feedrate is the spindle speed (S) * the pitch of the thread (PITCH). And, in the G95 (feed per revolution) mode, the cutting speed feedrate F is thread pitch (PITCH).

Notice:

In G74 and G84 cycle, the feeding rate switch and feeding retaining switch are ignored, i.e. feeding rate is retained at 100%, and can't be stopped before a fixed cycle completes; before cycle starts, the spindle should be specified to rotate in taping direction.

The programming example is as follows (tap with a pitch of 1 mm in therigid tapping mode):

O1234

G17 G90 G00 G54 X0. Y0.

G00 Z50

M29 S1000 (Note: M29 enters the rigid tapping mode, M28 cancels the rigid tapping mode <After executing the M28 command, the G84 command is executed as the elastic tapping mode>)

G84 X0. Y0. Z-30. R10. F1000.(Note: Cutting feed F = spindle speed S* thread pitch 1mm = 1000mm/min)

X-15.

X-30.

X-30. Y15

G80 G91 G28 Z0.

M28

G28 X0. Y0.

M30

%

(Tap with a pitch of 1 mm in the elastic tapping mode):

O1234

G17 G90 G00 G54 X0. Y0.

G00 Z50

M28 S1000 (Note: M29 enters the rigid tapping mode, M28 cancels the rigid tapping mode <After

executing the M28 command, the G84 command is executed as the elastic tapping mode>)

G84 X0. Y0. Z-30. R10. F1000 (Note: Cutting feed F = spindle speed S* thread pitch 1mm = 1000mm/min)

X-15.

X-30.

X-30. Y15

G80 G91 G28 Z0.

M28

G28 X0. Y0.

M30

%

When tapping with the Z-axis, the machining plane is G17 (X_Y).

The gear ratio of the spindle parameters No. 23 and No. 24 is generally 1/36, and the specific ratio is input according to the actual calculation result.

Description of the parameters related to tapping:

1. [Spindle] The numerator item of parameter 23 spindle gear ratio and the denominator item of No. 24 spindle gear ratio, (A) In the rigid tapping mode, the multiplication ratio and division ratio setting method of the servo spindle are the same as the No.1 parameter in [Axis Parameter], which are only set as the servo spindle and set by the rotary axis. (B) In the elastic tapping mode, the ratio of spindle encoder in a circle to the spindle tool in a circle is generally 1: 1. If there is a special case, for example, the ratio of spindle encoder in two circles to the tool of spindle in a circle is 2:1.

2. [Spindle] For the No.1 parameter, the spindle specifies the interface number. Note: If it is an analog spindle, this parameter must be set to 0. If it is a servo spindle, it is necessary to set the pulse axis number of the drive servo spindle.

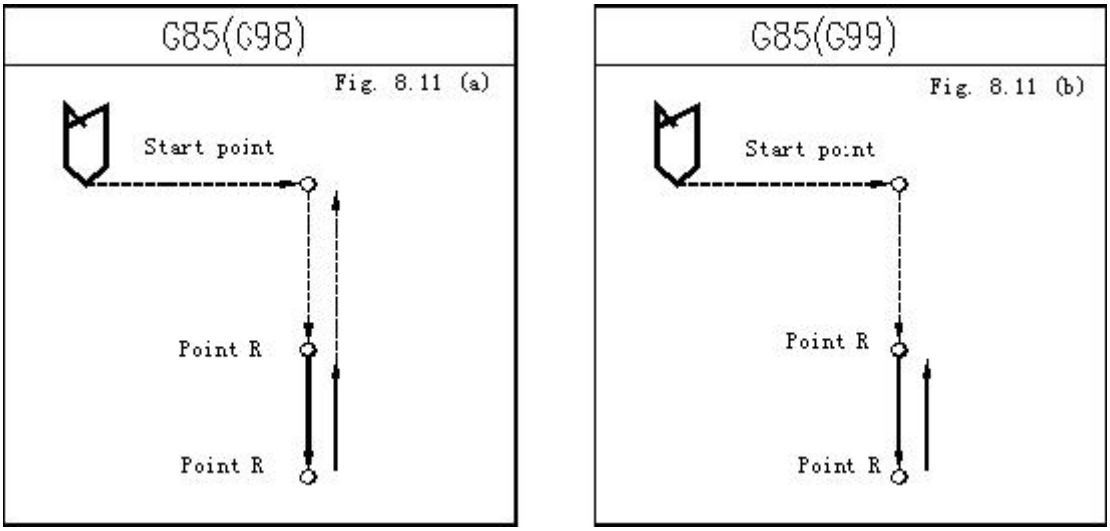
3. [Spindle] No.14 parameter, the number of spindle encoder lines, it is necessary to set this parameter when using the analog spindle + spindle encoder for elastic tapping, so that the Z-axis and the analog spindle can be used to follow the tapping function through the position of the encoder feedback. Do not set this parameter if the analog spindle is not connected to the encoder or if it is the servo spindle using pulse control. This parameter is defaulted to 0.

2.4.8. Boring cycle (G85)

Format:

G85 X_ Y_ Z_ R_ F_

Details:



Boring Cycle (G85) Diagram

This fixed cycle is very simple and the execution process follows:

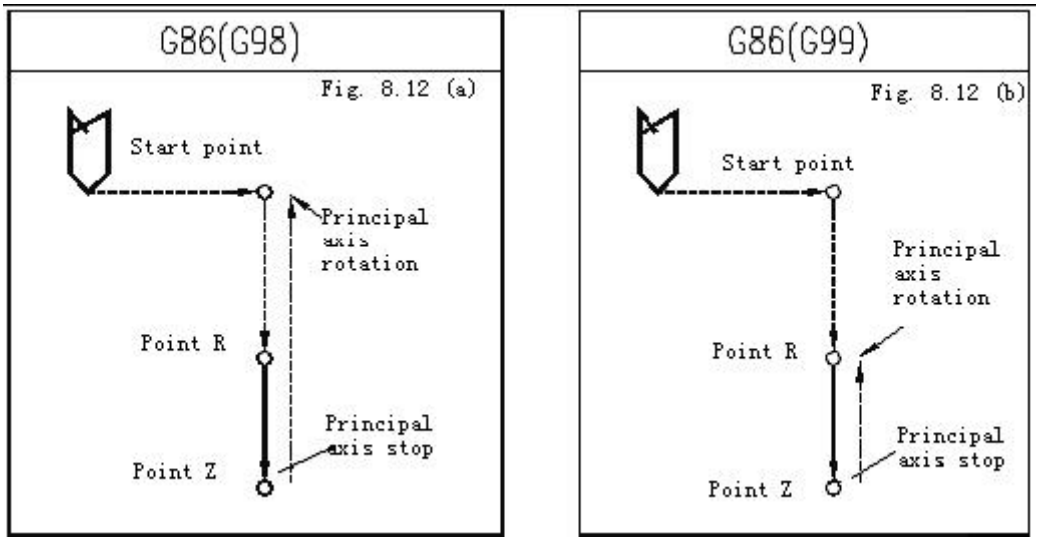
X, Y positioning, Z axis quickly moves to point R, feeds to point Z at the speed specified by F, Returns to point R at the speed specified by F, In G98 mode, return to point R and return to the start point quickly.

2.4.9. Boring cycle (G86)

Format:

G86 X_ Y_ Z_ R_ F_

Details:



Boring Cycle (G86) Diagram

☞ **Note:**

The execution of this fixed cycle is similar to G81; the difference is that the tool feeds to hole bottom in G86 to make the spindle stop, and quickly returns to point R or the start point to make the spindle to rotate in original direction and at original rotation.

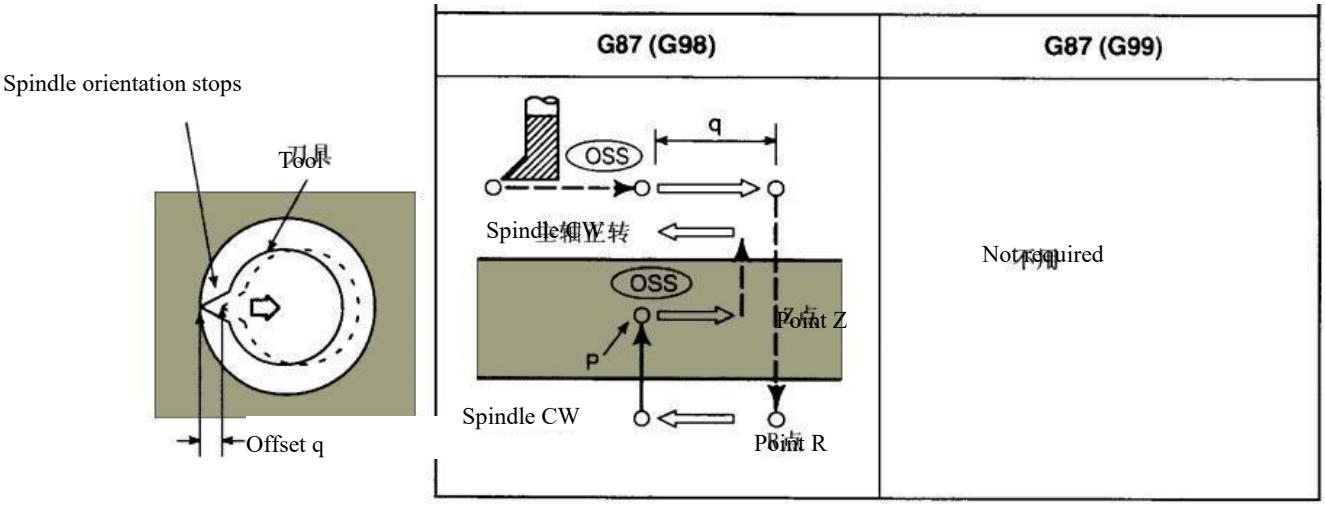
2.4.10. Back boring cycle (G87)

This cycle performs precision boring

Format:

Format G76 X_ Y_ Z_ R_ Q_ F_	
X_ Y_	Hole position
Z_	Note: Here Z is from the hole bottom to the Point Z
R_	Note: Here R is the position of hole bottom
Q_	Offset of tool in X direction
P_	Dwell time at the bottom of hole
F_	Cutting feed rate
K_	Repeat times (if necessary)

Details:



(Note: The offset of Q at the bottom of hole is the modal value stored in the fixed cycle. It must be specified carefully, because this value can also be used as the cutting depth of the G73 and G83 commands. In addition, since this command requires the spindle to position the borehole, so only the servo spindle can execute this command).

The sequence of actions can be as follows:

1. Perform G00 command to quickly locate to the positions of X and Y coordinates

2. Perform spindle positioning (Note: Perform servo spindle zeroing--by setting the spindle parameter [No. 15 spindle return-to-zero offset (Angle)] so that the boring tool tip is parallel to the X-axis negative direction)

3. The tool moves Q_in the opposite direction of the tool tip using X-axis (note: this offset Q_ is a relative movement amount)

4. Perform G00 fast positioning to the bottom of the hole (Point R)

5. The tool moves Q_ in the direction of the tool tip using X-axis (note: this offset Q_ is a relative movement amount)

6. Perform M03 spindle forward rotation

7. Execute the action of which G01moves to the Point Z in the positive direction of the Z-axis from the empty borehole

8. Perform spindle positioning (perform servo spindle zeroing--by setting the spindle parameter [No. 15 spindle return-to-zero offset (Angle)] so that the boring tool tip is parallel to the X-axis negative direction)

9. The tool moves Q_ in the opposite direction of the tool tip using X-axis (note: this offset Q_ is a relative movement amount)

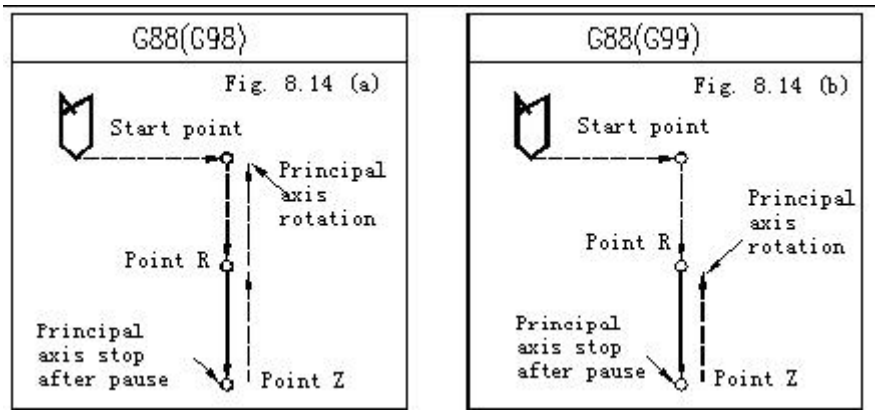
10. Perform G00 Z axis positioning to the starting position

11. The tool moves Q_ in the direction of the tool tip using X-axis (note: this offset Q_ is a relative movement amount)

12. Perform M03 spindle forward rotation

2.4.11. Boring cycle (G88)

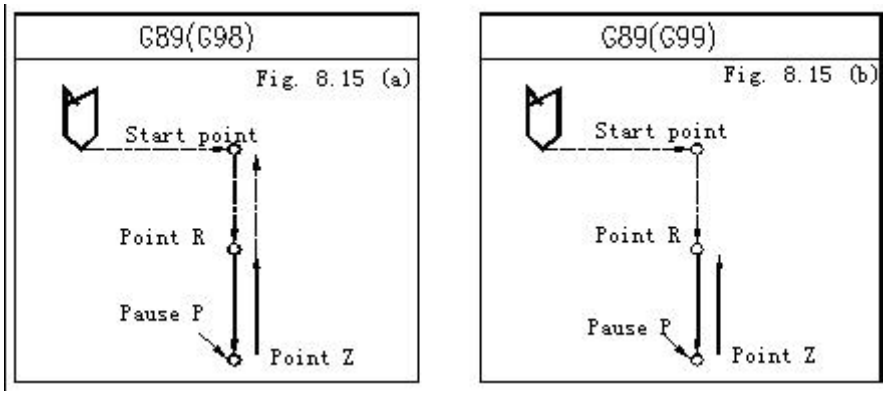
The fixed cycle G88 has manual return function and is used for boring.



Boring Cycle (G88) Diagram

2.4.12. Boring cycle (G89)

The fixed cycle G88 has manual return function and is used for boring.



Boring Cycle (G89) Diagram

Notes for using hole processing fixed cycle

a. During programming, it is necessary to use S and M code to rotate the spindle before the fixed cycle instruction.

```
M03    ; spindle positive rotation
.
G□□……;correct
..
M05; spindle stop
G□□……; false (instruction M03 or M04 is required before this segment )
```

b. In fixed cycle mode, the segment containing X, Y, Z, R will execute the fixed cycle; if a segment contains neither address above, this segment won't execute the fixed cycle, except address X in G04. In addition, address P in G04 won't change the P value in hole processing parameter.

```
G00 X_;
G81 X_ Y_ Z_ R_ F_ K_ ;
;(do not execute hole processing,)
F_; (do not execute hole processing, F value is updated)
M_; (do not execute hole processing, only executes auxiliary function)
G04 P_; (do not execute hole processing, use G04 P_ to change hole
processing data P)
```

c. Hole processing parameter Q, P must be specified in the segment in which the fixed cycle is executed, or else the specified Q, P value are invalid.

d. During executing the fixed cycle (e.g. G76, G84, etc.) that contains spindle control, the spindle hasn't reached the specified rotation when the tool starts cutting feeding. In this case, it is required to insert G04 pause instruction during the hole processing operation.

e. G code of group 01 also has the effect to cancel fixed cycle, and thus do not write fixed cycle instruction and G code of group 01 in the same segment.

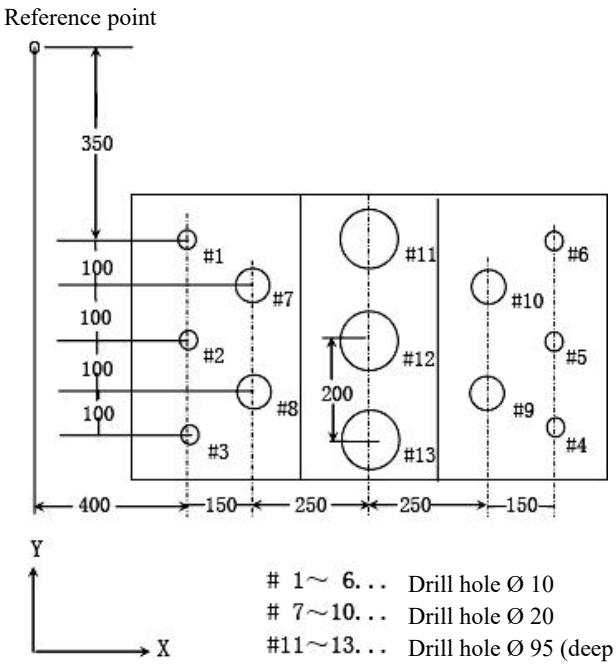
f. If the segment that executes the fixed cycle specifies an M code, the M code will be executed while the fixed cycle is positioning, and the signal that M instruction is executed is sent when Z axis returns to point R or the start point. If the fixed cycle is repeated with K parameter instruction, the M code in the same segment will be executed when the fixed cycle is first executed.

g. In fixed cycle mode, tool offset instructions G45~G48 will be ignored (won't be executed).

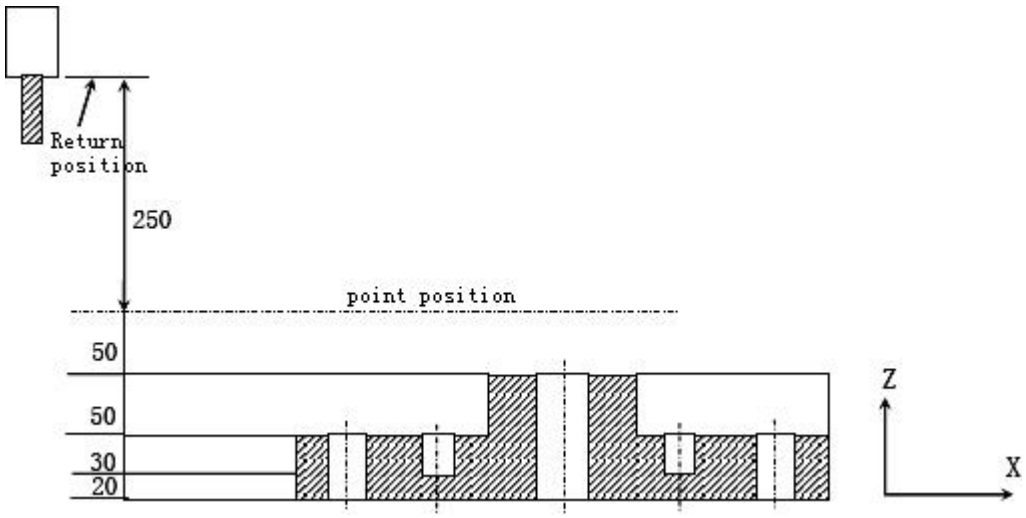
h. When single segment switch is in up position, the fixed cycle will stop after executes X, Y axis positioning, quickly feeds to point R and returns from the hole bottom (to point R or the start point). That is to say, it is required to press the cycle start button for three times to complete a hole processing. During the three stops, the first two stops are in feeding state, and the last one is in stopped state.

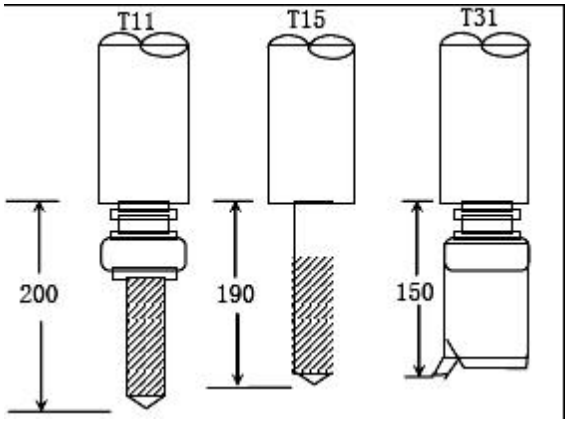
i. While executing G74 and G84 cycles, if you press the retain button when Z axis moves to point R to point Z or reverse, the feeding retaining indicator will be lighted immediately, but the machine tool action won't stop immediately, until Z axis returns to point R. In addition, feeding rate switch is invalid in G74 and G84 cycles, and it is fixed at 100%.

Examples of using tool length compensation and fixed cycle



1～ 6... Drill hole Ø 10
7～10... Drill hole Ø 20
#11～13... Drill hole Ø 95 (deep 50MM)





The value of offset No. 11 is 200.0,

The value of offset No. 15 is 190.0,

The value of offset No. 31 is 150.0,

The offsets are set separately. The program follows:

N001 G92 X0 Y0 Z0 ; the coordinate system is set at reference point
N002 G90 G00 Z250.0 T11 M6; change tool
N003 G43 Z0 H11 ; execute plane tool length compensation at the start point
N004 S30 M3 ;spindle starts
N005 G99 G81 X400.0 Y-350.0
Z-153.0 R-97.0 F120.0 ; process #1 hole after positioning
N006 Y-550.0 ; process #2 hole after positioning, return to point R plane

```
N007 G98 Y-750.0 ;              process #3 hole after positioning, return to start point plane
N008 G99 X1200.0 ;              process #4 hole after positioning, return to point R plane
N009 Y-550.0 ;                  process #5 hole after positioning, return to point R
plane
N010 G98 Y-350.0 ;              process #6 hole after positioning, return to start point plane
N011 G00 X0 Y0 M5 ;             return to reference point, spindle stops
N012 G49 Z250.0 T15 M6  ;       cancel tool length compensation, change tool
N013 G43 Z0 H15 ;               start point plane, tool length compensation
N014 S20 M3 ;spindle starts
N015 G99 G82 X550.0 Y-450.0 ;
Z-130.0 R-97.0 P30 F70;         process #7 hole after positioning, return to point R plane
N016 G98 Y-650.0 ;              process #8 hole after positioning, return to start point plane
N017 G99 X1050.0 ;              process #9 hole after positioning, return to point R plane
N018 G98 Y-450.0 ;              process #10 hole after positioning, return to start point plane
N019 G00 X0 Y0 M5 ;             return to reference point, spindle stops
N020 G49 Z250.0 T31 M6 ;        cancel tool length compensation, change tool
N021 G43 Z0 H31 ;               start point plane, tool length compensation
N022 S10 M3 ;spindle starts
N023 G85 G99 X800.0 Y-350.0 ;
Z-153.0 R47.0 F50 ;             process #11 hole after positioning, return to point R plane
N024 G91 Y-200.0 ;              process #12, #13 holes after positioning, return to point R plane
Y-200.0 ;
N025 G00 G90 X0 Y0 M5 ; return to reference point, spindle stops
N026 G49 Z0 ;                   cancel tool length compensation
N027 M30 ;%                     program stops
```

2.5 Conversion of G command

2.5.1. Program coordinates rotation G68 and G69

Format:

(1)Coordinate rotation function starts

G68 G17 X_ Y_ R_; Coordinate rotation starts

G68 : Coordinate rotation command

X_ Y_: Rotation center coordinates

Specify the 2 axes corresponding to the selected plane in X, Y, and Z in absolute position

G17 specifies X_ Y_, G18 specifies Z_ X_, G19 specifies Y_ Z_

R : Rotation angle, counterclockwise is +

(Note: This command needs to select the command plane in advance through G17~G19)

(2)Coordinate rotation cancelled

G69; Coordinate rotation cancelled

Detailed instructions:

(1) The rotation center coordinates (x1, y1) are always specified in absolute values. It is not treated as an incremental value even if it is specified with an incremental address.

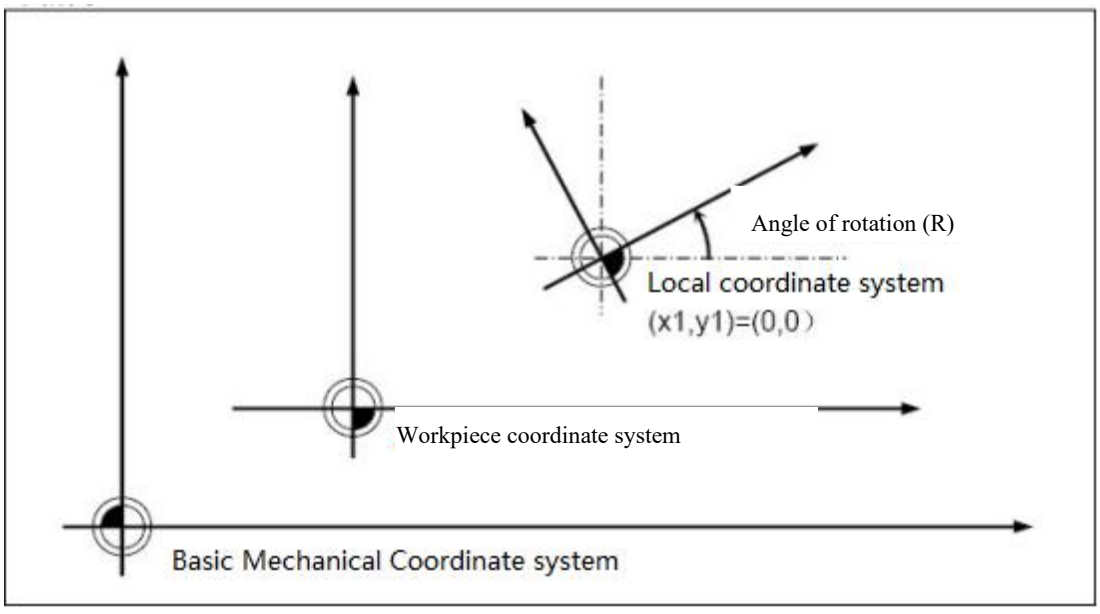
(2) If the G68 command rotation center coordinate (x1, y1) is omitted, the position of the G68 command will become the center of rotation.

(3) Rotate the angle specified by the rotation angle r1 counterclockwise.

(4) The setting range of the rotation angle r1 is -360.000 to 360.000. When a command of a degree exceeding 360 degrees is issued, the command will be the remainder after being divided by 360 degrees.

(5) The rotation angle r1 is modal data and does not change until the new angle is specified next time. Therefore, the command of the rotation angle r1 can be omitted. If the rotation angle is omitted when the G68 command is issued for the first time, r1 will be regarded as "0".

(6) The program coordinate rotation is a function on the local coordinate system. Therefore, the relationship between the rotated coordinate system, the workpiece coordinate system, and the basic machine coordinate is as shown in the figure below.



(7) The coordinate rotation command in the coordinate rotation is treated as a change in the center coordinate

and the angle of rotation.

(8) When the M02 and M30 command is issued or the reset signal is input in the coordinate rotation mode, the coordinate rotation will enter the cancel mode.

(9) In the coordinate rotation mode, G68 is displayed on the modal information screen, and G69 is displayed after the mode is canceled. (The angle of rotation command R does not display the modal value.)

(10) The program coordinate rotation function is only valid in the auto operation mode.

Program example

N10 G92 X0 Y0

N20 G68 X10 Y10 R-30

N30 G90 G42 G00 X10 Y10 F100 H01

N40 G91 X20

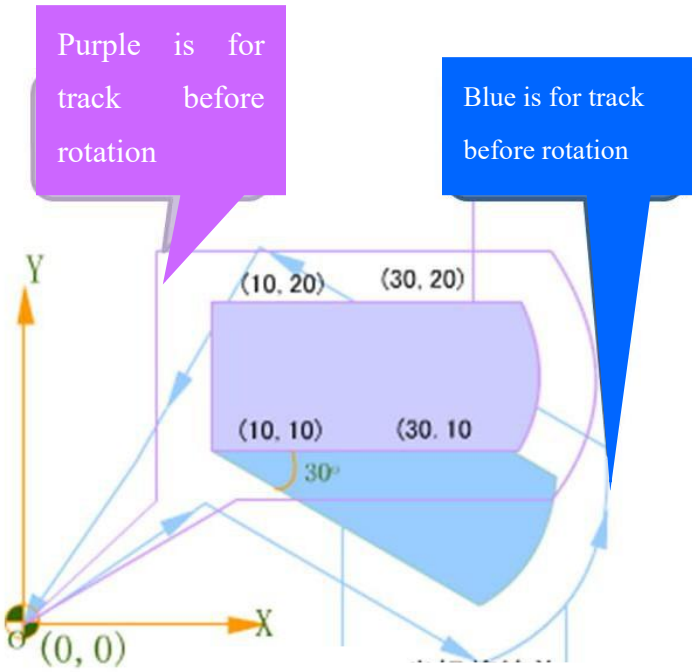
N50 G03 Y10 I-10 J 5

N60 G01 X-20

N70 Y-10

N80 G40 G90 X0 Y0

N90 G69 M30



The shape after coordinate system rotation

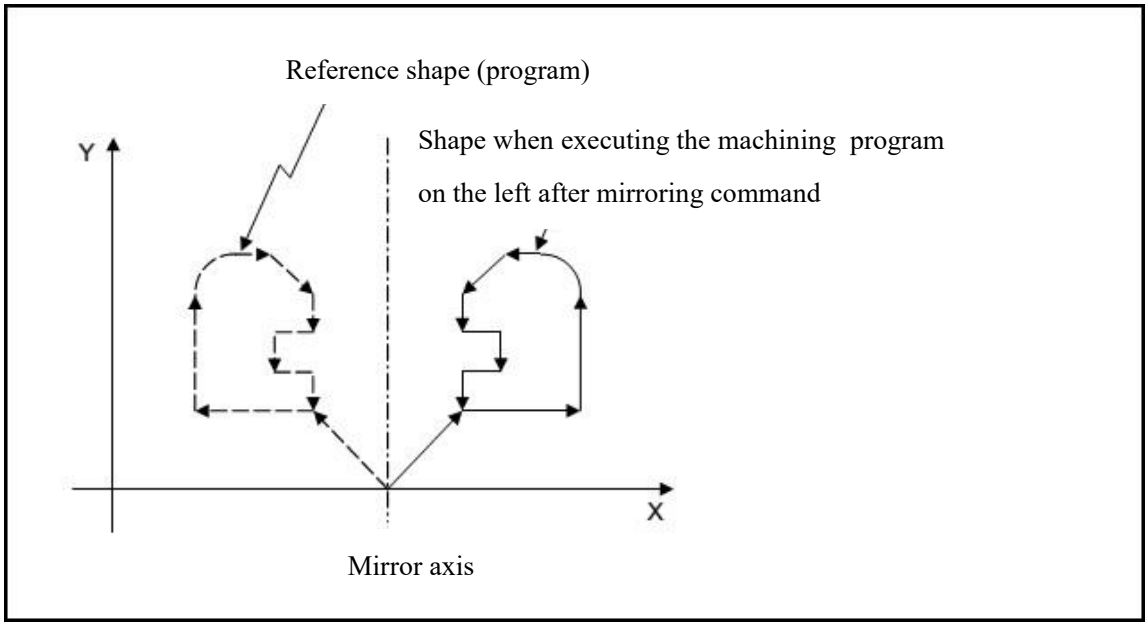
Tool path after coordinate system rotation and radius compensation

2.5.2. G51.1and G50.1 mirroring

Functions and purpose:

When cutting the left and right symmetrical patterns and shapes, the other side can be processed by only programming the left side or the right side, thereby saving the time required for the programming. At this point, the most effective feature is the mirroring feature.

For example, as shown in the following figure, when there is a program for machining the left side shape, the shape symmetrical to the left side can be completed on the right side by performing mirroring on the program.



Format:

G51.1 X_ Y_ Z; Format 1

G50.1 X_ Y_ Z; Format 2

G50.1; Format 3

Operating parameter descriptions

Function 1: Mirror function starts

G51.1 X_ Y_ Z_ Mirror function starts and points to the absolute coordinate position of the mirror axis

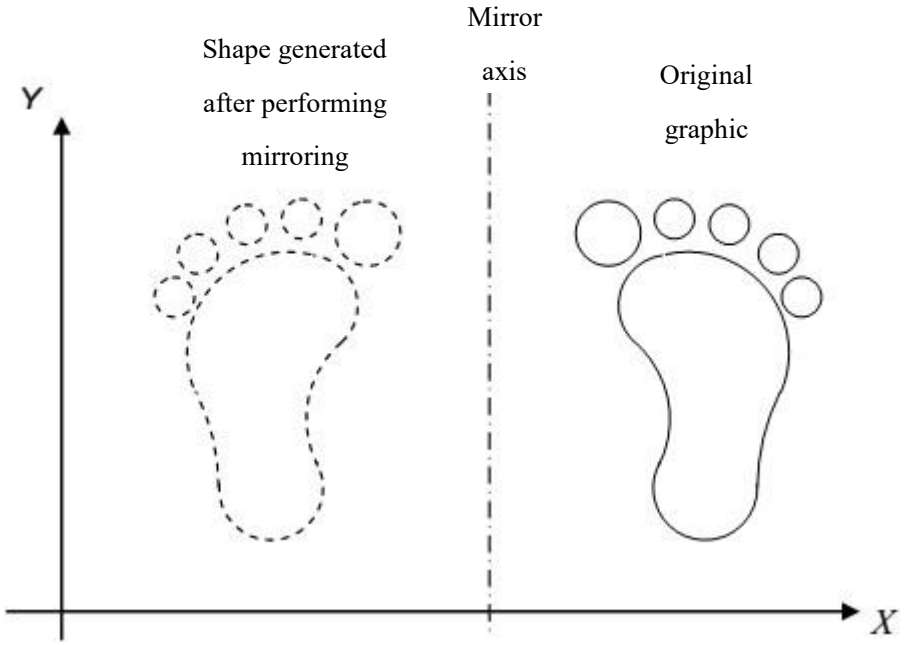
G50.1 X_ Y_ Z_ Cancels the mirror function of the specified axis, wherein the independent variable is arbitrary and meaningless.

G50.1 Cancel the mirror function of all axes.

(Note: This command needs to select the command plane in advance through G17~G19)

Action descriptions:

When cutting a symmetrical shape, you only need to select the machining program of one side, and through the mirror function, you can machine the symmetrical shape on the other side. As shown in the figure below: After the machining program for cutting the right sole, cut the same machining program using the mirror function, you can then produce the left sole.



Detailed instructions of commands:

(1) The mirror function command can specify the absolute coordinate position of the mirror axis according to the increment/absolute command.

(2) The mirror function command must be selected in correspondence to the plane, then its specified mirror axis would work. For example: G17 plane, only the X- and Y-axis mirror axes have an effect, and the Z-axis mirror axis has no effect.

(3) When the mirror function is enabled, the operation image when the intermediate point is reached is still valid when G28 is executed, but the process of resetting the origin via the intermediate point is invalid.

(4) When the mirror function is enabled, the process of returning to the intermediate point from the origin is invalid when G29 is executed, but the process from the intermediate point to the destination is valid.

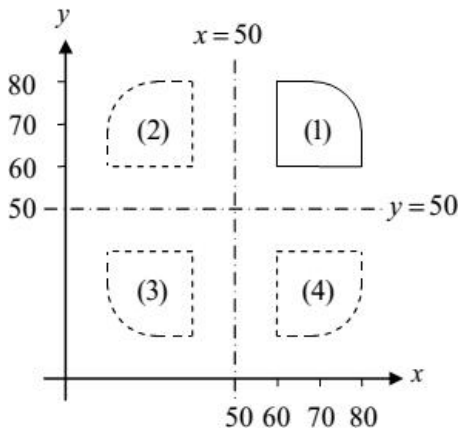
(5) The mirror axis changes position as the coordinate system shifts and the tool length is corrected.

(6) G53 command mirror is invalid. .

(7) RESET will cancel the mirror function when encountering M30, but the mirror function will not be cancelled when encountering M99 and M02.

Program example

The following is an example of a program for mirror function instructions, as shown in figure below:



Main program

O0003

G90 G54 G17 F1000

G00 X0 Y0

M98 P0002

G51.1 X50 Mirror X -axis absolute coordinate position 50

M98 P0002

G51.1 X50 Y50 Mirror X- and Y-axis absolute coordinate position 50

M98 P0002

G50.1 X0 Cancel mirroring of X-axis

M98 P0002

G50.1 Cancel mirroring of all axes

G00 X0 Y0

M30

%

Subprogram

O0002

G00 G90 X60 Y60

G01 X80

G01 Y70

G03 X70 Y80 R10

G01 X60

G01 Y60

M99

%

2.6 Probe G command

2.6.1. G31.1

Command for external signal detection jump

Format: G31.1 X_(Y_ Z_ A_ B_ C_) P_ F_

X(Y_ Z_ A_ B_ C_): When searching the distance, relative quantity is provided with a symbol. Search in the positive direction when it is a positive number, and search in the negative direction when it is a negative number.

P: The external signal input point, the signal is valid if it is connected with 0V, and stops when a signal deceleration is detected. Reduce the search speed if the precision position is required. .

F: Search speed

2.6.2. G31.2

Extend G31.2 command to search for tool setting function

Format: G31.2 X_(Y_ Z_ A_ B_ C_) F_ P_ Q_ L_

Z: Search for the position of tool regulator Z

F: Search for the speed of the tool regulator

P: Tool setting input point for tool regulator

Q: Limiting input point of the tool regulator

L: Effective level of the tool setting point and limit point

2.6.3. G31.3

Extend the G31.3 command, to read the distance between the two detection switches

Format: G31.3 X_(Y_ Z_) P_ Q_ L_ D_

Command introduction: X: 0: move in negative direction 1: move in positive direction

P: X-axis, corresponding detection switch input port

Q: Y-axis, corresponding detection switch input port

L: Detection port input level

D: spindle encoder input detection axis number, 3: indicates that the Z-axis encoder input is detected, and the main encoder line number and gear ratio are set according to the spindle parameters;

H: The minimum deviation value

Deviation save parameters, saved in #100

2.7 Machine coordinate positioning commands

2.7.1. G53

A command used to position each axis to the machine coordinate

Format: G53 X_(Y_ Z_ A_ B_ C_)

X(Y_ Z_ A_ B_ C_): Machine coordinate position.

Note: The speed uses the axis rapid traverse parameters.

2.7.2. G53.1

A command used to position the interpolation to machine coordinate

Format: G53.1 X_(Y_ Z_ A_ B_ C_) F_

X(Y_ Z_ A_ B_ C_): Machine coordinate position.

F: Interpolation positioning speed

3. Auxiliaryfunction

This machine tool uses S code to program spindle rotation, uses T code to program tool selection, and other auxiliary functions are achieved with M code.

3.1 M code list

Table 11.1M code list

M code	Function
M00	Program stop
M03	Spindle positive rotation
M04	Spindle negative rotation
M05	Spindle stop
M06	Tool exchange instruction
M08	Cooling on
M09	Cooling off
M32	Lubrication on
M33	Lubrication off
M30	Program ends and returns to program header
M98	Call subroutine
M99	Subroutine ends and returns/ Repeat
M56	Output 02 port is in high voltage level
M57	Output 02 port is in low voltage level
M58	Output 03 port is in high voltage level
M59	Output 03 port is in low voltage level
M10	Output 06 port is in high voltage level
M11	Output 06 port is in low voltage level
M20	Output 07 port is in high voltage level
M21	Output 07 port is in low voltage level
M12	Output 08 port is in high voltage level
M13	Output 08 port is in low voltage level
M14	Output 09 port is in high voltage level
M15	Output 09 port is in low voltage level

M code	Function
M16	Output 10 port is in high voltage level
M17	Output 10 port is in low voltage level
M18	Output 11 port is in high voltage level
M19	Output 11 port is in low voltage level
M40	Output 12 port is in high voltage level
M41	Output 12 port is in low voltage level
M42	Output 13 port is in high voltage level
M43	Output 13 port is in low voltage level
M44	Output 14 port is in high voltage level
M45	Output 14 port is in low voltage level
M46	Output 15 port is in high voltage level
M47	Output 15 port is in low voltage level
M48	Output 16 port is in high voltage level
M49	Output 16 port is in low voltage level
M50	Output 17 port is in high voltage level
M51	Output 17 port is in low voltage level
M66	Output 20 port is in high voltage level
M67	Output 20 port is in low voltage level
M64	Output 21 port is in high voltage level
M65	Output 21 port is in low voltage level
M62	Output 22 port is in high voltage level
M63	Output 22 port is in low voltage level
M60	Output 23 port is in high voltage level
M61	Output 23 port is in low voltage level
M88 Pn Lm	Check whether input IO (IN n) voltage level is m (high/low), continue to wait if not true
M89 Pn Lm Qt	Output: OUT n, voltage level: m, delay t ms output, or execute immediately if there is no t

In the machine tool, M code has two effects: one is to control the execution of the program, and the other is IO operation, which is used to control the execution of spindle, cooling system and other auxiliary devices.

M code for program control

M00……program stops. When NC executes M00, the program execution is interrupted; after reset, press the Start button to continue executing the program.

M30……program ends, and returns to program header

M98……call subroutine

M99……subroutine ends, and returns to the main program

Other M codes

M03……spindle positive rotation. Use this instruction to rotate the spindle counterclockwise (CCW) with currently specified spindle rotation.

M04……spindle reverse rotation. Use this instruction to rotate the spindle clockwise (CW) with currently specified spindle rotation.

M05……spindle stops.

M06……tool change starts. M06 T02 instruction is to change tool #2.

M08……cooling on

M09……cooling off

M32……lubrication on

M33……lubrication off

M88……specify input IO port to check the voltage level; continue to execute if the levels are same, or else wait. If no voltage level signal is specified, it is low voltage level signal by default. For example: M88 P0 L1, wait until IN0 is high voltage level, or else wait all along.

M89……specify output IO port to check the voltage level; if no voltage level signal is specified, it is low voltage level signal by default; if Q value is specified, the operation needs to delay for Q ms and then output IO signal.

For example: M89 P5 L0, specify OUT5 to output low voltage level.

Note:

If the motion instruction and M are in the same segment, M instruction will be executed first.

If the program has several M codes in current line, only one is valid, i.e. the last defined M code.

3.1.1. M00 program pause

When the system is running automatically, it reads M00, the program pauses; press the start button again to continue runing from the stop point. Press the reset button, the system running status will be reset, and the system stops running.

3.1.2. M03 spindle moves clockwise (CW)

When M03 is read at the time the system is running automatically, the spindle starts rotate clockwise and the spindle moves at the speed set by the S value.

3.1.3. M04 spindle moves counterclockwise (CCW)

When M04 is read at the time the system is running automatically, the spindle starts to counterclockwise and the spindle moves at the speed set by the S value.

3.1.4. M05 spindle stops

When M05 is read at the time the system is running automatically, the spindle is stopped for automatically running.

3.2 M command for input signal detection output

3.2.1. M88 input port signal detection

Format: M88 Pn Lm

Check if the input IO (IN n) level signal is m (high or low), if not, continue to wait

Example: Wait for the input terminal is IN10, and the port level is low at "0", continue to wait until the level is low. When the level is 0, the command ends and the next command is executed.

Example:

M88 P10 L0

G0 X10 Y10

3.2.2. M89 specifies output port control

MFormat: 89 Pn Lm Qt

Output OUT n, level is m, delay t millisecond output, and execute immediately without t

Example: Open the output port OUT02 and specify the status as ON (‘1’ is on, '0’ is off), and it takes effect immediately.

Example:

M89 P02 L1

G0 X10 Y10

3.3 Spindle speed function S

The rotation instruction of the spindle is specified by the S code, which is modular, i.e. always valid after the rotation is specified, until another S code changes the modular value.

The maximum value of the S command is limited by the value set by the spindle parameter "Maximum spindle speed".

There're two output modes for S command, and the spindle parameter "Spindle mapping axis" has the following effects:

- Spindle mapping axis: Analog spindle
- Mode of inverter analog quantity: according to the ratio of S value and spindle parameter "maximum spindle speed" multiplied by 10V, the analog voltage value that needs to be output is then obtained. S command needs to execute M03 or M04 to make analog output upon the designation;
- Spindle mapping axis: To specify one feed axis in the axis group

  It means the current spindle is the AB phase pulse control mode. At this time, the S value is set by the spindle encoder and then determine the pulse frequency.

3.4 Tool function

Machine tool magazine uses random tool selection mode, i.e. two digits T code TXX specifies the tool No., regardless which tool set it is in; the range of address T is any integral between 1 and 99.

Waring:

Tool meter must be set properly, or else it will damage the machine tool and cause unpredictable results.

4. Category B macro function

4.1 Variable instruction

Function:

All the address values in the program are not described with fixed value, and are replaced with variables; when the program is running, variables are referenced to improve the versatility of the program. This function is called as variable instruction.

Format:

#ΔΔΔ=○○○○○○○○○ or #ΔΔΔ=[expression]

Details:

(1) Representation of variables:

(a) # m	M=0~9 constituted value	#100
(b) # [f]……	f has the following meanings	
	Value m	123
	Variable	#543
	Expression	#110+#119
	- (symbol) expression	-#120
	Function expression	SIN [#110]

Note:

Standard operating symbols are +, -, ×, /.

When the function expression is ignored, the function can't be executed.

The variable No. can't be negative, e.g. # -100 is illegal.

Below are false variable representations:

False		Correct
#6/2	→	#[6/2]
#－［#1］	→	#[－#1]

#——5	→	#[－[－5]]

(2) Types of variables

Type	Variable	Function description
Global variable	#100~#199 #500~#999	Both main program and subroutine can be called #100~#199 are non-retentive variables, and will be reset automatically when the system is repowered #500~#999 are retentive variables, and the values still exist when the system power system is cut off.
Local variable	#1~#32	Can be called in the same program
System variable	No	

(3) Variable reference

(a) Except O, N and / (slash)

(b) Specify with variables directly

G01X#1Y#100

(c) Take the complement of the variables directly

G01X-#2

(d) Variable defines variable

#3=-#105 ; take the complement of #105 directly and evaluate to #3

#4=1000 ; evaluate 1000 to #4 directly

(e) Define the evaluation with expression

#1=#3+#2-100; the value #1 equals to the result of #3+#2-100

X[#1+#3+1000]; the value of X is the result of expression [#1+#3+1000]

☞ **Note:**

Function evaluation and expression evaluation must be written separately, and can't be in the same line.

False		Correct
X#1＝#3＋100	→	#1＝#3＋100 X#1

[] can be embedded up to five levels.

```
#543＝－[[[[[#120]/2＋15.]*3-#100]/ #520+#125+#128]* #130+#132
```

The variable values must be 0~±9999999 (seven significant figures); if exceeding the maximum value, the calculation error will be enlarged.

4.2 Macro program call

Using macro calling function

Function

Same as subroutine calling, the macro program can transfer variables to subroutine during calling, which is different from M98 subroutine calling.

The following G codes are instructions to call macro program:

Table 12.1 Macro Program Calling Instruction

G code	Function
G65	Macro program calling
G66	Macro program calling mode A (call motion instruction)
G661	Macro program calling mode B (call every segment)
G67	Cancel macro program calling mode

Details:

The macro programs specified after G66 (or G661) instruction is specified, before G67 (cancel) instruction, and after the segments with motion instruction are executed (or every segment is executed).

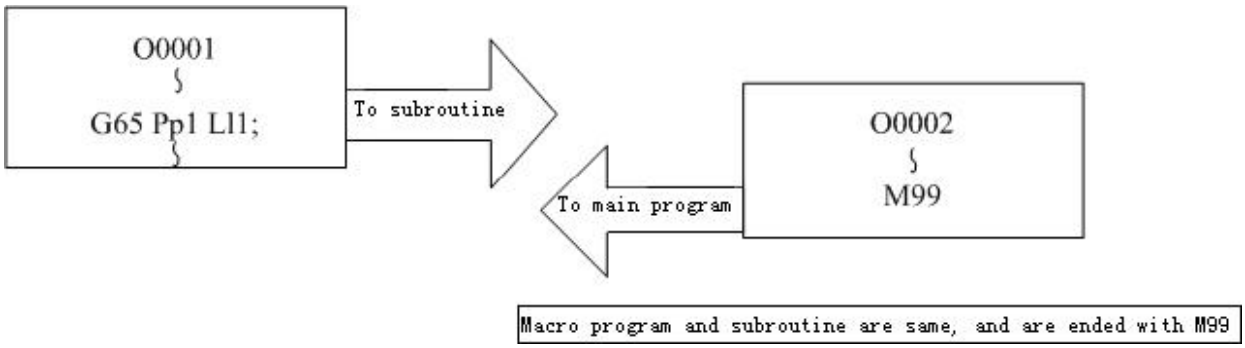
G66 (or G661) and G67 instructions must be used in pair in the same program.

4.3 Macro program calling command

Function and purpose:

Macro program calling instructions include simply callings that are called by calling instruction only, calling modes (A&B) of single segment fixed calling.

(1) Simply calling



Format:

G65 P_ L_ <argument>;	
P_	: subroutine No.
L_	: repeat times

The <argument> function in G65 is a method that the main program uses bit address to transfer parameters to subroutine; this method uses local variable to transfer; the argument is described below.

Argument format:

A_B_C_...X_Y_Z_

Details:

Except G, L, N, O, P, all bit addresses can be specified as arguments.

The bit addresses that do not need to transfer can be ignored.

In G65 instruction segment, all the bit addresses are considered as the arguments of G65.

For example:

G65P0002N100G01G90X100.Y200.F400R1000,

G01 instruction isn't executed, and all bit addresses are considered as the arguments of G65.

The comparison between the bit addresses specified by the arguments and local variable number follows:

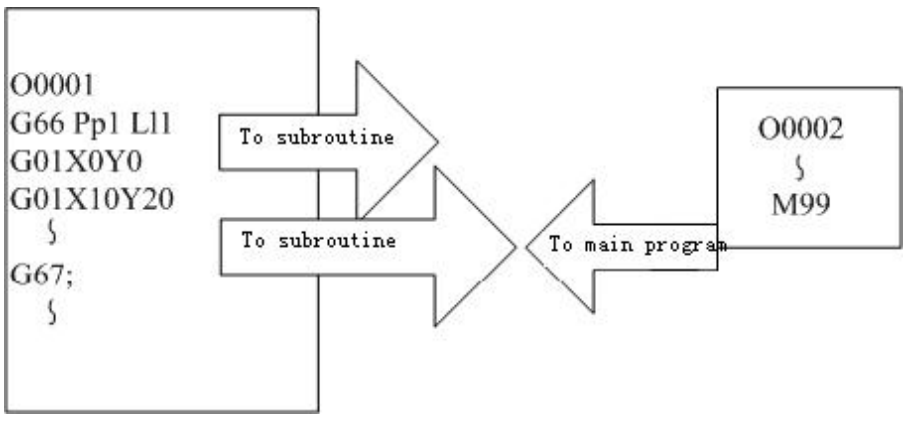
Table 12.2 Comparison between Argument Specified Bit Addresses and Local Variables

Address	Variable No.	G65, G66, G661
A	#1	○
B	#2	○
C	#3	○
D	#7	○
E	#8	○

F	#9	○
G	×	×
H	#11	○
I	#4	○
J	#5	○
K	#6	○
L	×	×
M	#13	○
N	×	×
O	×	×
P	×	×
Q	#17	○
R	#18	○
S	#19	○
T	#20	○
U	#21	○
V	#22	○
W	#23	○
X	#24	○
Y	#25	○
Z	#26	○

○: can be used ×: can't be used

(2) Mode calling A (motion instruction calling)



Between G66 and G67, after the segment with motion instruction is executed, all the specified macro subroutines are called and executed, and the execution times are specified by L.

Format:

G66 P_ L_ <argument>;	
P_	:subroutine No.
L_	:repeat times

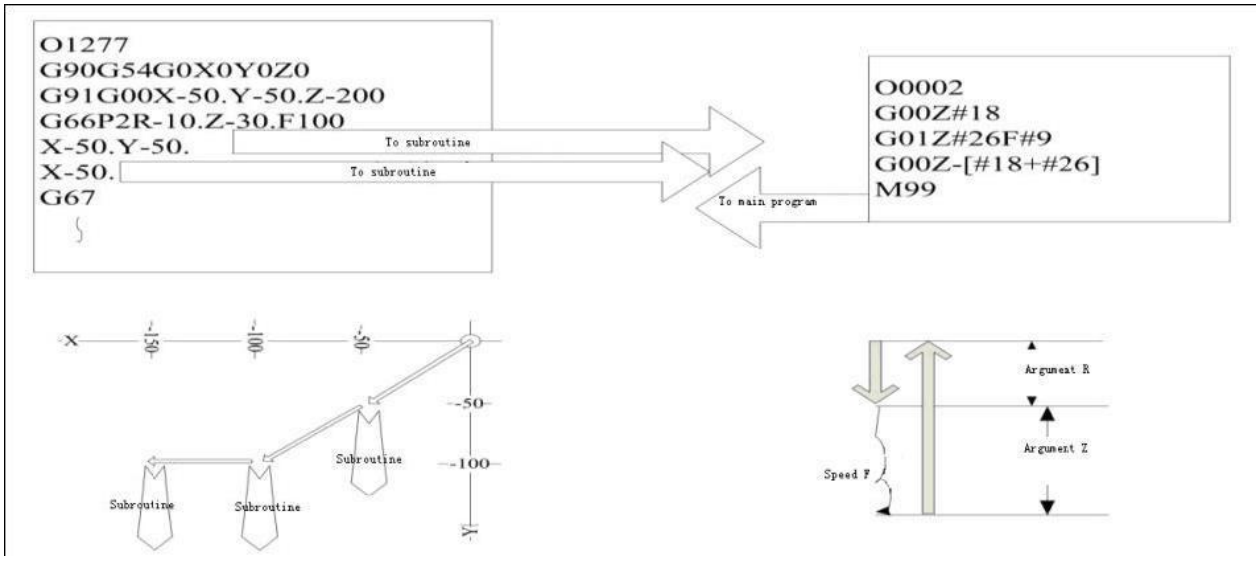
Details:

After G66 instruction is specified and before G67 (cancel) instruction is specified, all the segments with motion instruction will call G66 specified macro subroutine automatically after executed.

G66 and G67 instructions are in the same program, and must be specified in pair. If G66 instruction isn't executed first and G67 instruction is executed directly, the system will alarm.

In G66 instruction segment, all the bit addresses are considered as the arguments of G65.

For example: drilling cycle



Note:

G66 instruction executes the subroutine for the first time, and later motion instructions will call the subroutine automatically.

After G67 instruction takes effect, the subroutine won't be executed.

(3) Mode calling B (every segment calls)

Between G661 and G67, every instruction segment will call the specified macro subroutine unconditionally.

Format:

G661 P_ L_ <argument>;
P_ :subroutine
L_ : repeat times

Details:

In G661 mode, all the read codes except O, N and G codes of every segment will be used as arguments.

In G661 instruction segment, all the bit addresses are considered as the arguments of G661.

For example

G661P0002N100G01G90X100.Y200.F400R1000,

G01 instruction isn't executed, and all bit addresses are considered as the arguments of G661.

4.4 Variable

Function and purpose:

Variable is a useful function of macro. Four types of variables are available, which are local variable, global non-retentive variable, global retentive variable and system variable. These variables make the writing of macro very convenient and universal.

Using multiple variables:

- Macro calls variable, and the variable can be specified by multiple or expression. As below:

#1＝10	
#10＝20	According to #1=10,#[#[#1]]=#[#10]
#20＝30	According to #10=20,#[#10]=#20
#5＝#[#[#1]];	Therefore, #5=#20 or #5=30
#10=5	
#10=20	According to #1=10,#[#[#1]]=#[#10]
#20=30	According to #10=20,#[#10]=#20
#5=1000	Therefore, #20=#5 or #20=1000
#[#[#1]]=#5	

- Example of specifying multiple variables:

#10=5 #5=100 #6=##10	##10 and #[#10]have the same meaning

- Replace the number with expression:

#10=5	
#[#10+1]=1000	#6=1000
#[#10-1]=-1000	#4=-1000
#[10*3]=100	#15=100
#[#10/2]=-100	#2=-100

Undefined variables:

The variables haven't been defined after the system is started are blank by default. The local variables that the arguments haven't been specified are also used as blank variables. The #0 of the system is also blank variable. In the calculation, blank variables can be used as 0; generally, #0 can't be used as expression L-value for calculation. However, if the programmers edit falsely, the program won't report error and this measure doesn't have any effect.

- Calculation formula

#1=#0;.....#1=<Null> #2=#0+1; ……#2=1 #3=1+#0; ……#3=1 #4=#0*10;……#4=0 #5=#0+#0;……#5=0	Please note that the <blank> in the calculation formula indicates 0. <blank> + <blank> = 0; <blank> + <fixed number> = <fixed number> <fixed number> + <blank> = <fixed number>

- Variable reference

#1=<blank> G0X#1Y1000;.....equals to G0X0Y1000 G0X#1+10Y1000;.....equals to G0X10Y1000

- Conditional
- In conditional determination, blank variable is equivalent to 0 in logic conditional operator.

4.5 Types of variables

(1) Public variables

Any bit address can use public variables, which contain 600 groups; among those, #100~#199 are non-retentive public variables after power failure, #500~#999 are retentive public variables.

(2) Local variables (#1-#32)

When calling subroutine, local variables can be defined with <argument> and only can be used in programs; the local variable of every macro program is independent, and thus can be repeated. (up to four levels)

G65 Pp1 Ll1 <argument>;
p1 : subroutine No.
l1 : repeat times

<Arguments> are Aa1 Bb1 Cc1… Zz1, etc.; the bit address specified by <argument> and the local variables in the subroutine are shown below:

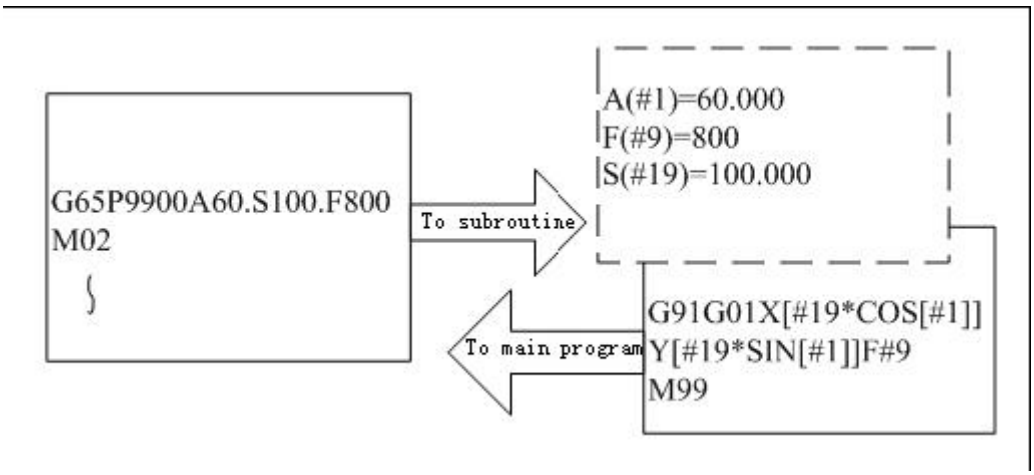
Bit address	Variable No.	Subroutine	Bit address	Variable No.	Subroutine
A	#1	○	N	×	×
B	#2	○	O	×	×
C	#3	○	P	×	×
D	#7	○	Q	#17	○
E	#8	○	R	#18	○
F	#9	○	S	#19	○
G	×	×	T	#20	○
H	#11	○	U	#21	○
I	#4	○	V	#22	○
J	#5	○	W	#23	○
K	#6	○	X	#24	○
L	×	×	Y	#25	○
M	#13	○	Z	#26	○

The argument bit addresses marked with "×" can't be used.

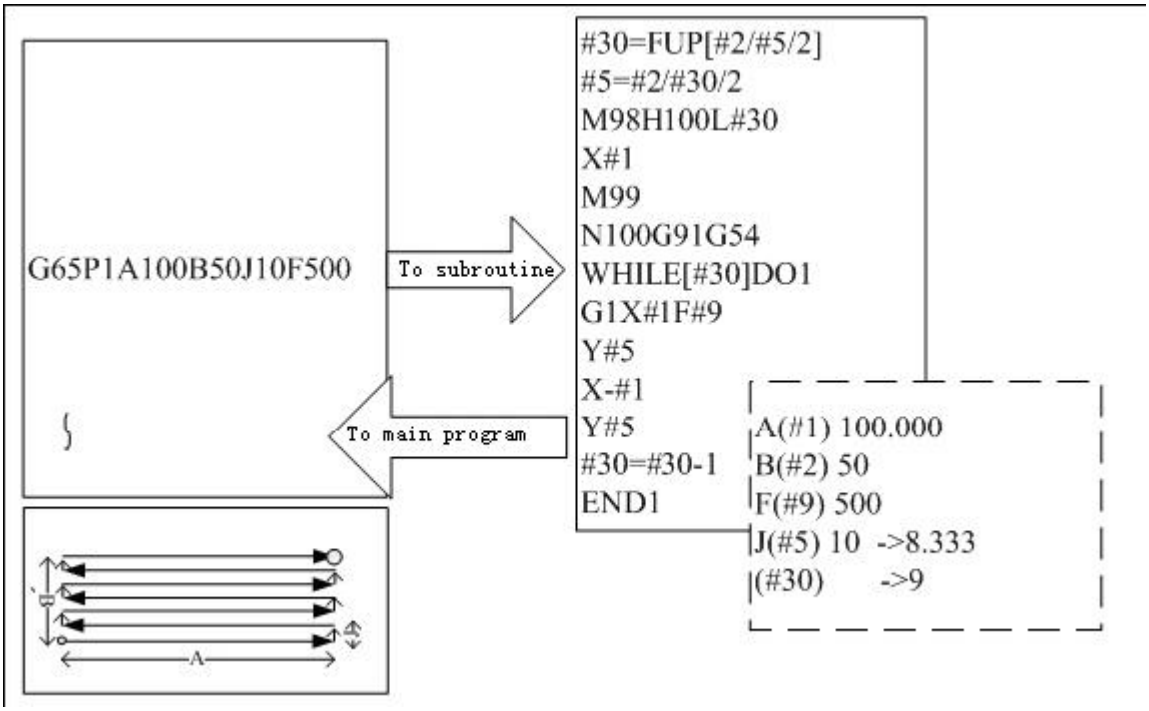
The argument bit addresses marked with "○" can be used.

① While calling macro program, the local variables in subroutine can be defined by specifying the

<argument>



② Local variables can be used in respective subroutine freely.

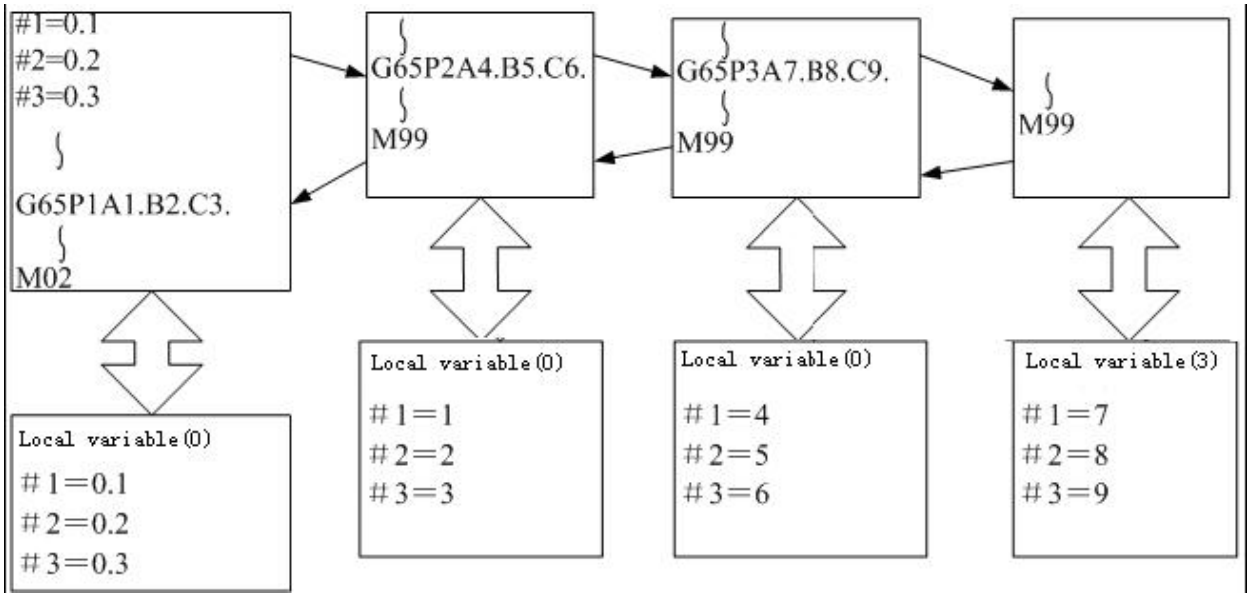


In face milling examples, argument J indicates that the spacing is 10mm during face milling; however, to ensure equal spacing processing, the spacing is changed to 8.333mm.

Secondly, local variable #30 is the calculation result of reciprocating processing times data.

③ Local variables can be used for macro calling of every level independently up to four levels.

The main program (macro level 0) provides specific local variables; however, local variables can't use argument at level 0.



4.6 Calculus instruction

The variables allow various calculus expressions.

Format:

#i＝[expression] The variables allow various calculus expressions.

In the table below, #j, #k can be replaced with constants.

Calculation method	#i=#j	Definition/replacement
Addition and subtraction	#i=#j+#k	Addition
	#i=#j-#k	Subtraction
	#i=#j OR #k or #i=#j|#k	32-bit OR calculation (logical AND)
	#i=#j XOR #k or #i=#j^#k	32-bit XOR calculation
Multiplication and division	#i=#j*#k	Multiplication
	#i=#j/#k	Division
	#i=#j MOD #k	Remainder
	#i=#j AND #k or #i=#j & #k	32-bit AND calculation (logical product)
Function calculation	#i=SIN[#k]	Sine
	#i=COS[#k]	Cosine
	#i=TAN[#k]	Tangent tanθ equals to sinθ/cosθ
	#i=ASIN[#k]	Arcsine
	#i=ATAN[#k]	Arctangent

	#i=ACOS[#k]	Arc cosine
	#i=SQRT[#k]	Square root
	#i=ABS[#k]	Absolute value
	#i=ROUND[#k]	Rounding
	#i=FIX[#k]	Abandon the decimal point
	#i=FUP[#k]	Carry the decimal point
	#i=LN[#k]	Natural logarithm
	#i=EXP[#k]	e(=2.718…) is exponent of the base

☞ **Note:**

The values without decimal point are considered same as the values with decimal point (1=1.000)

The expression after the function must be bracketed with [ ].

Expression calculation priority:

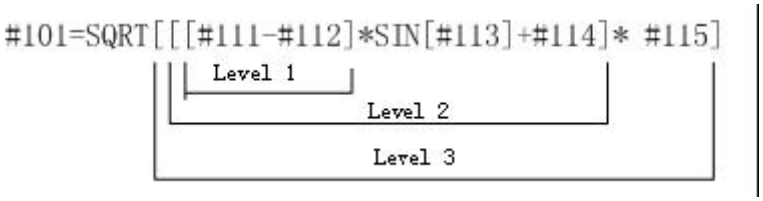
Smaller number indicates higher priority	Calculation symbol
1	#
2	[ ]
3	Function (SIN,COS,EXP…)
4	*,/,MOD
5	+,-
6	GE,GT,LE,LT
7	EQ,NE
8	AND,XOR,OR
9	=

☞ **Note:**

The calculation expression of the same level follows the sequence from left to right.

The calculation expression has more priorities; if the expression is too long, please enforce the priority with [ ].

[ ] can be embedded in the calculation for up to five levels. As below:



✍ **Example of calculation commands:**

(1) Specifying main program and argument	#i=#j	Definition/replacement
(2) Definition/replacement (=)	#1=1000 #2=1000 #3=#101 #4=#102 #5=#41	#1 1000.000 #2 1000.000 #3 100.000 #4 200.000 #5 -10.000
(3) Addition and subtraction (+ -)	#11=#1+1000 #12=#2-50 #13-#101+#1 #14=#41-3 #15=#41+#102	#11 2000.000 #12 950.000 #13 1100.000 #14 -13.000 #15 190.000
(4) Logical AND (OR)	#3=100 #4=#3 XOR 14	#3=01100100 14=00001110 #4=01101110=110
(5) XOR (XOR)	#3 = 100 #4 = #3 XOR 14	#3=01100100 14=00001110 #4=01101010=106
(6) Multiplication and division (* /)	#21=100*100 #22=100.*100 #23=100*100. #24=100.*100 #25=100/100 #26=100./100. #27=100/100. #28=100./100. #29=#41*#101 #30=#41/#102	#21 10000.000 #22 10000.000 #23 10000.000 #24 10000.000 #25 1.000 #26 1.000 #27 1.000 #28 1.000 #29 -1000.000 #30 -0.050
(7) Remainder (MOD)	#31=#19 MOD #20	#19 48.000 #20 9.000 #31 3.000
(5) XOR (XOR)	#9 = 100 #10= #9 AND 15	#9 =01100100 15 =00001111

(1) Specifying main program and argument	#i=#j	Definition/replacement
		#10=00000100=4
(9)Sine (SIN)	#501=SIN[60] #502=1000*SIN[60]	#501 0.860 #502 866.025
(10)Cosine (COS)	#541=COS[45] #542=1000*COS[45.]	#541 0.707 #542 707.107
(11)Tangent (TAN)	#551=TAN[60] #552=1000*TAN[60]	#551 1.732 #552 1732.051
(12)Arcsine (ASIN)	#531=ASIN[100.500/201.] #532=ASIN[0.500] #533=ASIN[-0.500]	#531 30.000 #532 30.000 #533 -30.000
(13)Arc tangent (ATAN)	#561=ATAN[173205/100000] #562=ATAN[173205/100.] #563=ATAN[173.205/100000] #564=ATAN[173.205/100.] #565=ATAN[1.732]	#561 60.000 #562 60.000 #563 60.000 #564 60.000 #565 59.999
(14)Arc cosine (ACOS)	#521=ACOS[100./141.421] #522=ACOS[10/14.142] #523=ACOS[0.707]	#521 45.000 #522 44.999 #523 45.009
(15)Square (SQRT)	#571=SQRT[1000] #572=SQRT[10.*10.+20.*20] #573=SQRT[#14*#14+#15*#15]	#571 31.623 #572 22.361 #573 190.444
(16)Absolute Value (ABS)	#576=-1000 #577=ABS[#576] #3 = 70. #4=-50. #580=ABS[#4-#3]	#576 -1000.000 #577 1000.000 #580 120.000
(17)		
(18)Rounding (ROUND)	#21=ROUND[14/3] #22=ROUND[-14/3]	#21 5.000 #22 -5.000
(19)Abandon the	#21=FIX[14/3]	#21 4.000

(1) Specifying main program and argument	#i=#j	Definition/replacement
(FIX)	#22=FIX[-14/3]	#22 -4.000
(20)Carry the decimal point (FUP)	#21=FUP[14/3] #22=FUP[-14/3.]	#21 5.000 #22 -5.000
(21)Natural logarithm (LN)	#101=LN[5] #102=LN[0.5] #103=LN[-5]	#101 1.609 #102 -0.693 Erro
(22)Exponent (EXP)	#104=EXP[2] #105=EXP[1] #106=EXP[-2]	#104 7.389 #105 2.718 #106 0.135

Calculation precision

Macro variable contains seven significant figures, and thus the precision may be reduced if single calculation value is too large or too small (9999999.000~0.0000001), and repeated calculation will cause cumulative error. Therefore, the macro variable should be in a reasonable range; in addition, while calculating trigonometric and exponential functions, too large value is also a reason of doubled error due to calculation error of the functions.

4.7 Control instruction

Conditional instruction

Format:

IF [conditional expression] GOTO n; (n is the order No. in the program)

The types of [conditional expression] are shown in the table below:

#i EQ #j	= when #iequals to #j
#i NE #j	≠ when #idoesn't equal to#j
#i GT #j	>when #i is larger than #j
#i LT #j	<when #iis smaller than #j
#i GE #j	≥ when #iis larger than or equals to #j
#i LE #j	≤ when #iis smaller than or equals to #j

Details:

When the condition is established, the program will go to execute line n; if it isn't established, it will

execute the following in sequence.

When the [conditional expression] is ignored, the program will execute the GOTO sentence unconditionally.

The n of GOTO sentence must exist in the program, or else the program will alarm.

#i, #j, and n can be replaced with variables. For the segments that contain the order No. n specified by GOTO n, the order No. n must be in front of the segment, or else error may occur due to lack of keywords when the program jumps.

If the specified segment contains "/" in the front and is followed by Nn, the ignoring function of the segment will be invalid, and this segment will still go to execute.

When GOTO instruction is executed, the system will search downwards first; if not found, the system will return and search downwards from the program header; if still not found until the calling segment, the system will send alarm information.

EQ and NE only can be used for integers, and the values with decimal fraction should be compared with GT, GE, LT, and LE instructions.

Cycle conditional instruction

Format:

```
WHILE[expression]DO m;(m=1,2,3…127)
…
END m;
```

Details:

When the conditional expression is established, the programs between WHILE and END will be executed repeatedly; if not established, the program will go to next segment of END m directly.

WHILE [expression] DO m and END m should be used in pair. If the LEaan line of WHILE [expression] is ignored, the segments between DO m and END m will be repeated endlessly. The range of M is 1-127.

WHILE allows nesting up to 27 levels.

(1) Same identifier and can be used repeatedly

Correct

(2) The identifier of WHILE~Dom can be specified freely

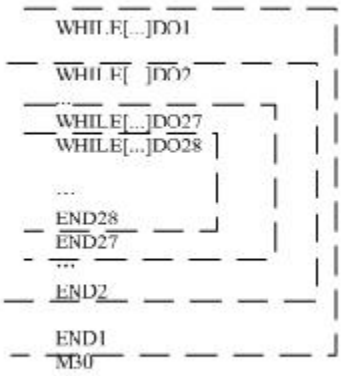
Correct

(3) WHILE~DO m contains up to 27 level; the range of m is 1~127, and can be specified freely.

Correct

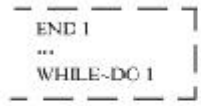
Note: m can't be used repeatedly after specified in nesting

(4) The level of WHILE~DO m can't exceed 28



incorrect

(5)WHILE~DO m must be specified before END m



Incorrect

(6)WHILE~DO m must be corresponded in one program

Incorrect

(7)WHILE~DO m can' t be crossed

Incorrect

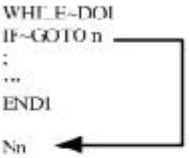
(8) The calling of M98, G65, G66 and other subroutines can be executed between WHILE~DO m

Allow

(9)GOTO can' t jump into WHILE cycle

Incorrect

(10)GOTO can jump out of WHILE cycle

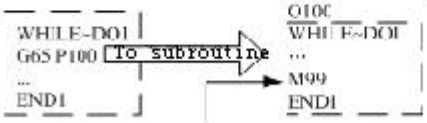


Correct

(11)Call subroutine in WHILE~DO cycle, and execute WHILE~DO cycle in the subroutine; at this moment, the nesting levels of WHILE is the sum of main program and subroutine, and can' t exceed 27.

(12)In macro program, M99 will cause program error if WHILE and END are not used in pair.

Incorrect



M99 causes that DO and END can' t be matched

4.8 Notes of using macro

Macro program uses variables to calculate and combine the NC program described by the logic, making the program more versatile. However, since the logical calculation is flexible, it may lead to some hidden errors; to avoid logic errors, it is necessary to note the mode when writing macros.

(1) Variable initialization; all the variables used in the program should be initialized at the beginning of the program; the variables for transfer also require an intermediate variable, in order to avoid error due to parameters modified by the program during multiple processing.

(2) In main program, subroutine or macro, please use local variables as much as possible; all the local variables will be cleared during program calling, in order to keep a clean environment for programming. Even if the reference is false, it will be located easily.

(3) Same as subroutine, macro can't be used in tool radius compensation; therefore, please cancel the compensation before calling.

4.9 Macro variable user parameters system configuration

- Macro variables contain [User] menu, which is used to rename the macro variable addresses related to process parameters, in order to make the operation more intuitional; the specific method is to configure the system with csv file;
- CSV is an Excel format. Please create a configuration datasheet in the following format in Excel, save as CSV file, name the file as SYSTABLE.CSV, and save it in directory ADT. Select [Parameter > Management > Import CSV system configuration], and the system will automatically check whether the file exists; if yes, the system menus will be configured with this file.
- The configuration of CSV macro variable user parameters follows:

Example of user macro configuration: the range of sequence number is 17~100 and the range of corresponding macro address is 500~999; this macro address is nonvolatile. The user can customize up to 50 addresses.

	A	B	C	D
1	User macro configuration	S/N	User-defined name	Corresponding macro address
2	User-defined macro variable name 1	17	User-defined name 1	500
3	User-defined macro variable name 2	18	User-defined name 2	501
4	User-defined macro variable name 3	19	User-defined name 3	502
5	User-defined macro variable name 4	20	User-defined name 4	503
6	User-defined macro variable name 5	21	User-defined name 5	504
7				
8				
9	User-defined alarm setting	S/N	External alarm S/N	User-defined alarm prompt
10	User-defined alarm setting 1	200	1	External alarm 1
11	User-defined alarm setting 2	201	2	External alarm 2
12	User-defined alarm setting 3	202	3	External alarm 3
13	User-defined alarm setting 4	203	4	External alarm 4
14	User-defined alarm setting 5	204	5	External alarm 5
15				

Example of user-defined alarm configuration: the range of the sequence number is 200~215, the range of corresponding external alarm sequence number is 1~16, sequence number corresponds to bit number 1~16 of external alarm register, and the later alarm prompt is the content generated by the alarm of current number. No sequence number can be repeated.

4.10 Extended special macro functions

4.10.1. RCOOR read workpiece coordinates

Function description: Read G54-G599 workpiece coordinates

Parameter: AXIS read axis No. 1-6 corresponding to Ax Ay….Ac

COOR reads workpiece coordinates 123….15 corresponding to G45 G55….G599

float RCOOR(INT8U AXIS,INT8U COOR)

Example: The program for reading the variable X-#100 in the G54 coordinates is as below:

#100=RCOOR[1,1]

4.10.2. RMACPOS read machine tool coordinates

Function description: Read machine tool's coordinates

Parameter: AXIS No.

Returned value: It return the current machine tool's coordinates Unit mm

float RMACPOS(INT8U AXIS)

Example: The program for reading the variable X-#100 in the machine tool coordinates is as below:

#100= RMACPOS[1]

4.10.3. WMACPOS write machine tool coordinates

Function description: Write machine tool's coordinates

Parameter: AXIS NO. VAL writes coordinates Unit mm

Returned value 0

INT16U WMACPOS (INT8U AXIS,float VAL)

Example: The program for reading the variable X-#100 in the machine tool coordinates is as below:

#100=20 (the input value for the X mathine tool is 20)

WMACPOS [1,#100]

4.10.4. SPEEDS set interpolation speed

Function description: Set the interpolation speed

Parameter: STARTV initial speed of interpolation mm/min set to 0 in default

SPEEDV drive speed of interpolation mm/min set to 0 in default

Returned value 0

INT16U SPEEDS(INT32U START,INT32U speed)

Example: If the set initial interpolation speed is 200, the drive speed is 1500, the usage is as follows:

SPEEDS[200,1500]

4.10.5. SPEEDA set positioning speed

Function description: Set positioning speed

Parameter: AXIS No.

STARTV the initial speed of AXIS mm/min set to 0 in default

SPEEDV the drive speed of AXIS mm/min set to 0 in default

Returned value 0

INT16U SPEEDA(INT8U AXIS,INT32U START,INT32U speed)

Example: Set No. 7 AXIS, the initial speed and running speed as 200 and 2000 respectively, as follows

SPEEDA[7,200,2000]

4.10.6. MOVEABS single axis moves to the machine's position

Function description: Single axis moving to the machine's position The number of the shifted single axis POS mm Move to the machine's position

Parameter MODE: 0: Waiting for the end of the motion 1: Not waiting for the end of the motion

AXIS: Number of the motion axis

POS: Move the machine's position Unit: mm

Returned value: If correct, it returns 0; If error, it returns the value >1

INT16U MOVEABS(INT8U MODE,INT8U AXIS,float POS)

Example: set the position of No. 7 AXIS to move to the machine's coordinates 100, the program is as follows:

SPEEDA[7,200,2000] (Set the running speed)

MOVEABS[0,7,100]

4.10.7. MOVEREL relative moved position of single axis

Function description: Relative moved position of single axis AXIS Number of the moved single axis POS mm Relatively moved position

Parameter: MODE: 0: wait for the end of the motion 1: not wait for the end of the motion

AXIS: Number of the motion axis

POS: Relatively moved position Unit: mm

Returned value: If correct, it returns 0; If error, it returns the value >1

INT16U MOVEREL (INT8U MODE, INT8U AXIS, float POS)

4.10.8. MOVEASA two axes move to the machine's position (positioning or interpolation)

Function description: Two axes move to the machine's position

Parameter MODE: Motion mode MODE motion mode (0: Positioning - wait for the end of the motion; 1: interpolation - wait for the end of the motion; 2: positioning - not wait for the end of the motion; 3: interpolation – not wait for the end of the motion)

AXIS: A1: Number of the motion axis 1 P1: motion displacement 1 mm A2: Number of the motion axis 2 p2: motion displacement 2 mm

Returned value: If correct, it returns 0; If error, it returns the value >1

INT16U MOVEASA(INT8U MODE,INT8U A1,float P1,INT8UA2,float P2)

4.10.9. MOVERSA relative moved position of two axes (positioningor interpolation)

Function description: Relative moved position of two axes

Parameter: MODE: Motion mode MODE motion mode (0: Positioning - wait for the end of the motion 1: Interpolation - wait for the end of the motion 2: Posiioning - not wait for the end of the motion 3: Interpolation - not wait for the end of the motion)

AXIS: A1: Number of the motion axis 1 P1: Motion displacement 1 mm A2: Number of the motion axis 2 p2: Motion displacement 2mm)

Returned value: If correct, it returns 0; If error, it returns the value >1

INT16U MOVERSA(INT8U MODE,INT8U A1,float P1,INT8UA2,float P2)

4.10.10. MOVEASB three axes move to the machine's position (positioning or interpolation)

Function description: Three axes move to the machine's position

Parameter: MODE: Motion mode MODE motion mode (0: Positioning wait for the end of the motion 1: Interpolation wait for the end of the motion 2: Positioning not wait for the end of the motion 3: Interpolation not wait for the end of the motion)

AXIS: A1 Number of the motion axis 1 P1: Motion displacement 1mm A2: Number of the motion axis 2 p2: Motion displacement 2mm….

Returned value: If correct, it returns 0; If error, it returns the value >1

INT16U MOVEASB(INT8U MODE,INT8U A1,float P1,INT8U A2,float P2,INT8U A3,float P3)

4.10.11. MOVERSB Relative Position of Motion of Three axes

Function description: Relative position of motion of three axes

Parameter MODE: Motion mode MODE motion mode (0: Positioning wait for the end of the motion 1: Interpolation wait for the end of the motion 2: Positioning not wait for the end of the motion 3: Interpolation not wait for the end of the motion)

AXIS: A1 Number of the motion axis 1 P1: Motion displacement 1mm A2: Number of the motion axis 2 p2: Motion displacement 2mm….

Returned value: If correct, it returns 0; If error, it returns the value >1

INT16U MOVERSB(INT8U MODE,INT8U A1,float P1,INT8U A2,float P2,INT8U A3,float P3)

4.10.12. MOVEASC absolute position of motion of multiple axes (positioning or interpolation)

Function description: Absolute position of motion of multiple axes

Parameter: MODE: Motion mode MODE motion mode (0: Positioning wait for the end of the motion 1: Interpolation wait for the end of the motion 2: Positioning not wait for the end of the motion 3: Interpolation not wait for the end of the motion)

AXISTOTAL Total motion axis (pay attention to the six maximum axis and one minimum axis)… variable parameter (for example, No. 1 axis Position 1 No. 2 axis Position 2 No. 3 axis Position 3)

Example: Two-axis motion MOVEASC[0,2,1,100,2,100]

Example: Three-axis motion MOVEASC[0,3,1,100,2,100,3,100]

Example: Four-axis motion MOVEASC[0,4,1,100,2,100,3,100,4,100]

Example: Five-axis motion MOVEASC[0,5,1,100,2,100,3,100,4,100,5,100]

Example: Six-axis motion MOVEASC[0,6,1,100,2,100,3,100,4,100,5,100,6,100]

Returned value: If correct, it returns 0; If error, it returns the value >1

4.10.13. MOVERSC relative position of motion of multiple axes (positioning or interpolation)

Function description: Relative position of motion of multiple axes

Parameter: MODE: Motion mode MODE motion mode (0: Positioning wait for the end of the motion 1: Interpolation wait for the end of the motion 2: Positioning not wait for the end of the motion 3: Interpolation not wait for the end of the motion)

AXISTOTAL Total motion axis (pay attention to the six maximum axis and one minimum axis)… variable parameter (for example, No. 1 axis Position 1 No. 2 axis Position 2 No. 3 axis Position 3)

Example: Two-axis motion MOVEASC[0,2,1,100,2,100]

Example: Three-axis motion MOVEASC[0,3,1,100,2,100,3,100]

Example: Four-axis motion MOVEASC[0,4,1,100,2,100,3,100,4,100]

Example: Five-axis motion MOVEASC[0,5,1,100,2,100,3,100,4,100,5,100]

Example: Six-axis motion MOVEASC[0,6,1,100,2,100,3,100,4,100,5,100,6,100]

Returned value: If correct, it returns 0; If error, it returns the value >1

4.10.14. WRITEOUT Write Physical Output

Function description: Write physical output (OUT)

Parameter NUM Port No. VALUE output level (0 or 1)

Returned value: If correct, it returns 0; If error, it returns the value >1

INT16U WRITEOUT(INT32U NUM,INT8U VALUE)

4.10.15. WRITELED Write Physical LED

Function description: Write physical LED

Parameter NUM: Port No. VALUE output level (0 or 1)

Returned value: If correct, it returns 0; If error, it returns the value >1

INT16U WRITELED(INT32U NUM,INT8U VALUE)

4.10.16. READOUT Read Physical Output

Function description: Read the status of physical output (OUT)

Parameter NUM: Port No.

Returned value: If correct, it returns 0 for low electrical level or 1 for high electrical level;If error, it returns the value>1

INT16U READOUT(INT32U NUM)

4.10.17. READIN Read Physical Input

Function description: Read the status of Physical Input (IN)

Parameter NUM: Port No.

Returned value: If correct, it returns 0 for low electrical level or 1 for high electrical level;If error, it returns the value>1

INT16U READIN(INT32U NUM)

4.10.18. READLED Read Physical LED

Function description: Read the output status of the physical LED

Parameter NUM: Port No.

Returned value: If correct, it returns 0 for low electrical level or 1 for high electrical level;If error, it returns the value>1

INT16U READLED(INT32U NUM)

4.10.19. MOVEWAITIN Search and Wait for the Input Signals in Motion

Function description: Search and wait for the IN signal to effectively stop in motion

Parameter AXIS: Number of the motion axis SPEEDV: Searching speed POS: Searching distance (relative distance Unit mm) IN: Wait input signals TIMEOUT: timeout Unit ms wait continuously when it is 0

Returned value: 0: Detect the IN stop the motion of the axis Timeout: 1 Error: Return>1

INT16U MOVEWAITIN(INT8U AXIS,INT32U SPEEDV,float POS,INT8U IN, INT32U TIMEOUT)

4.10.20. WAITMOVE Wait for the End of the Motion of All Axes

Function description: Wait for the end of the motion of all axes

Parameter: Number of the axis A1 A2 A3 and A4 after the end of motion (Note: Inputting 0 for A1, A2, A3 and A4 results to invalid axis which will not wait)

Returned value: If all axes end the motion, it returns 0; If error, it returns the value>1.

INT16U WAITMOVE(INT8U A1,INT8U A2,INT8U A3,INT8U A4)

4.10.21. WAITMOVED Wait for the End of Motion of All Axes

Function description: Wait for the end of the motion of all axes (Note: This function is used together with the MOVExxx command and will err when it is used alone.)

Parameter: AXISTOTAL Total number of the axis that wait to end the motion (pay attention to the six maximum axis and one minimum axis)…variable parameter (For example: No. 1 AXIS No. 3 AXIS…)

Eample: See below

WAITMOVE[1,2] Wait for No. 2 AXIS

WAITMOVE[2,2,3] Wait for No. 2 and No. 3 AXIS

WAITMOVE[3,4,5,6] Wait for No. 3, No. 4, No. 5 and No. 6 AXIS

WAITMOVE[4,1,2,3,5] Wait for No. 1, No. 2, No. 3 and No. 5 AXIS

WAITMOVE[5,1,2,3,5,6] Wait for No. 1, No. 2, No. 3, No. 5 and No. 6 AXIS

WAITMOVE[6,1,2,3,4,5,6] Wait for No. 1, No. 2, No. 3, No. 4, No. 5 and No. 6 AXIS

Returned value: If all axes end the motion, it returns 0; If it errs, it returns the value>1.

4.10.22. WRITEPLC Write Physical or Auxiliary Output Point

Function description: Write the auxiliary point with the signal quantity interacting with PLC and larger than 1024

Parameter: NUM: Port No. VAL Output Signal value (0 or 1)

Note: (NUM<512 Write physical output point OUT)

(NUM >=1024 NUM < = 7999 write auxiliary point, which is the auxiliary point with the signal quantity interacting with PLC)

Returned value: If correct, it returns 0; If error, it returns the value >1

INT16U WRITEPLC(INT32U NUM,INT8U VAL)

4.10.23. READPLC Read Physical or Auxiliary Output Point

Function description: Read physical or auxiliary output point

Note: (NUM < 512 Read physical input point)

(NUM >=1024 NUM < = 7999 Read auxiliary output point, which is the auxiliary point with the signal quantity interacting with PLC)

If correct, it returns the auxiliary output coil value 0 or 1; If error, it returns value>1

INT16U READPLC(INT32U NUM)

4.10.24. WAITPLC Timeout Waiting for Read of Physical or Auxiliary Input Point

Function description: Timeout waiting for read of physical or auxiliary input point

Parameter NUM physical input point 0-511 or the auxiliarypoint 1024-7999 with signal quantity interacting with PLC VAL read effective value (0 or 1) TIMEOUT>0 Timeout (ms) 0 Wait continuously until the signal is detected or [Press the Reset to quit]

Returned value In case of VAL, it returns 0 In case of no VAL after timeout, returns 1; In case of error, it returns the value>1

INT16U WAITPLC(INT32U NUM,INT8U VAL, INT32U TIMEOUT)

4.11 Special M Code

M Code for axis synchronization and dynamic mapping

4.11.1. Cancel the synchronization of all axis or switch function (M10002)

Execute this M order to restore the default synchronization of all axes

4.11.2. Process after switching Axis X to Axis A (M10003)

Execute this M order to switch the Axis X to Axis A for pulse port output

4.11.3. Process Axis X and Axis A synchronously (M10004)

Execute this M order to output the pulse of Axis X and Axis A at the X position synchronously

4.11.4. Process after switching Axis Y to Axis A (M10005)

Execute this M order to switch the Axis Y to Axis A for pulse port output

4.11.5. Process Axis Y and Axis A synchronously (M10006)

Execute this M order to output the pulse of Axis Y and Axis A at the Y position synchronously

4.11.6. Process after switching Axis Z to Axis A (M10000)

Execute this M order to switch the Axis Z to Axis A for pulse port output

4.11.7. Process Axis Z and Axis A synchronously (M10001)

Execute this M order to output the pulse of Axis Z and Axis A at the Z position synchronously

4.11.8. Process after switching Axis X to Axis B (M10007)

Execute this M order to switch the Axis X to Axis B for pulse port output

4.11.9. M10008 Process Axis X and Axis B synchronously (M10008)

Execute this M order to output the pulse of Axis X and Axis B at the X position synchronously

4.11.10. Process after switching Axis X to Axis C (M10009)

Execute this M order to switch the Axis X to Axis C for pulse port output

4.11.11. Process Axis X and Axis C synchronously (M10010)

Execute this M order to output the pulse of Axis X and Axis C at the X position synchronously

4.11.12. Process after switching Axis Y to Axis B (M10011)

Execute this M order to switch the Axis Y to Axis B for pulse port output

4.11.13. Process Axis Y and Axis B synchronously (M10012)

Execute this M order to output the pulse of Axis Y and Axis B at the Y position synchronously

4.11.14. Process after switching Axis Y to Axis C (M10013)

Execute this M order to switch the Axis Y to Axis C for pulse port output

4.11.15. M10014 Process Axis Y and Axis C synchronously (M10014)

Execute this M order to output the pulse of Axis Y and Axis C at the Y position synchronously

4.11.16. Process after switching Axis Z to Axis B (M10015)

Execute this M order to switch the Axis Z to Axis B for pulse port output

4.11.17. Process Axis Z and Axis B synchronously (M10016)

Execute this M order to output the pulse of Axis Z and Axis B at the Z position synchronously

4.11.18. Process Axis A, Axis B and Axis C synchronously (M10017)

Execute this M order to output the pulse of Axis Z, Axis A, Axis B and Axis C at the Z position synchronously

4.12 Special Program Segment

2. Regarding to the system start-up, the M Code and automatic cutter regulating program in the No. 23 program of [Parameter], [Management] of the system should be set to be User-Def and then a M_FUNC.NC file will be generated under the MACRO of D: local disc in the [File] of file management.

4.12.1. M2000

M2000: Automatic system zeroing order. When M2000 appears in the processing program of the running system, the system will execute the mechanical zeroing operation.

4.12.2. M2203

M2203: When the system is started, if there is O2203….M3000% in the M_FUNC_NC, the system will call M2203 program segment automatically.

4.12.3. M2201

M2201: When the processing program is started, if there is O2201….M3000% in the M_FUNC_NC after the system is started, the system will call M2201 program segment automatically.

4.12.4. M2202

M2202: When the processing program is over, if there is O2202….M3000% in the M_FUNC_NC after the system is started, the system will call M2202program segment automatically.

4.12.5. M2200

M2200: If there is O2200…M3000% in the M_FUNC_NC before the processing program executes G00 order, the system will call M 2200 program segment.

4.12.6. M2205

M2205: Automatic cutter regulating macro program. If there is O2205…M3000% in the M_FUNC.NC, the system will call this macro program to complete the cutter regulation automatically.

4.12.7. M2206

M2206: Alarm and automatic execution of macro program. If there is O2206…M3000% in the M_FUNC.NC, the system will alarm and execute macro program automatically.

4.12.8. M2207

M2207: Reset and automatic execution of macro program. If there is O2207…M3000% in the M_FUNC.NC, the system will reset and execute the macro program automatically.

4.12.9. M2208

M2208: Execution of macro program before automatic zeroing of the system. If there is O2208…M3000% in the M_FUNC.NC, the system will execute the macro program automatically before zeroing.

4.12.10. M2209

M2209: Execution of macro program after automatic zeroing of the system. If there is O2208…M3000% in the M_FUNC.NC, the system will execute the macro program automatically after zeroing.

4.12.11. M2212

M2212: Execution of M Code after suspension

4.12.12. M2213

M2213: Restart andexecution of M Code after suspension

4.12.13. M2214

M2214: Execution of M Code before suspension

4.12.14. M2216

M2216: Custom M Code for moving Axis X manually

4.12.15. M2217

M2217: Custom M Code for moving Axis Y manually

4.12.16. M2218

M2218: Custom M Code for moving Axis Z manually

4.12.17. M2219

M2219: Custom M Code for moving Axis A manually

4.12.18. M2220

M2220: Custom M Code for moving Axis B manually

4.12.19. M2221

M2221: Custom M Code for moving Axis C manually

4.12.20. M2222

M2222: Execution of M Code before MDI is started for processing

4.13 M Code Segment Activated by External Input Point

3. Execution of M Code triggered through the external input signal INxx

Two hundred input points of IN00-IN99 corresponding to M2000-M2199 of the M Code triggered through the external input signal. (Note: M2000 is within the O20000…M3000% program segment in the M_FUNC.NC) For example (OUT00 output controlled through the IN14 input point):

Set the No. 23 program of [Parameter], [Management] of the system should be set to be User-Def and then a M_FUNC.NC file will be generated under the MACRO of D: local disc in the [File] of file management.

Add the following program segment to the end of the file.

```
O2014
IF[#1014==0] (IN14 input signal connection/disconnection detected)
{
#1400=1 (OUT00 output controlled to open)
}
IF[#1014==1] ( IN14 input signal disconnection detected)
{
#1400=0 (OUT00 output controlled to close)
}

M3000
%
```

4.14 Auxiliary Channel GRUN 4, 5, 6 and 7

The system can be configured with two major channels and four auxiliary channels which includes GURN4, GURN5 and GURN6. User can configure to start together with the major channels. It can be applied to some simple feeding mechanisms to raise the efficiency of the equipment. GRUN7 is an auxiliary channel which can start automatically once electrified. The program files of the channel are all saved in the GRUN4_NC, GRUN5_NC, GRUN6_NC AND GRUN7_NC under the C:\MACRO. Attentions: The auxiliary channels are only applied in the occasions that real-time performance is not highly required. Besides, delay order like G04 PXX needs to be added to the auxiliary channel cycle to avoid other channels to be affected by the dispatch.

5. Instruction on Custom CAM

5.1 Overview

CAM instruction is used as an interface customization function. User can customize the pictures, parameter names, value range, processing order on the CAM interface. With this interface, user can search, load, save, delete, fast program, copy, stick, clear and help functions by operating the correspondent menus. Currently user can configure as many as 30 kinds of CAM types through CAMTEACH.CT. As it has instruction function, user can edit multiple processing pictures. When using the CAM instruction, user needs to put the CAMTEACH folder and CAMTEACH.CT file into the ADT folder of D disc of the controller and then restart it. There is no need to restart the system when the file is separately put into the CAMTEACH folder.

5.2 Introduction of CAM Instruction Interface

The interface is shown in the followingFigure 1 after the CAM instruction is configured.

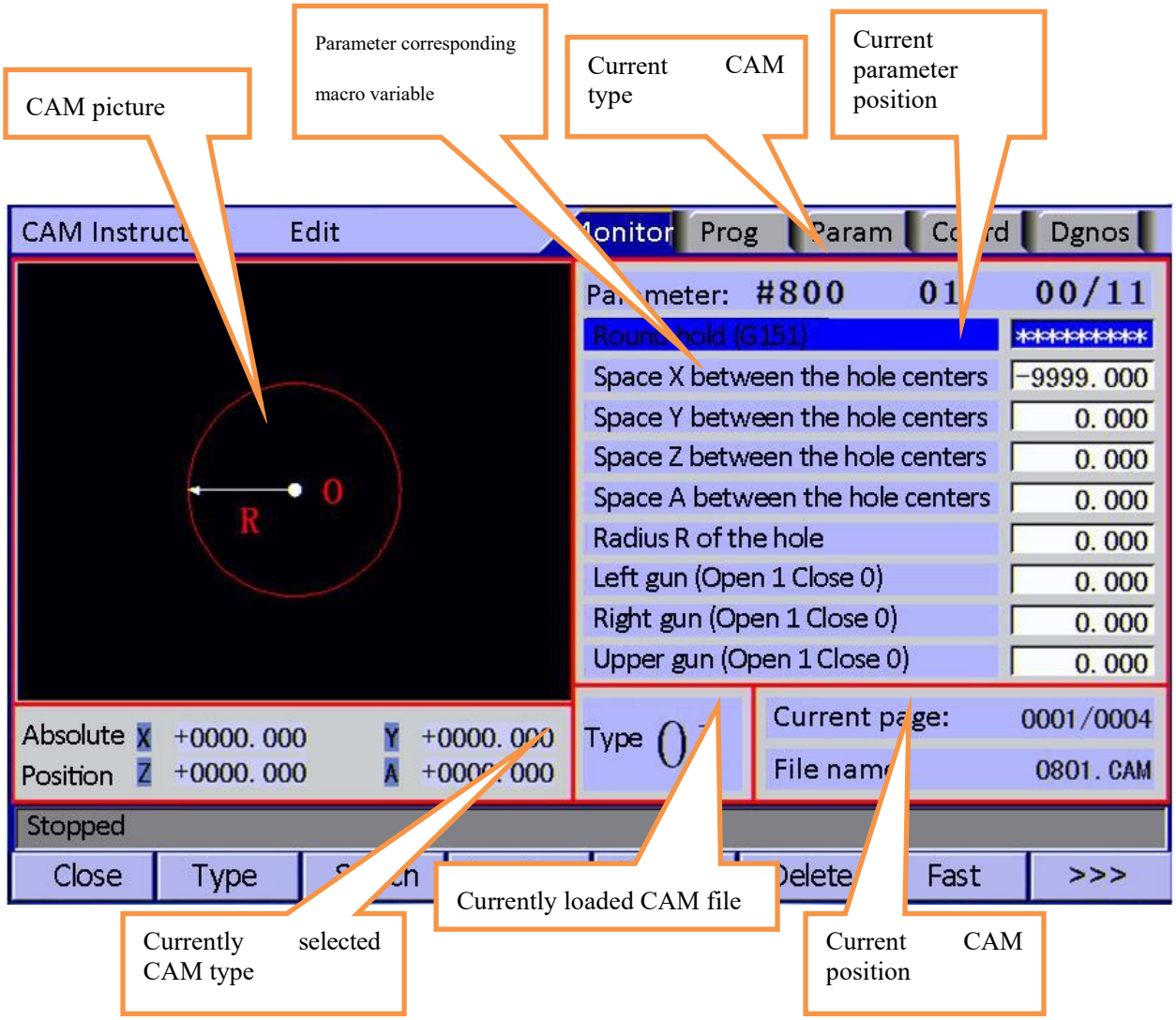


Figure 1 Interface of CAM instruction

Each part of the CAM instruction interface are explained as follows.

CAM picture

It displays 24-digit colored 380x380 pixels BMP picture customized by user so that the parameter can be easily understood.

Parameter corresponding macro variable:

Current parameter corresponding macro variable used when user is programming

Current CAM type:

The current CAM type uses the type listed in the CAMTEACH.CT configuration table.

Current parameter's position:

The previous data displays the current parameter position of the cursor and the latter data displays how many process parameters in the current CAM picture.

Currently selected CAM type:

The current CAM type in the CAMTEACH.CT configuration table for element to be inserted

Currently loaded CAM file:

If the instruction file has been saved, it displays the name of the instruction file which is being operated. Otherwise, it displays nothing.

The previous data displays the current position of the CAM element in the current file and the latter data displays how many CAM elements in the current file.

5.3 CAM instruction menu functions

There are several menus in the CAM instruction interface to help user search, copy, stick, delete, clear and fast program. The functions are detailed in the Table 1.

Name of menu	Functions
Type	In case of the element to be inserted, choose CAM type in the configuration file CAMTEACH.CT.
Search	Designate the CAM position in the location data file
Load	Read the content of the CAM instruction file which has been saved
Saved as	Save the current CAM instruction file
Delete	Delete a designated CAM data from the current position.

Fast program	Designate the number of copies of CAM from the current position and then stick automatically for the designated times.
Copy	Number of copies of CAM from the current position
Stick	Insert the copied CAM data into the current position
Clear	Remove the current file
Help	Help

Table 1. CAM instruction menu functions

5.4 CAM Instruction Configuration File

The CAMTEACH.CT order of the CAM instruction file is explained as follows.

CTSTART: Configure the starting mark for the file

CTEND: Configure the ending mark for the file

CAMINFO: Starting mark of CAM data

A: CAM type, parameter range 1-30. Currently it supports 30 types at most and can be expanded according to the situation.

B: Number of parameters

C: CAM picture saving path, for example: 0:\\ADT\\CAMTEACH\\G151.bmp

D: Picture type: 380x380

E: The correspondent G code is G151-G180

G: The value after the parameter is the initial value involved by FGH.

F: Parameter name definition involved by FGH. The first CAM parameter is suggested to be used as note.

H: Value range involved by FGH. The first value is a minimum one and the second a maximum one.

The examples of the 'CAMTEACH.CT' of the CAM instruction file are as follows.

(Format description)

(A CAM type, parameter range 1-30)

(B Number of parameters)

(C Picture saving path, for example: 0:\\ADT\\CAMTEACH\\G151.bmp)

(D Picture type: 240x180 800x480)

(E The correspondent G code G151-G180)

(F Definition of parameter name, used together with FGH)

(G Parameter value, used together with FGH)

(H Value range, used together with FGH)

CTSTART

CAMINFO

A 1

B 11

C \ADT\CAMTEACH\G151.bmp

D 380x300

E 151

F Round hold (G151)

G 10.0

H 0,0

F Space X between the hole centers (O)

G -9999.0

H -9999,9999

F Space Y between the hole centers (O)

G 0.0

H -9999,9999

F Space Z between the hole centers (O)

G 0.0

H -9999,9999

F Space A between the hole centers (O)

G 0.0

H -9999,9999

F Radius R of the hole

G 0.0

H 0.5,9999

F Left gun (Open 1 Close 0)

G 0

H 0,1

F Right gun (Open 1 Close 0)

G 0

H 0,1

F Upper gun (Open 1 Close 0)

G 0

H 0,1

F Radius of the lead arc

G 0

H 0.5,100

F plane (G17G18G19)

G 0

H 0,2

CAMINFO

A 2

B 12

C \ADT\CAMTEACH\G152.bmp

D 380x300

E 152

F Square hole (G152)

G 0.0

H 0,0

F Space X between the hole centers (O)

G 0.0

H -9999,9999

F Space Y between the hole centers (O)

G 0.0

H -9999,9999

F Space Z between the hole centers (O)

G 0.0

H -9999,9999

F Space A between the hole centers (O)

G 0.0

H -9999,9999

F Length of the square hole (L)

G 0.0

H 0,9999

F Width of the square hole (W)

G 0.0

H 0,9999

F Left gun (Open 1 Close 0)

G 0

H 0,1

F Right gun (Open 1 Close 0)

G 0

H 0,1

F Upper gun (Open 1 Close 0)

G 0

H 0,1

F Radius of the lead arc

G 0

H 0.5,100

F Plane G 0(G17G18G19)

H 0,2

CAMINFO

A 3

B 12

C \ADT\CAMTEACH\G153.bmp

D 380x300

E 153

F Oblong hole (G153)

G 0.0

H 0,0

F Space X between the hole centers (O)

G 0.0

H -9999,9999

F Space Y between the hole centers (O)

G 0.0

H -9999,9999

F Space Z between the hole centers (O)

G 0.0

H -9999,9999

F Space A between the hole centers (O)

G 0.0

H -9999,9999

F Width of oblong hole (L)

G 0.0

H 0,9999

F Radius of oblong hole (R)

G 0.0

H 0,9999

F Left gun (Open 1 Close 0)

G 0

H 0,1

F Right gun (Open 1 Close 0)

G 0

H 0,1

F Upper gun (Open 1 Close 0)

G 0

H 0,1

F Radius of the lead arc

G 0

H 0.5,100

F Plane (G17G18G19)

G 0

H 0,2

CAMINFO

A 4

B 23

C \ADT\CAMTEACH\G154.bmp

D 380x300

E 154

F Oblong hole (G154)

G 0.0

H 0,0

F Space X between the hole centers (O)

G 0.0

H -9999,9999

F Space Y between the hole centers (O)

G 0.0

H -9999,9999

F Space Z between the hole centers (O)

G 0.0

H -9999,9999

F Space A between the hole centers (O)

G 0.0

H -9999,9999

F Arc radius (R1)

G 0.0

H 0,9999

F Arc radius (R)

G 0.0

H 0,9999

F Arc radius (R3)

G 0.0

H 0,9999

F Arc radius (R4)

G 0.0

H 0,9999

F Length of one side (L1)

G 0.0

H 0,9999

F Length of one side (L2)

G 0.0

H 0,9999

F Length of one side (L3)

G 0.0

H 0,9999

F Length of one side (L4)

G 0.0

H 0,9999

F Length of one side (L5)

G 0.0

H 0,9999

F Height (H1)

G 0.0

H 0,9999

F Height (H2)

G 0.0

H 0,9999

F Height (H3)

G 0.0

H 0,9999

F Height (H4)

G 0.0

H 0,9999

F Height (H5)

G 0.0

H 0,9999

F Left gun (Open 1 Close 0)

G 0

H 0,1

F Right gun (Open 1 Close 0)

G 0

H 0,1

F Upper gun (Open 1 Close 0)

G 0

H 0,1

F Radius of lead arc

G 0

H 0.5,100

F Plane (G17G18G19)

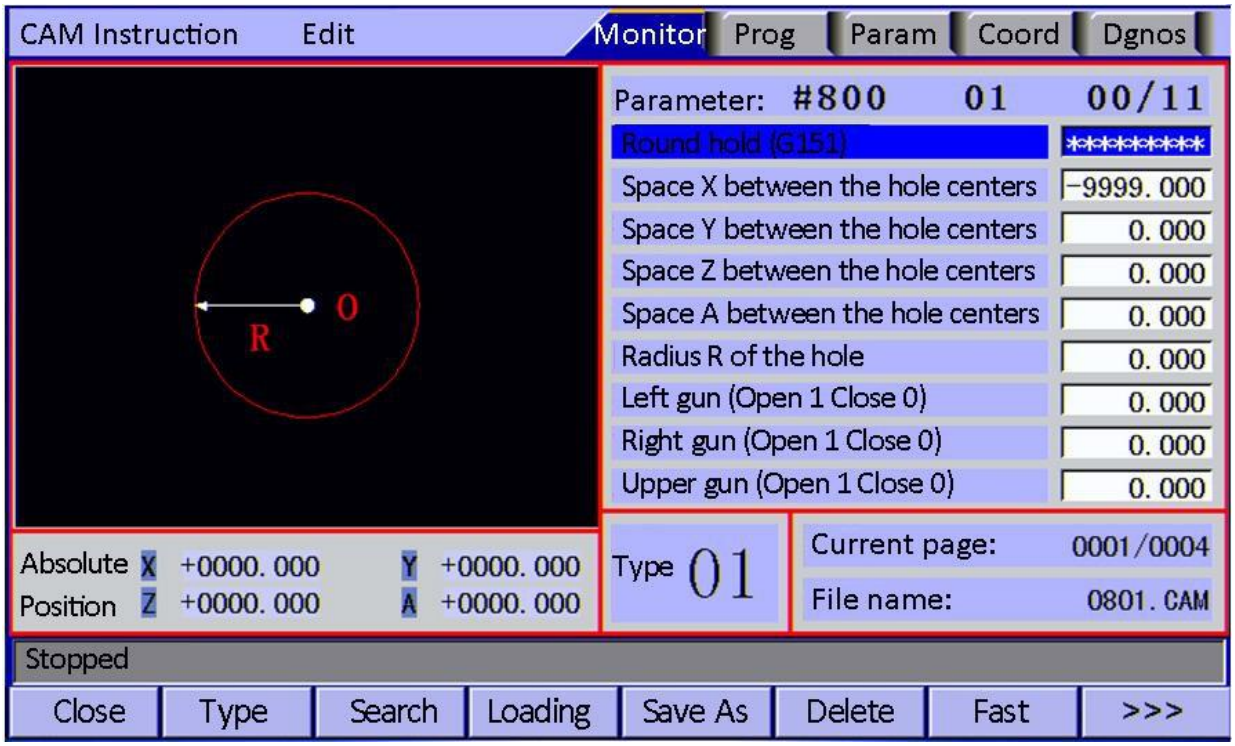
G 0

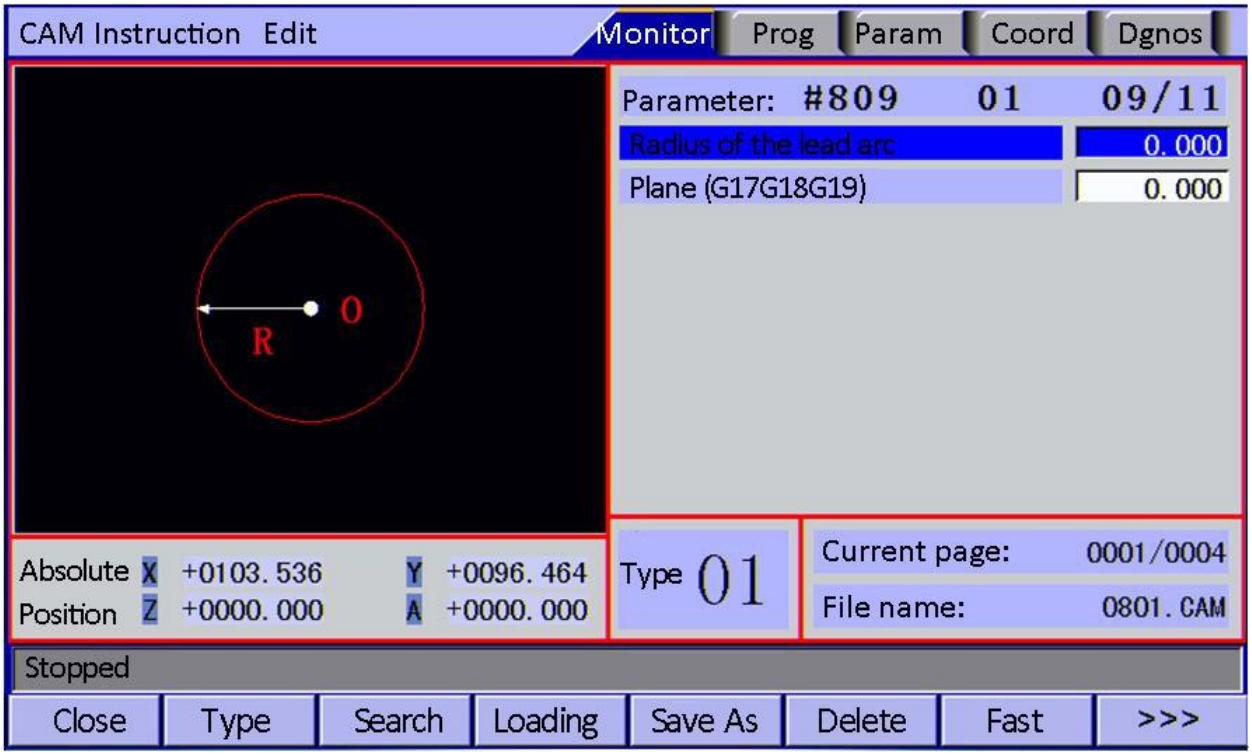
H 0,2

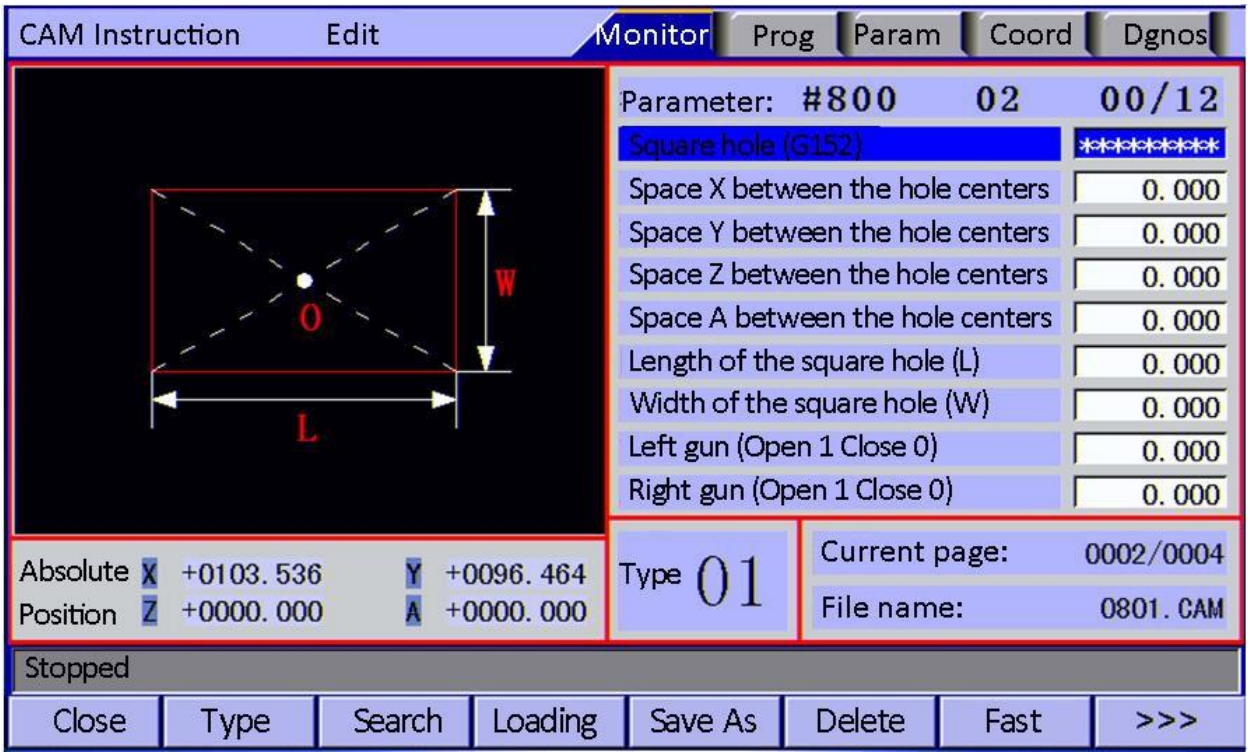
CTEND

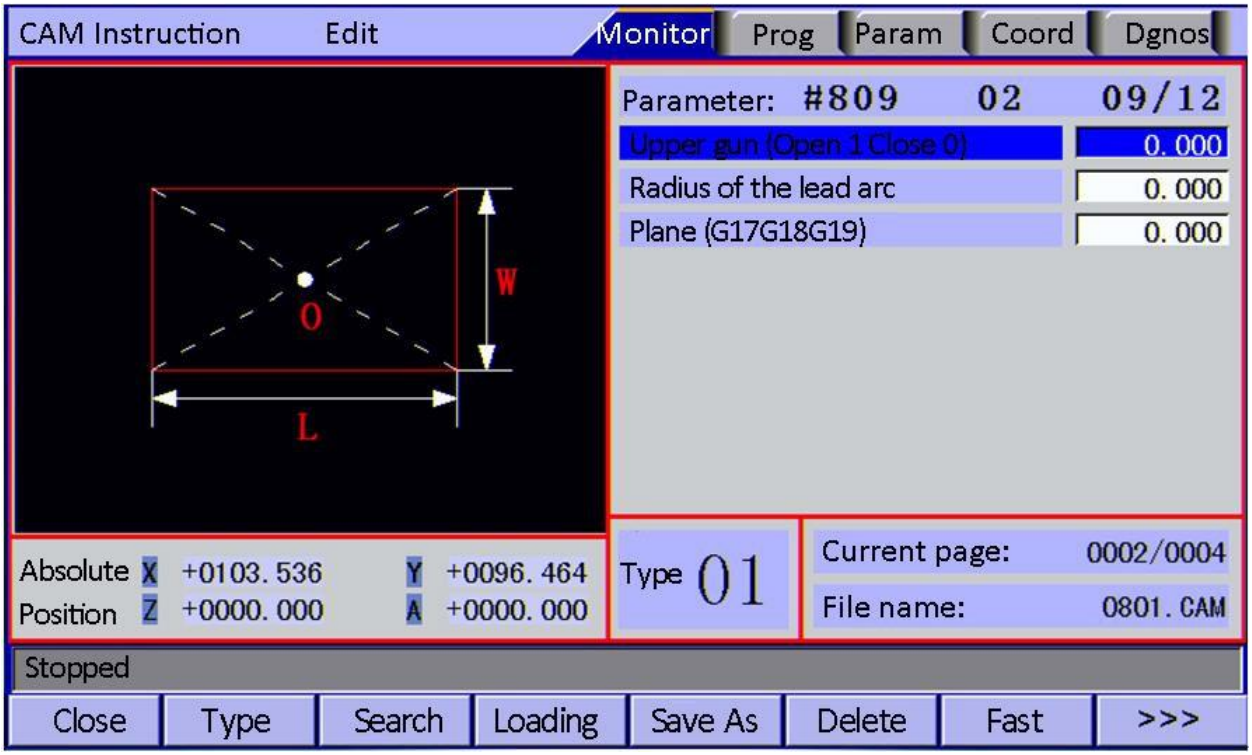
5.5 Schematic Diagram of CAM Instruction

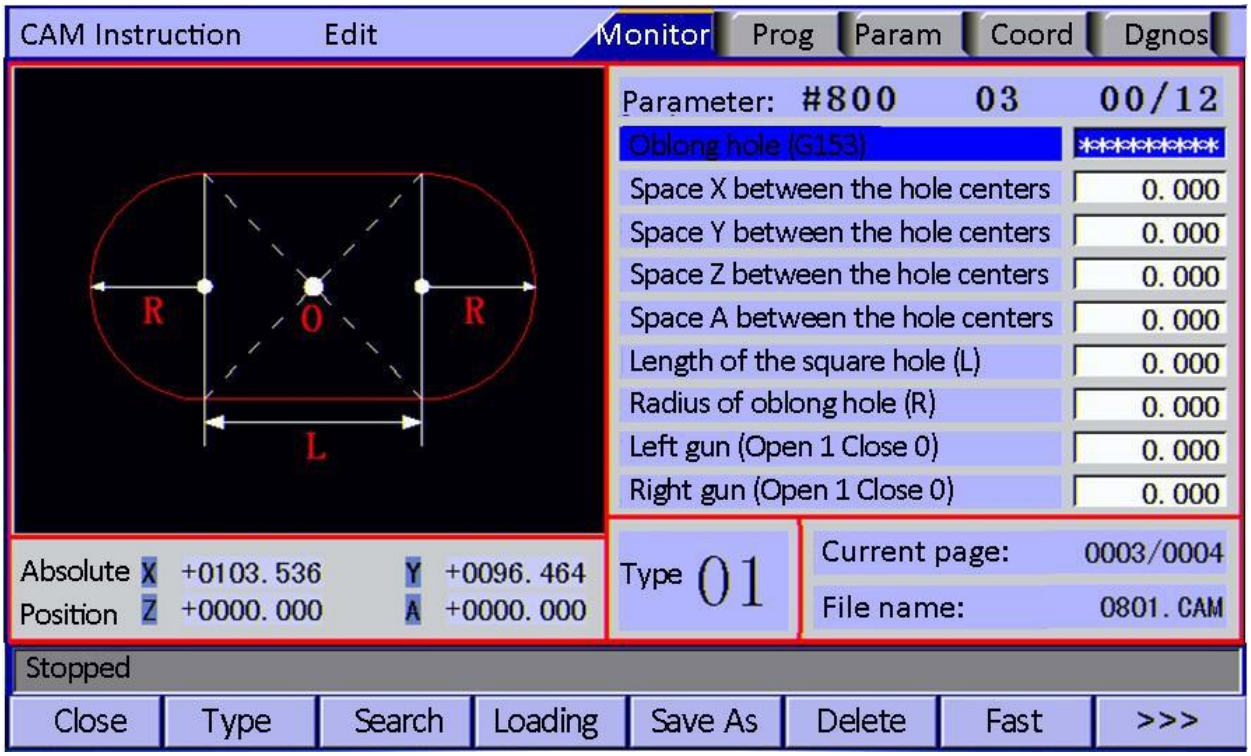
The schematic diagrams of CAM instruction are shown as follows.

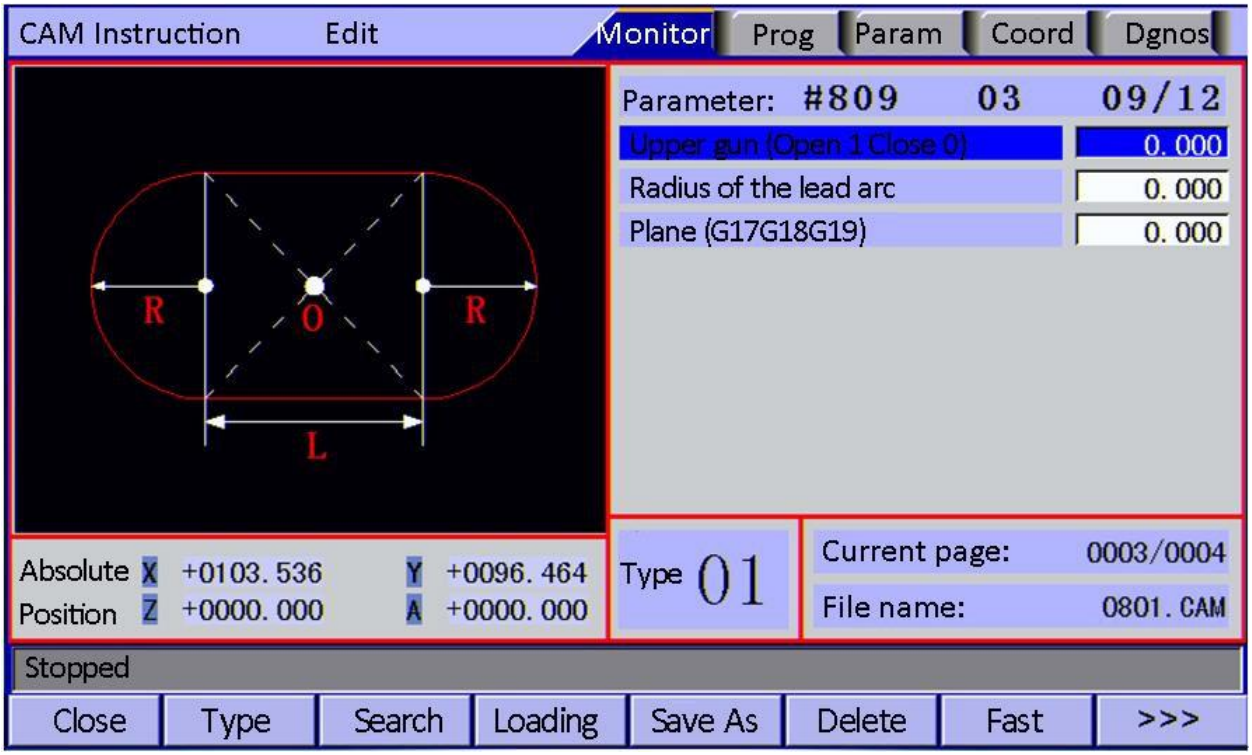


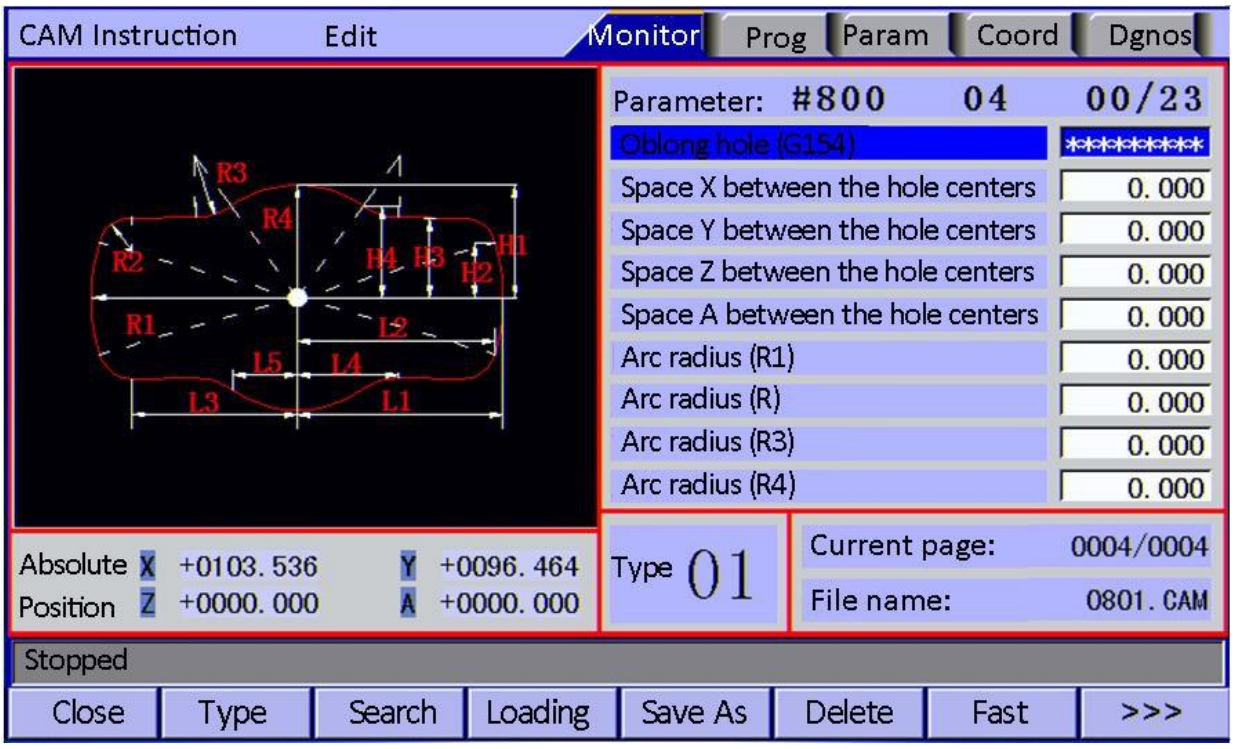

CAM Instruction Edit
Monitor
Prog
Param
Coord
Dgnos
R
O
Parameter: #809 01 09/11
Radius of the lead arc
0. 000
Plane (G17G18G19)
0. 000
Absolute X +0103. 536 Y +0096. 464
Position Z +0000. 000 A +0000. 000
Type 01
Current page: 0001/0004
File name: 0801. CAM
Stopped
Close
Type
Search
Loading
Save As
Delete
Fast
>>>

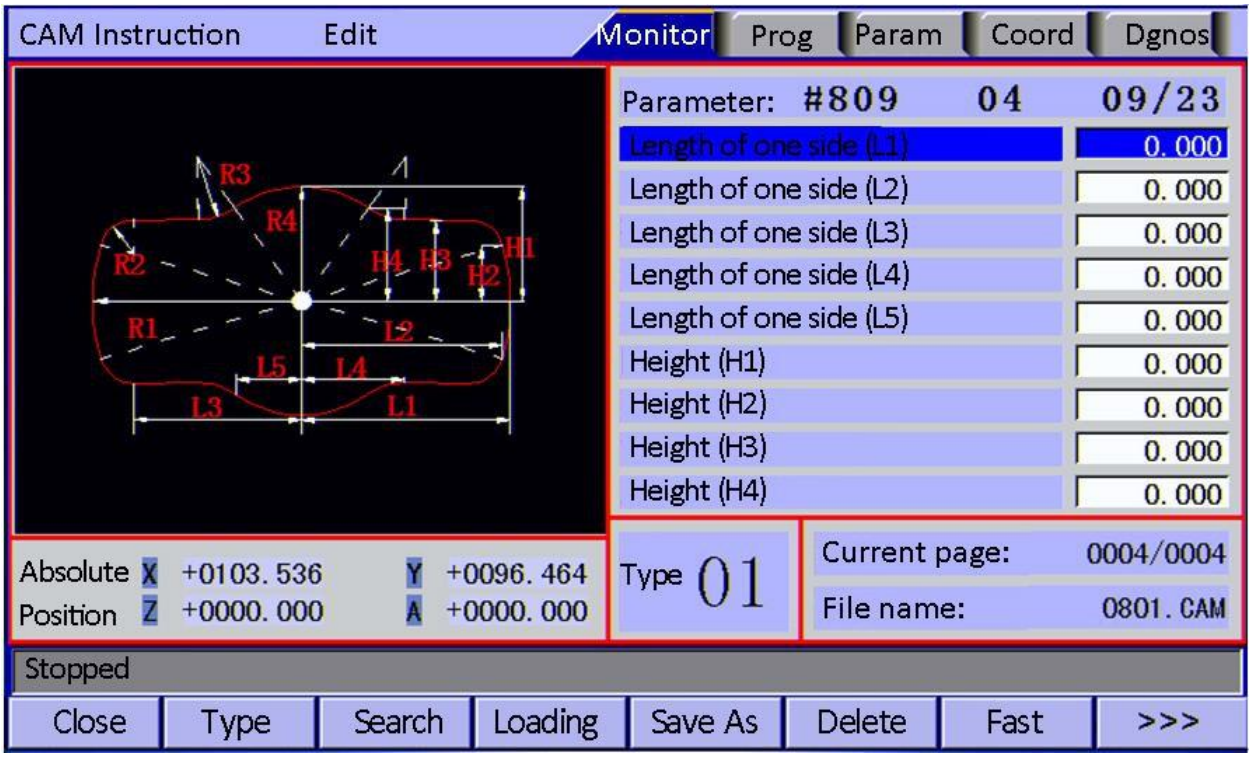

CAM Instruction Edit
Monitor
Prog
Param
Coord
Dgnos
W
O
L
Parameter: #800 02 00/12
Square hole (G152)
****
Space X between the hole centers
0. 000
Space Y between the hole centers
0. 000
Space Z between the hole centers
0. 000
Space A between the hole centers
0. 000
Length of the square hole (L)
0. 000
Width of the square hole (W)
0. 000
Left gun (Open 1 Close 0)
0. 000
Right gun (Open 1 Close 0)
0. 000
Absolute X +0103. 536 Y +0096. 464
Position Z +0000. 000 A +0000. 000
Type 01
Current page: 0002/0004
File name: 0801. CAM
Stopped
Close
Type
Search
Loading
Save As
Delete
Fast
>>>

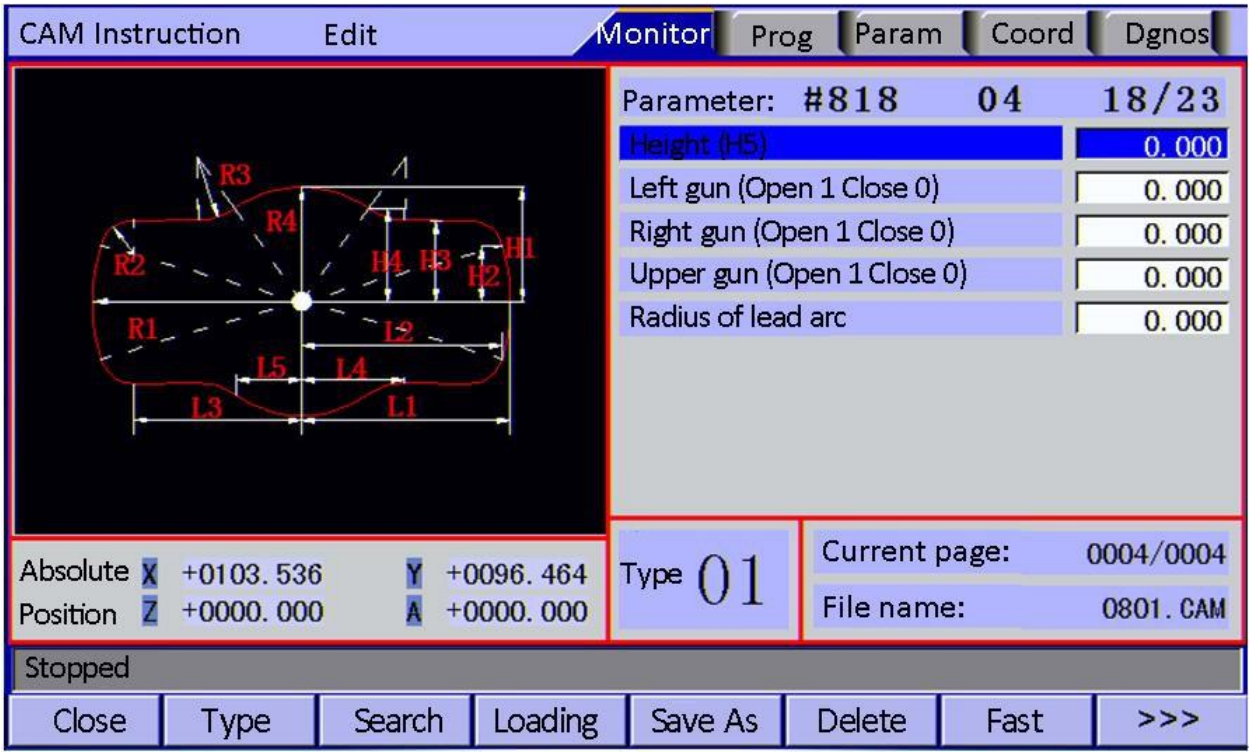

CAM Instruction
Edit
Monitor
Prog
Param
Coord
Dgnos
W
O
L
Parameter: #809 02 09/12
Upper gun (Open 1 Close 0) 0. 000
Radius of the lead arc 0. 000
Plane (G17G18G19) 0. 000
Absolute X +0103. 536 Y +0096. 464
Position Z +0000. 000 A +0000. 000
Type 01
Current page: 0002/0004
File name: 0801. CAM
Stopped
Close
Type
Search
Loading
Save As
Delete
Fast
>>>


CAM Instruction
Edit
Monitor
Prog
Param
Coord
Dgnos
R
O
R
L
Parameter: #800 03 00/12
Oblong hole (G153) *********
Space X between the hole centers 0. 000
Space Y between the hole centers 0. 000
Space Z between the hole centers 0. 000
Space A between the hole centers 0. 000
Length of the square hole (L) 0. 000
Radius of oblong hole (R) 0. 000
Left gun (Open 1 Close 0) 0. 000
Right gun (Open 1 Close 0) 0. 000
Absolute X +0103. 536 Y +0096. 464
Position Z +0000. 000 A +0000. 000
Type 01
Current page: 0003/0004
File name: 0801. CAM
Stopped
Close
Type
Search
Loading
Save As
Delete
Fast
>>>


CAM Instruction
Edit
Monitor
Prog
Param
Coord
Dgnos
R
O
R
L
Parameter: #809 03 09/12
Upper gun (Open 1 Close 0) 0. 000
Radius of the lead arc 0. 000
Plane (G17G18G19) 0. 000
Absolute X +0103. 536 Y +0096. 464
Position Z +0000. 000 A +0000. 000
Type 01
Current page: 0003/0004
File name: 0801. CAM
Stopped
Close
Type
Search
Loading
Save As
Delete
Fast
>>>


CAM Instruction
Edit
Monitor
Prog
Param
Coord
Dgnos
R3
R4
R2
H4
H3
H2
H1
R1
L2
L5
L4
L3
L1
Parameter: #800 04 00/23
Oblong hole (G154) *********
Space X between the hole centers 0. 000
Space Y between the hole centers 0. 000
Space Z between the hole centers 0. 000
Space A between the hole centers 0. 000
Arc radius (R1) 0. 000
Arc radius (R) 0. 000
Arc radius (R3) 0. 000
Arc radius (R4) 0. 000
Absolute X +0103. 536 Y +0096. 464
Position Z +0000. 000 A +0000. 000
Type 01
Current page: 0004/0004
File name: 0801. CAM
Stopped
Close
Type
Search
Loading
Save As
Delete
Fast
>>>





5.6 Generation of Processing Programs

When the program is defining the G code function, user can decide whether to use the G code inside the system according to his own need. If there is no need to use the system default G code, user can write his own code in the M_FUNC_NC file. The relationship between the user's custom M code and the CAM instruction G

code are shown in the table 2.

G code corresponding to M code	G code corresponding to M code	G code corresponding to M code
G151 -> M1510	G161 -> M1610	G171 -> M1710
G152 -> M1520	G162 -> M1620	G172 -> M1720
G153 -> M1530	G163 -> M1630	G173 -> M1730
G154 -> M1540	G164 -> M1640	G174 -> M1740
G155 -> M1550	G165 -> M1650	G175 -> M1750
G156 -> M1560	G166 -> M1660	G176 -> M1760
G157 -> M1570	G167 -> M1670	G177 -> M1770
G158 -> M1580	G168 -> M1680	G178 -> M1780
G159 -> M1590	G169 -> M1690	G179 -> M1790
G160 -> M1600	G170 -> M1700	G180 -> M1800

The generated program is as follows:

```
O1618
G90 G54 G17
G00 Z0 A0
G00 X0 Y0
G150

#801 = -9999.000
#802 = 0.000
#803 = 0.000
#804 = 0.000
#805 = 0.000
#806 = 0.000
#807 = 0.000
#808 = 0.000
#809 = 0.000
#810 = 0.000
```

G151

#801 = 0.000

#802 = 0.000

#803 = 0.000

#804 = 0.000

#805 = 0.000

#806 = 0.000

#807 = 0.000

#808 = 0.000

#809 = 0.000

#810 = 0.000

#811 = 0.000

G152

#801 = 0.000

#802 = 0.000

#803 = 0.000

#804 = 0.000

#805 = 0.000

#806 = 0.000

#807 = 0.000

#808 = 0.000

#809 = 0.000

#810 = 0.000

#811 = 0.000

G153

#801 = 0.000

#802 = 0.000

#803 = 0.000

#804 = 0.000

#805 = 0.000

#806 = 0.000

#807 = 0.000

#808 = 0.000

#809 = 0.000

#810 = 0.000

#811 = 0.000

#812 = 0.000

#813 = 0.000

#814 = 0.000

#815 = 0.000

#816 = 0.000

#817 = 0.000

#818 = 0.000

#819 = 0.000

#820 = 0.000

#821 = 0.000

#822 = 0.000

G154

G00 Z0 A0

G00 X0 Y0

M30

%

6. CAD DXF Conversion

6.1 Function

Before drawing, it is required to define the AUTOCAD processing layers, totally 16 layers; the layer names correspond to ADTLAYER1 to ADTLAYER16, and other layers can't be recognized by the system. The elements supported by the system contain point, line, arc, line segment, regular polygon, rectangle and circle, while other elements aren't supported by the system.

In DXF files, the drawn elements are classified into three types: point, line, including straight line, line segment, regular polygon and rectangle; arc, including arc and circle;

Template file is a script language file, which configures the DXF graphic files to generate different codes by modifying the script; its usage corresponds to DXF files. The name of template file is GTEMPLET.GT, which is saved in system directory ADT. After restarted every time, this file is loaded automatically; write and configure the template file with PC and copy to the system.

Format of template file

```
<HEADER>              //Template header
%
O0001
G54G90G17
<ADTLAYER 1 HEAD>          //Layer 1 head
T1M06                      //Tool change and other configuration
<POINT>                    //Point configuration
G00X<X>Y<Y>                //Point punching
<LINE>                     //Line configuration
G01X<X>Y<Y>D2              //Straight line punching
<ARCW>                     //Forward arc configuration
G02X<X>Y<Y>I<I>J<J>D2
<ARCI>                     //Reverse arc configuration
G03X<X>Y<Y>I<I>J<J>D2
<CUTTERBACK>               //Tool jump
G00X<X>Y<Y>                //Quickly move to the starting position
<ADTLAYER 1 HEADEND>       //Layer 1 end
G00X0Y0Z0                  //Resetting configuration
```

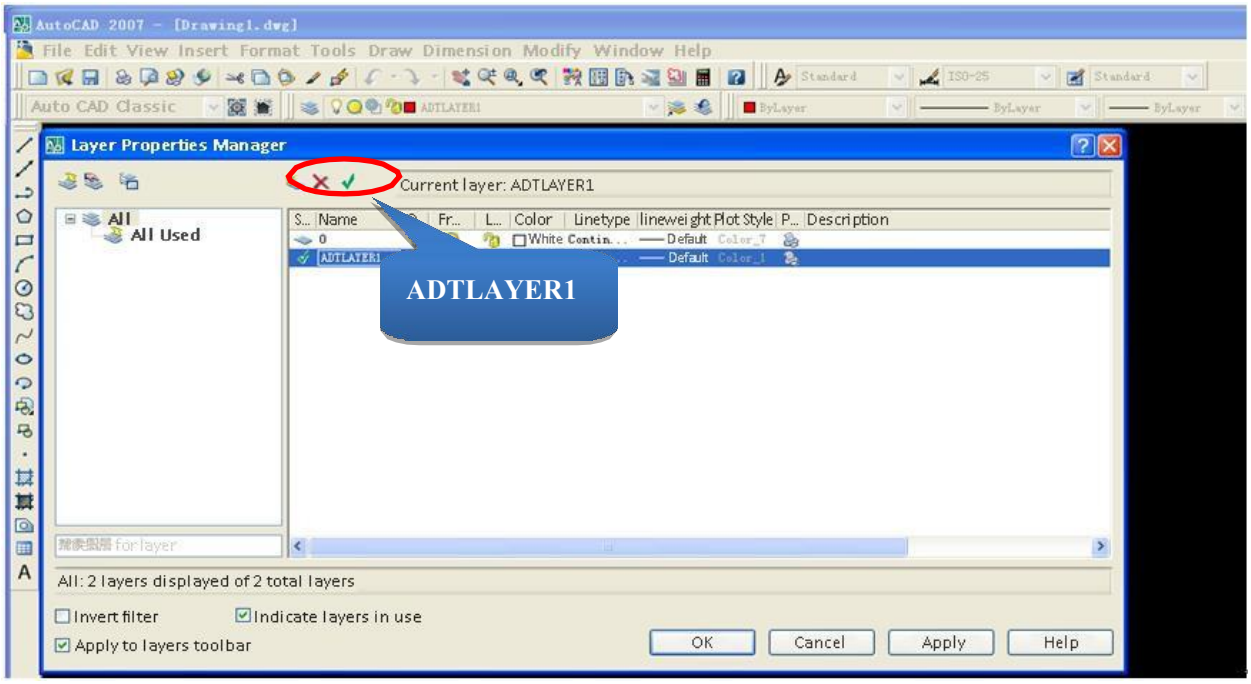
```
<END>                    //Template end
M30
%
```

6.2 Keywords description

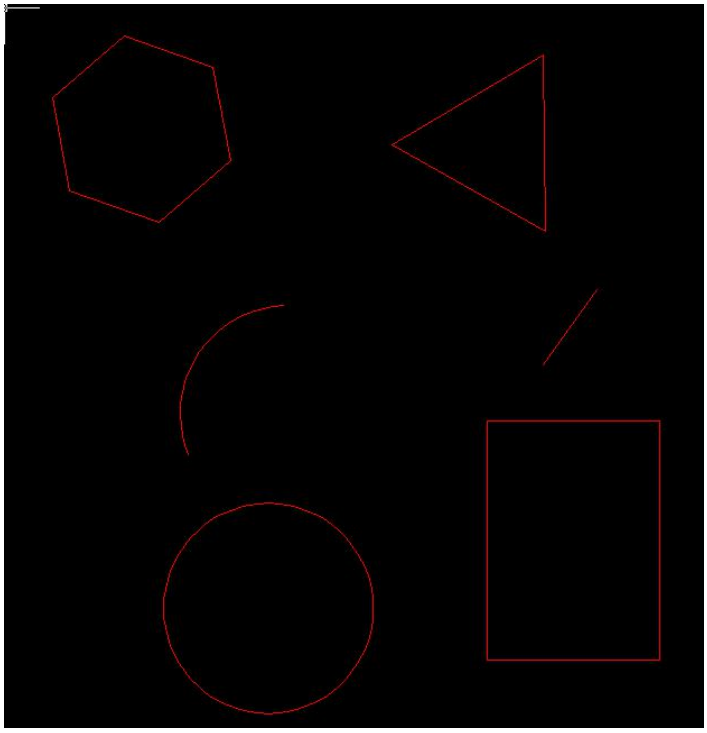
	Keyword	Description
Program header/end and process control	<HEADER>	Template header, used to configure program start, initialize code;
	<END>	Template end, used to configure end code of the program
Processing elements and the breakpoint configuration	<POINT>	Point configuration of current layer
	<LINE>	Line configuration of current layer
	<ARCW>	Forward arc configuration of current layer
	<ARCI>	Reverse arc configuration of current layer
	<CUTTERBACK>	Tool jump configuration of discontinuous point in current layer
Coordinate data configuration	<X>, <Y>	Configure point coordinates and end coordinates of the line
	<I>, <J>	Configure the offset of arc center relative to the starting point
Layer configuration keyword	<ADTLAYER 1 HEAD>	The head of layer 1, used to configure the initialization code of current level, such as tool change command
	<ADTLAYER 1 HEADEND>	The end of layer 1, used to configure the end code of current layer

6.3 Example

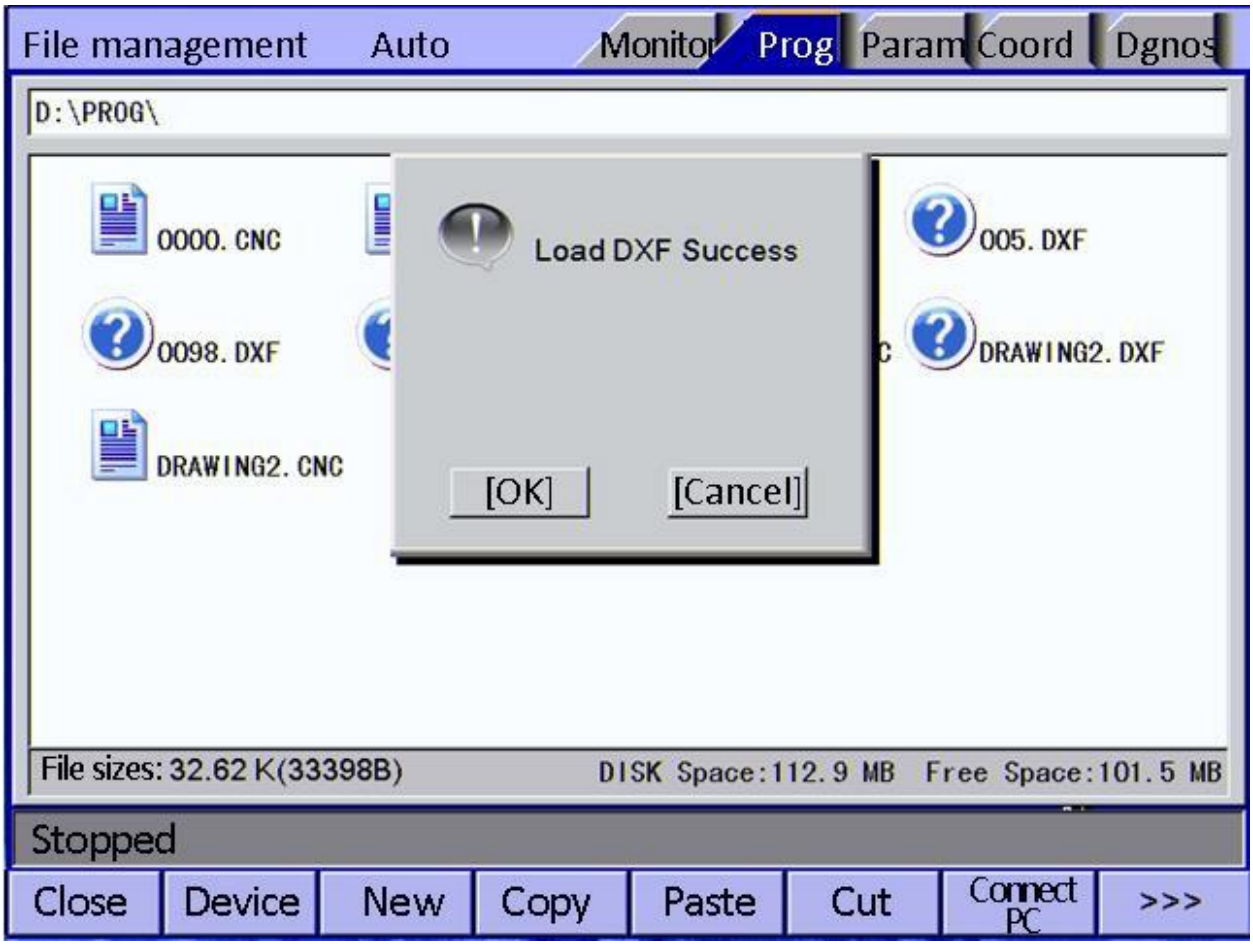
Here is an example of how to use CAD drawing and manual path editing with AUTOCAD2007. Open the AUTOCAD2007 software, create a new file, create a new graphic conversion layer "ADTLAYER1" and set the color to red, as shown below.



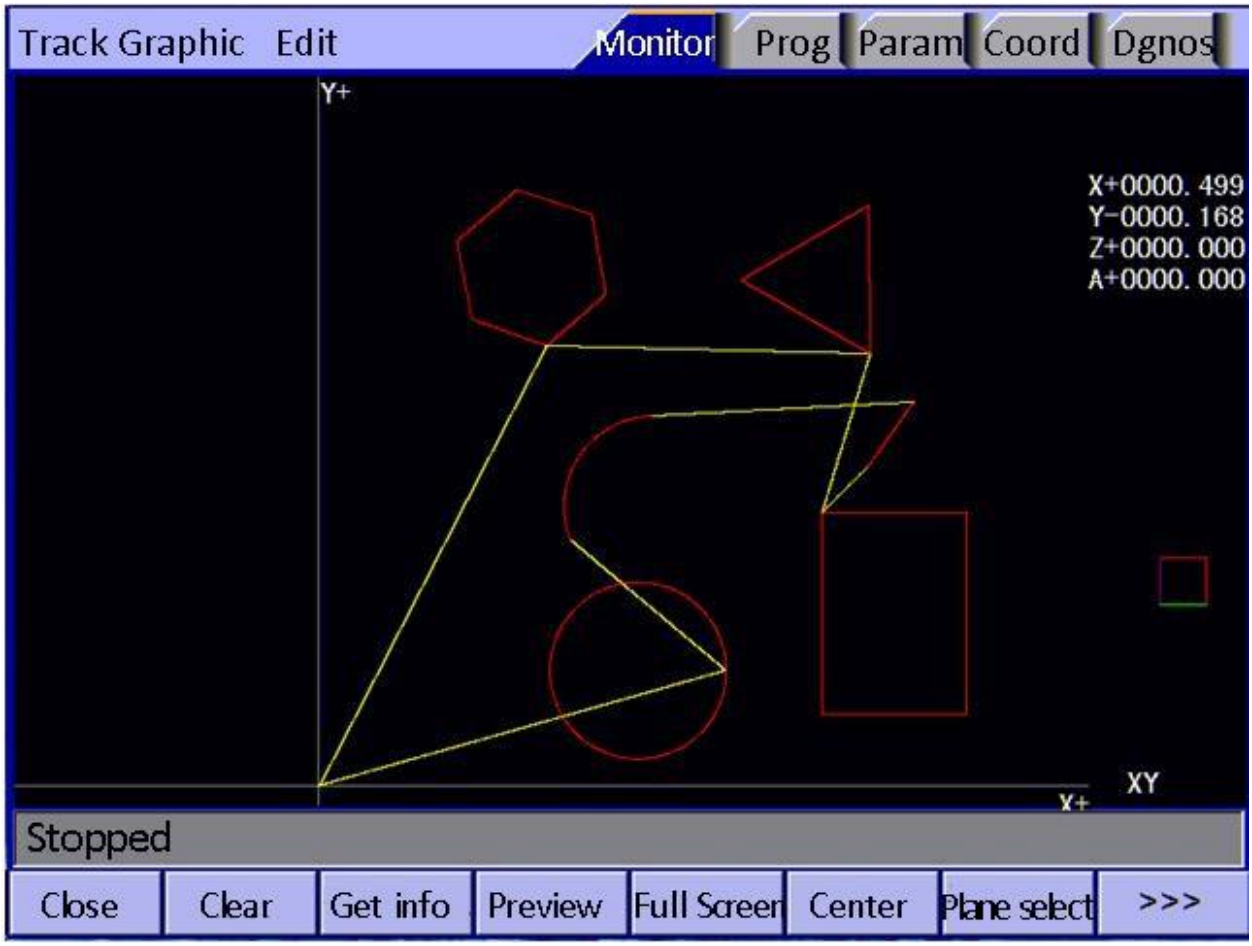
Select currently defined layer ADTLAYER1, and draw graphics such as point, line and arc. As shown in the figure below. After drawing, save as DXF file and copy to the system.



After drawing the graphics, click Save as the 2014DXF file in the File menu and copy it to the system. In the file management screen of the system, select the DXF file to be processed, the system will pop up a dialog box, press OK to complete the conversion of the DXF file, and convert the generated code file name to DXF file name with the suffix as ".CNC". Then load the file again. The figure above shows the Drawing2.dxf file. After selecting it in the file management interface, press the "EOB" button, and the following prompt screen pops up:

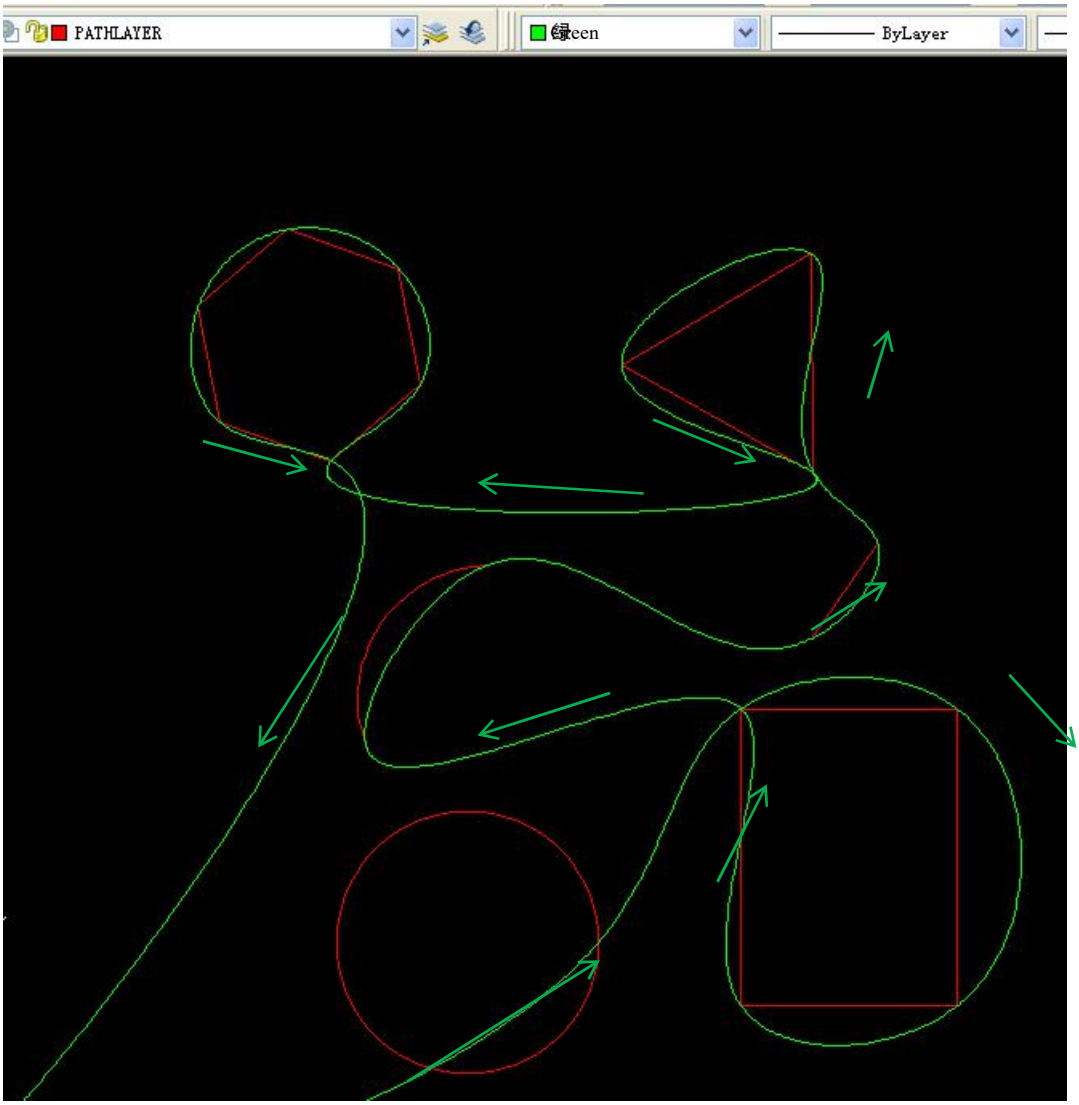


Press the “EOB” button again to confirm that the DRAWING2.CNC file is stored in the same directory after the conversion is completed. After selecting DRAWING2.CNC and loading the target code file, you can then preview the converted DXF file. The figure below shows the preview locus.

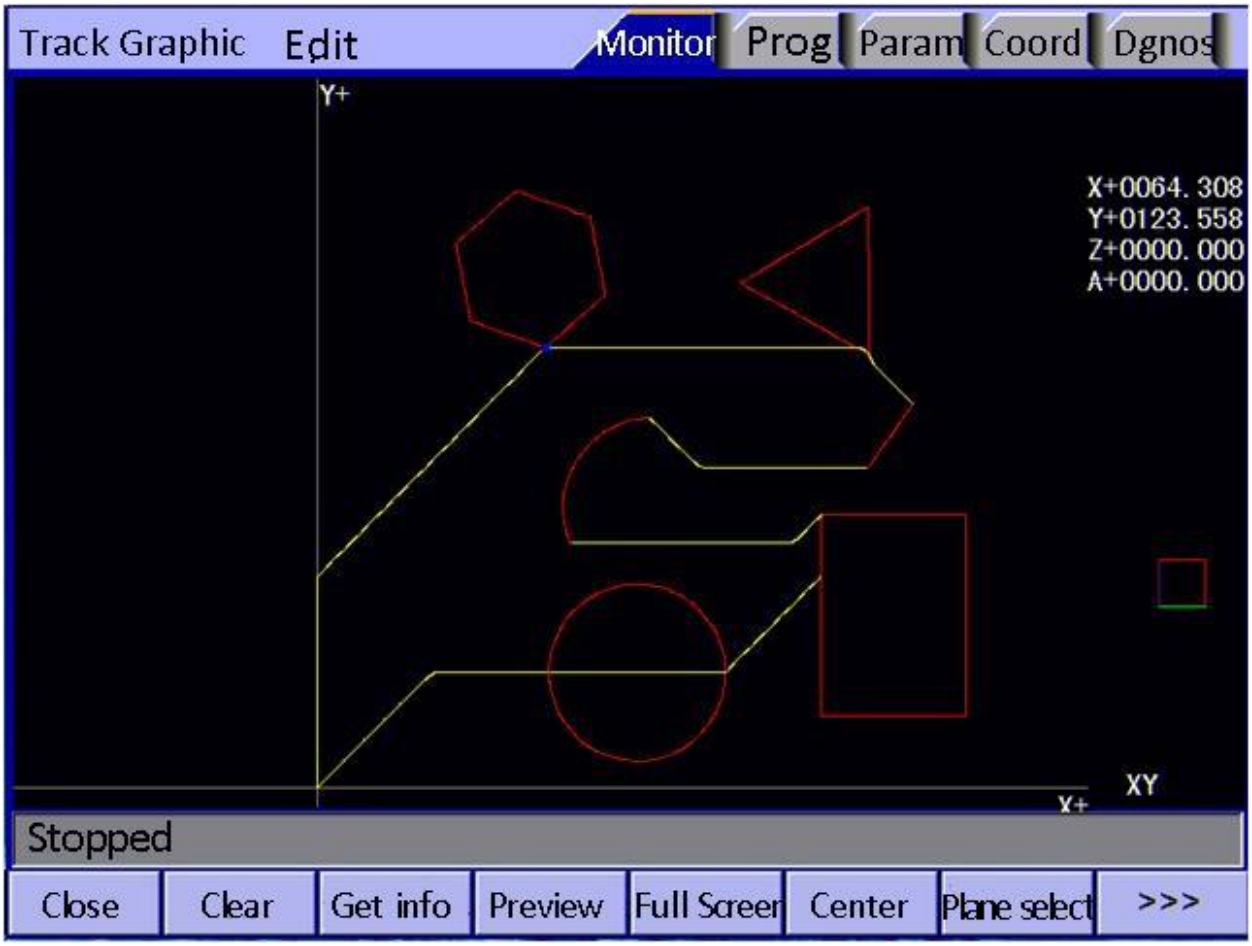


6.4 DXF file manual path processing

To make the reasonable planning of the movement path, improve the operation efficiency and increase the output, a path processing layer: "PATHLAYER" will be added. The processing method is as follows: first select the PATHLAYER layer, then select the spline curve button in the drawing toolbar, connect one by one from the starting point based on the planned path, and the last point is set to the safe position of the target retraction. Note: Polygon type, straight line, and the end point of the arc should be on the spline curve. As shown below:



After the planning, save the file into the DXF file of AUTOCAD2004 version. When loading the DXF file, the system will convert the G code file according to the spline connection order. The optimized path in the above example is shown below:



Additional note: DXF is only available to the AUTOCAD 2004 version and is only available for small and easy drawings. It may not be suitable for all graphics. For complex tracks, please choose another professional CAM software. The standard G code is generated and transmitted to the CNC system.

7. Automatic tool change (ATC)

Automatic tool change function is realized through manipulator (automatic tool change structure) and CNC system related control instructions. Taking armless tool magazine for example, the system diagram is shown below.

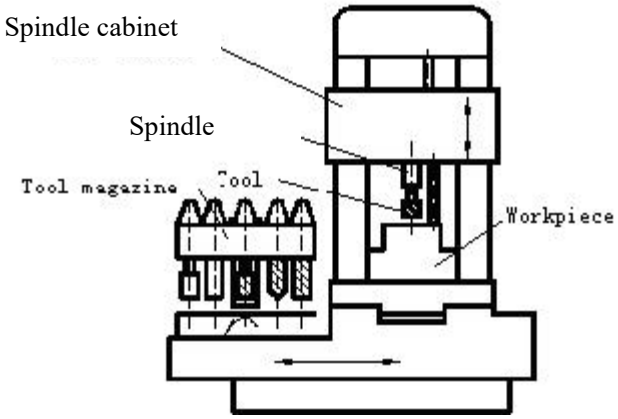


Fig. 14.1 Tool Magazine and Machine Tool Integrated CNC Machine Tool

The tool change can be realized with G code; edit T_FUNC.NC code, and select external tool magazine enable in the parameter; when the main program executes M06TXX tool change instruction, the system will call this program automatically, and send the tool number variable to tool change program to execute the programming of tool change.

The tool change process includes tool installation, selection and change. The spindle stops working, moves to tool change position to take out the tool, select tool in the tool magazine and install on the spindle position. To change tool, take out the tool from the spindle and put back to the tool magazine; the tool magazine should be moved to the position to receive spindle tool in advance.

Many methods are available for programming tool change, as described in the macro program below.

```
00123                                    (Program number)
G90 G599                                 (Use absolute programming after switching to
tool change, use G599 coordinate system, can't be used in processing file)
#201=#4121                               (read current tool number to #201)
IF[[#200] == 0]GOTO 100                  (#200 is the tool number to be changed; if the
changed tool number is 0, exit from tool change)
IF[[#200] == #201]GOTO 100               (if current tool is same to the tool to be
changed, exit from tool change)
```

```
IF[#400 > 24]                    (the system alarms if the maximum tool
number exceeds 24)
{ #3000
=1                               (Warning: set tool number exceeds the
maximum tool number of the tool magazine!) (system parameter 3001 alarm; alarm
content can be modified)
}
IF[[[#200] > [#400]] || [[#201] > [#400]]]
(alarm if the tool number to be changed and current tool number of the system
exceed the maximum tool number)
{ #3000
=1
(Warning: set tool number exceeds the maximum tool number of the tool magazine!)
(system parameter 3001 alarm; alarm content can be modified)
}
IF[#201==0]                      (alarm if current tool number is 0)
{ #3000
=1                               (current tool number zero error!)
}
G01 Z[#403+#404] F#405           (Z axis rises to a safe altitude)
M09                              (turn off cooling)
M89 P8 L1                        (output spindle quasi-stop signal)
M89 P13 L1                       (spindle blowing)
(G04 X#407                       (spindle blowing delay)
M89 P13 L0                       (turn off spindle blowing)
M88 P4 L0                        (wait for spindle quasi-stop in-place)
G01 X[#401] Y[#402]    F#406     (machine tool moves to X, Y axis
reference point)
IF[[#201]!=0]                    (check whether current tool is 0)
{                  (execute following codes if nonzero)
G01 Z[#403] F#405          (machine tool moves to Z axis reference point)
M89 P11 L1                       (output cutter exit signal)
M88 P6 L0                        (wait for exit in-place signal)
```

```
M89 P12 L1                          (output tool release signal)
G04 P300                            (delay 300 ms)
G01 Z[#403+2.5] F1000               (Z axis rises 2.5 ms, prevent pressing the
cutter during tool release)
M88 P9 L0                           (wait for tool release in-place)
G01 Z[#403+#404] F#405              (Z axis rises to a safe altitude)
}
#1=0                                (cutter forward/reverse rotation sign)
IF[#201 > [#400/2]] GOTO 1
IF[[#201 >= #200] || [#200 > [#201+[#400/2]]]] GOTO 2
M89 P9 L1                           (cutter forward rotation)
#1=0                                (the sign is 0)
GOTO 3
N2
M89 P10 L1                          (cutter reverse rotation)
#1=1                                (the sign is 1)
GOTO 3
N1
IF[[#201 >= #200 && #200 <= #400] && [#200 > [#201+#400/2]MOD#400]]
GOTO 4
M89 P9 L1
#1=0
GOTO 3
N4
M89 P10 L1
#1=1
N3
#2=#201                     (save current tool number in temporary variable)
WHILE[#2!=#200] DO1         (check whether equals to the tool number to
be changed)
M88 P7 L0                   (wait for cutter count signal becoming low)
M88 P7 L1                   (wait for cutter count signal becoming high)
IF[#1==1] GOTO 7            (check forward/reverse rotation through the sign)
```

```
#2 = #2+1       (forward rotation variable increases one tool position every time)
IF[#2>#400] #2=1                    (start counting from 1 if larger than tool number
of the system)
GOTO 8
N7
#2 = #2-1       (forward rotation variable increases one tool position every time)
IF[#2<=0] #2=#400                   (start counting from the maximum tool number if
smaller than 0)
N8
END1                                (cycle end keyword)
IF[#1==1] GOTO 5
G04 P#408                           (turn off forward rotation after delay)
M89 P9 L0                           (turn off forward rotation signal)
GOTO 6
N5
G04 P#409                           (turn off reverse rotation after delay)
M89 P10 L0                          (turn off reverse rotation signal)
N6
M89 P11 L1                          (output cutter exit signal)
M88 P6 L0                           (wait for exit in-place)
M89 P13 L1                          (spindle blowing turn on)
G04 X#407                           (spindle blowing delay)
M89 P13 L0                          (blowing off)
M89 P12 L1                          (spindle tool release signal on)
M88 P9 L0                           (tool release signal in-place)
G01 Z[#403+2.5] F#405               (Z axis moves to 2.5 ms above reference
point)
M89 P12 L0                          (spindle tool clamping)
G01 Z#403  F6000            (move Z axis to the reference point simultaneously,
preventing spindle clamping the cutter in the process of clamping)
M88 P10 L0                          (clamping in-place signal valid)
M89 P11 L0                          (cutter returns)
M88 P5 L0                           (exit in-place)
```

```
M89 P8 L0                          (spindle quasi-stop signal invalid)
G01 Z[#403+#404] F#405             (Z axis rises to a safe altitude)
#5223=#[409+#200]                  (set the tool setting value corresponding to
current tool number in the coordinate system, and realize the tool compensation
function of different lengths)
N100
M30
)%
```

Macro address description

#200 tool No. to be changed; #400 system maximum tool No.; can be customized

#4121 current system tool No.; #3000 macro program alarm address

#403 Z axis tool change reference point; #404 Z axis tool change safe altitude