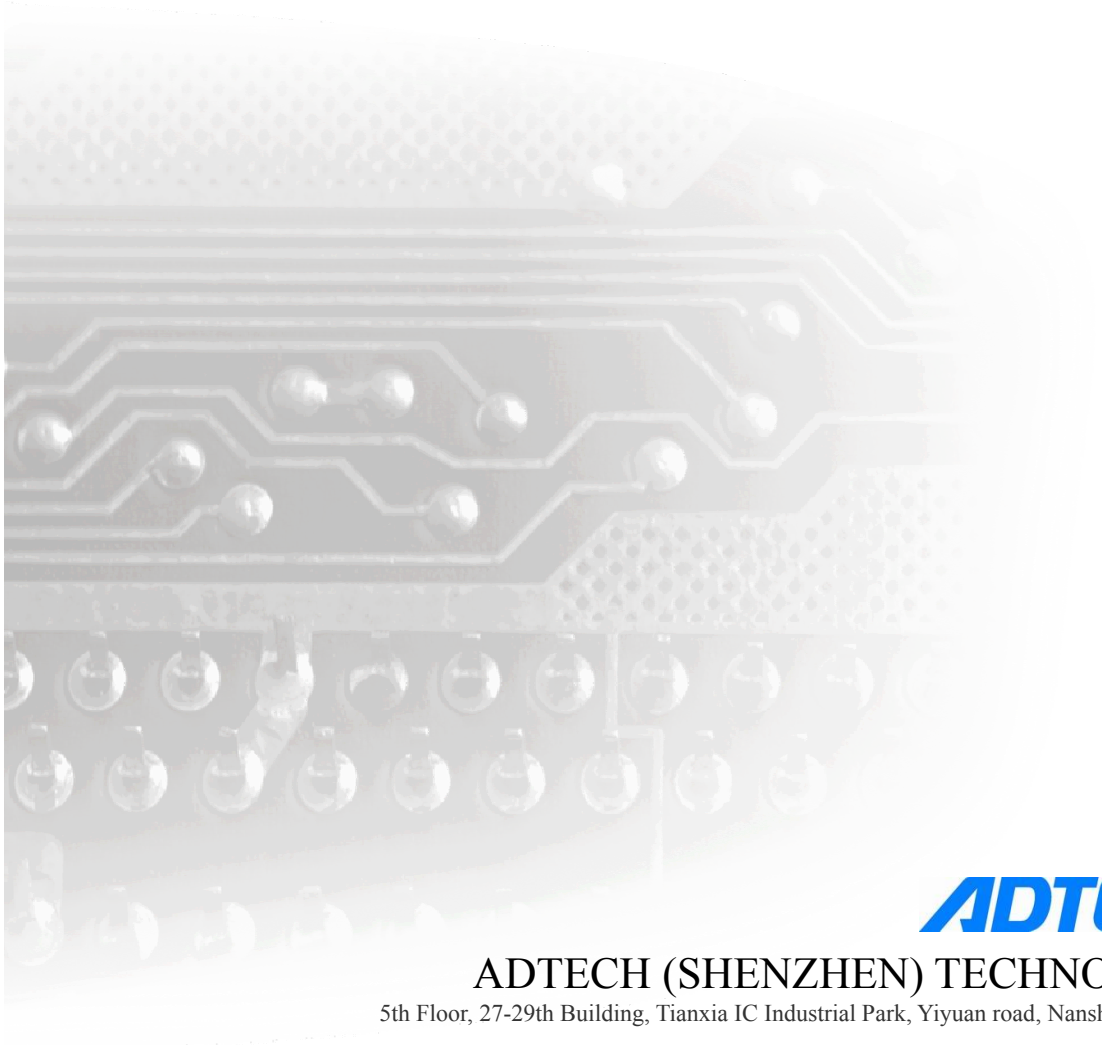
NCT-04/03/02

Punch CNC Systen (Programming)

User Manual



ADTECH (SHENZHEN) TECHNOLOGY CO., LTD
5th Floor, 27-29th Building, Tianxia IC Industrial Park, Yiyuan road, Nanshan District, Shenzhen Post code: 518052
Tel: 0755-26099116 Fax: 0755-26722718
E-mail: export@machine-controller.com www.machine-controller.com

Copyright Notice

Adtech (Shenzhen) Technology Co., Ltd.

Basic Information

Item No.	Initial uploading	Version No.	Total Page	Prepared by	Typeset by
BZ001B028A	2013-09-07	A0401	85	Zhang Qinggang	

Proofreading Records			
Date	Version/Page	Result	Confirmation

Precautions and Explanations

※Transport and storage:

- ☞ Do not stack product package more than six layers;
- ☞ Do not climb, stand on or place heavy stuff on the product package;
- ☞ Do not pull the cable still connecting with machine to move product.
- ☞ Forbid impact and scratch on the panel and display;
- ☞ Prevent the product package from humidity, sun exposure, and rain.

※Open-box inspection:

- ☞ Open the package to confirm the product to be purchased by you.
- ☞ Check damages situation after transportation;
- ☞ Confirm the integrity of parts comparing with the parts list or damages situation;
- ☞ Contact our company promptly for discrepant models, shortage accessories, or transport damages.

※Wiring

- ☞ Ensure the persons involved into wiring and inspecting are specialized staff;
- ☞ Guarantee the product is grounded with less than 4Ω grounding resistance. Do not use neutral line (N) to substitute earth wire.
- ☞ Ensure grounding to be correct and solid, in order to avoid product failures or unexpected consequences;
- ☞ Connect the surge absorption diodes to the product in the required direction, otherwise, the product will be damaged;
- ☞ Ensure the power switch is OFF before inserting or removing plug, or disassembling chassis.

※Overhauling

- ☞ Ensure the power is OFF before overhauling or components replacement;
- ☞ Make sure to check failures after short circuit or overloading, and then restart the machine after troubleshooting
- ☞ Do not allow to frequently connect and disconnect the power, and at least one minute interval between power-on and power-off.

※Miscellaneous

- ☞ Do not open housing without permit;
- ☞ Keep power OFF if not in use for a long time;
- ☞ Pay close attention to keep dust and ferrous powder away from control;
- ☞ Fix freewheel diode on relay coil in parallel if non-solid state relay is used as output relay. Check whether power supply meets the requirement to ensure not burning the control.
- ☞ Install cooling fan if processing field is in high temperature, due to close relationship between service life of the control and environmental temperature. Keep proper operative temperature range for the control: 0℃ ~ 60℃.
- ☞ Avoid using the product in the overheating, humid, dusty, or corrosive environments;
- ☞ Add rubber rails as cushion on the place with strong vibration.

※Maintenance:

Please implement routine inspection and regular check upon the following items, under the general usage conditions (i.e. environmental condition: daily average 30℃, load rate: 80%, and operating rate: 12 hours/ day)

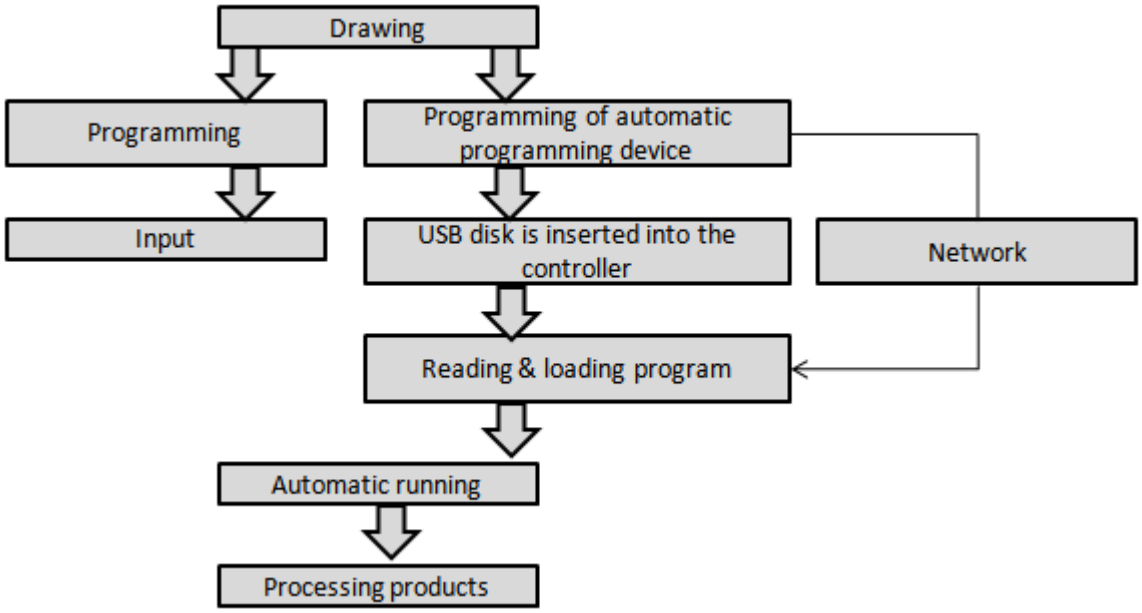
Routine Inspection	Routine	● Confirm environmental temperature, humidity, dust, or foreign objects. ● Confirm abnormal vibration and noise; ● Check whether vents are blocked by yarn etc.
Regular Check	One year	● Check whether solid components are loose ● Confirm whether terminal block is damaged

Contents

1. Programming Fundamentals

Flow from drawings to products

The flow from the products described in the drawings to the products through CNC turret punch is as follows:



Programming sequence

Determining processing methods

Determine the processing methods before considering the product shape, material size, thickness, product quantity and machinery.

[For example]

1. Cut the shapes with shearing machines and then process the materials with fixed size with punch, or reverse?

2. How many products can be processed with one portion of material when processing multiple small items? In addition, whether the shape is cut out with a punch or shearing machine?

3. When there are different shapes of products, how many kinds can be cut into with one piece of material?

Determining the position of workpiece clamp

● Please keep the workpiece clamping interval as large as possible.

● When using formed or deformed materials, please confirm that the clamp can re-clamp the workpiece (check if workpiece clamp is firm).

Note

● When programming, use the following method to prevent the workpiece clamp entering into dead point.

• Rotate the shape for 180° and grip the opposite end.

• Change the position of the workpiece clamp (interval).

• Change the mold turret position.

• Use the relocation.

Confirm the position number of necessary processing mold and installed mold.

Note

● The mold size and number that can be installed depend on the specifications of turret. If the position occupied by the large-diameter mold is not enough, retooling midway isn't as efficient as using nibbling and supplementary punching.

Determining processing sequence

While considering shortening the processing time and maintaining accuracy, please determine the order of processing positions, and number in different colors on the drawing.

Note

● General punching sequence is determined as follows.

• Rotate one circle starting from the top right of the drawings.

• From small holes to big holes, appearance, finally flanging and forming.

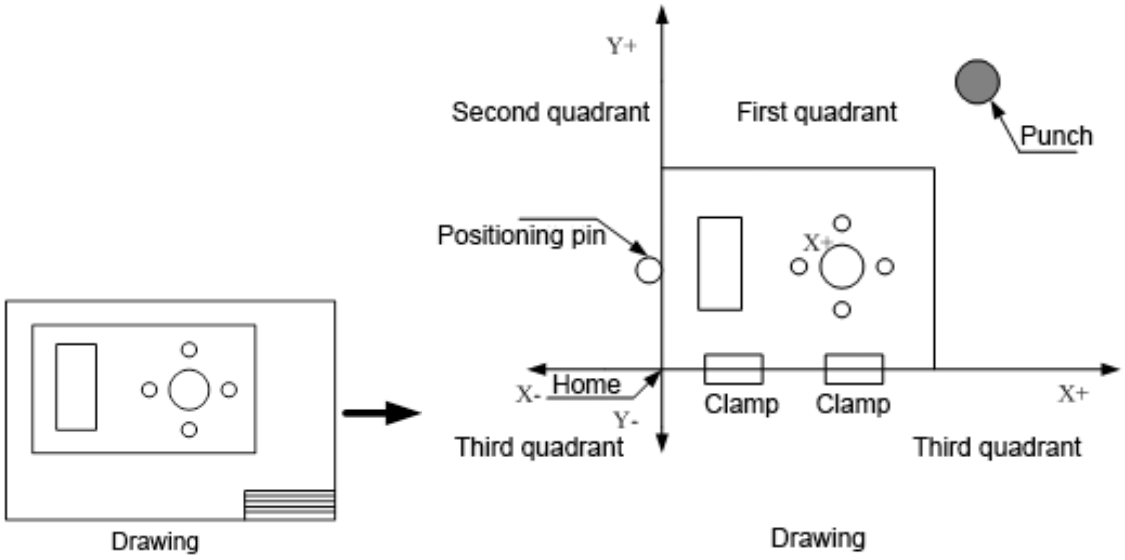
• Do not use the same mold more than 2 times. (However, relocation is the exception)

Calculating the coordinates

Please read the values in the drawing, and calculate the coordinates of the machining positions. Please measure the distance in X direction and the distance in Y direction from the origin (X0, Y0) in the processing positions (holes and other centers).

The clamped end of the material is the X axis. Please set the distance unit to "mm".

In the ±direction of the axis, place the drawing into the first quadrant of the "X-Y coordinate system" and measure from the base point.



● Set the right direction to "+ X" ● Set the left direction to "-X"

● Set the up direction to "+ Y" ● Set the down direction to "-Y"

Note

● The following two command modes are available to specify the machining position with the program.

• Absolute value command

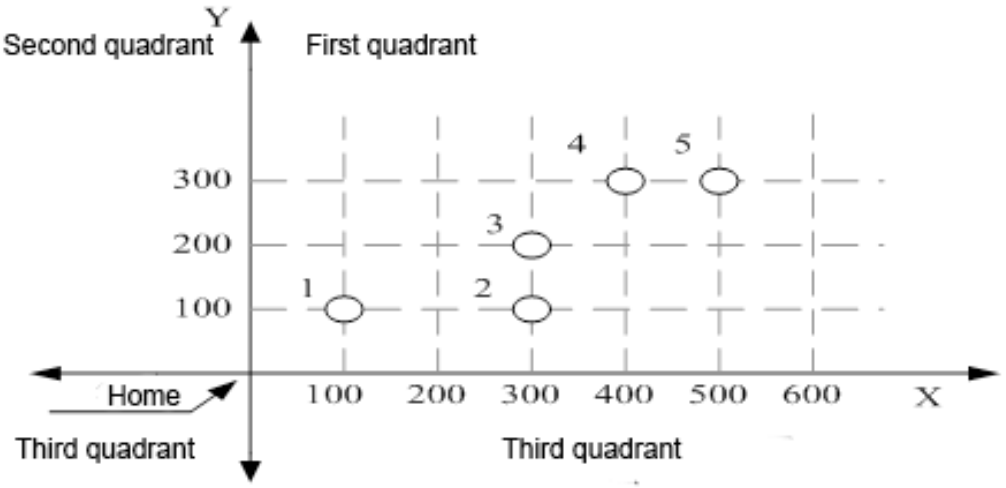
The distance measured from the origin (X0, Y0) is usually used to specify the position.

• Incremental command

Specify with the increment measured from the immediately preceding position rather than the origin.

[For example]

When specifying holes ① ~ ⑤ in the following figure, the absolute value command and the increment command are described in the following table.



Position of Hole Coordinate System

	Absolute value		Increment	
	X	Y	X	Y
①	100.00	100.00	100.00 (absolute value)	100.00 (absolute value)
②	300.00	100.00	200.00	0
③	300.00	200.00	0.00	100.00
④	400.00	300.00	100.00	100.00
⑤	500.00	300.00	100.00	0

2. Identifying the Machine Tool

2.1 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.

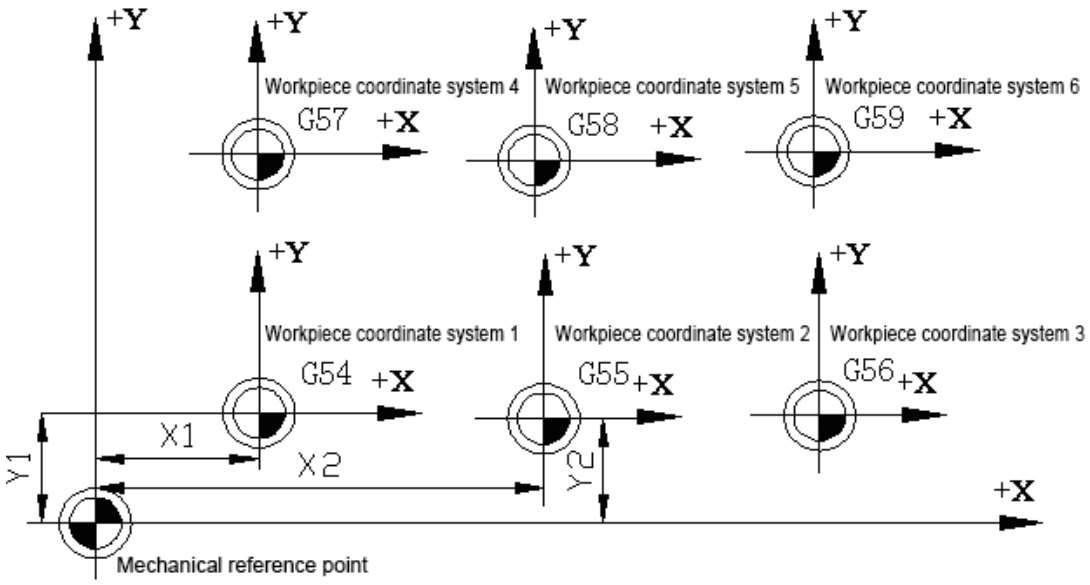


Fig. 2.3 Workpiece Coordinate System Diagram

3. Preparation Functions

3.1 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

G00X_;
Y_;
Z_; } G00 is valid in this range
G70X_;

3.2 Standard G Codes List

G Code	Name	Format	Page
*G00	Single punch	G00 X_ Y_	26
G04	Pause, accurate stop	G04 X_/P_	错误！未定义书签。
G26	Uniform circular hole	G26 I_J_K_	27
G27	Return to and check reference point		
G28	Return to reference point		
G29	Return from reference point		
*G54	Workpiece coordinate system 1		
G55	Workpiece coordinate system 2		

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	Macro program command		
G70	Quick positioning	G70 X_Y_	25
G72	Mode reference point setting	G72 X_Y_	26
G75	Relocation	G75 X_	
G76	Linear uniform holes	G76 I_J_K_	28
G77	Circular uniform holes	G77 I_J_P_K_	28
G78	Grid -X	G78 I_P _J_K_	29
G79	Grid -Y	G79 I_P_J_K_	29

G800	Single positioning punch	G800 X_Y_	43
G801	Linear fixed number punching / fixed pitch punching	G801 X_Y_Q_ or G801 X_Y_D_	43
G802	Arc fixed angle punching (CW)	G802 X_Y_I_J_Q__ or G802 X_Y_I_J_D_	44
G803	Arc fixed angle punching (CCW)	G803 X_Y_I_J_Q__ or G803 X_Y_I_J_D_	44
G804	Staggered punching compound command 1	G804 X_ Y_ R_ Q_ L_ D_ H_	46
G805	Arc punching compound command	G805 X_Y_R_Q_L_D_	46
G806	Linear punching compound command	G806 X_Y_Q_L_D_	46
G807	Peripheral circulation compound command	G807 X_Y_R_Q_D_	46
G808	Grid compound command Y	G808 X_Y_ R_Q_D_L_	46
G809	Grid compound command X	G809 X_Y_ R_Q_D_L_	46
G810	Staggered punching compound command 2	G810 X_ Y_ R_ Q_ L_ D_ H_	46
G811	Isosceles trapezoid processing compound command	G811 X_ Y_ P_ Q_ H_ R_ L_D_K_	46
G812	Rectangular trapezoid processing compound command	G812 X_Y_P_Q_H_R_L_D_I_K	46
G813	Grid specific direction processing compound command	G813 X_Y_P_Q_D_L_	46
G814	Circumference uniform holes compound command	G814 X_Y_R_P_Q_D_	46
G815	M processing compound command	G815X_Y _L_D_H_R_Q_	46
G816	Z processing compound command	G816X_Y_R_Q_H_I_D_L_	46

G817	Single pitch processing compound command	G817X_Y_R_Q_D_L_I_J_	46
G818	Double pitch processing compound command	G818X_Y_P_Q_D_L_I_J_E_H_	46
G86	Straight punching	G86 I _J _P _Q	30
G87	Rectangle command	G87 I _J _ P _ Q _	31
G88	Arc nibbling	G88 I _J _K_ P d_Q _	32
G89	Straight nibbling	G89 I _J _ P _ Q _	33
*G90	Absolute value programming		22
G91	Increment programming		22
G92.1	Machining origin setting		22

Note:

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

3.3 M Codes List

M Code	Function	Page
M00	Program stops	
M01	Program selection stops	
M03	Mold pin in	
M04	Mode pin out	
M05	Clamp tight	
M06	Clamp loose	

M08	Main motor on	
M09	Main motor off	
M20	Lubrication on	
M21	Lubrication off	
M22	Clutch on	
M23	Stop at top dead center, clutch off	
M30	Program ends and returns to the program header	
M98	Call subroutine	
M99	Subroutine ends, returns / repeats	
M56	Output No. 02 interrupt port high level	
M57	Output No. 02 interrupt port low level	
M58	Output No. 03 interrupt port high level	
M59	Output No. 03 interrupt port low level	
M10	Output No. 06 interrupt port high level	
M11	Output No. 06 interrupt port high level	
M20	Output No. 07 interrupt port high level	
M21	Output No. 07 interrupt port low level	
M12	Output No. 08 interrupt port high level	
M13	Output No. 08 interrupt port low level	
M14	Output No. 09 interrupt port high level	
M15	Output No. 09 interrupt port low level	

M16	Output No. 10 interrupt port high level	
M17	Output No. 10 interrupt port low level	
M18	Output No. 11 interrupt port high level	
M19	Output No. 11 interrupt port low level	
M40	Output No. 12 interrupt port high level	
M41	Output No. 12 interrupt port low level	
M42	Output No. 13 interrupt port high level	
M43	Output No. 13 interrupt port low level	
M44	Output No. 14 interrupt port high level	
M45	Output No. 14 interrupt port low level	
M46	Output No. 15 interrupt port high level	
M47	Output No. 15 interrupt port low level	
M48	Output No. 16 interrupt port high level	
M49	Output No. 16 interrupt port low level	
M50	Output No. 17 interrupt port high level	
M51	Output No. 17 interrupt port low level	
M66	Output No. 20 interrupt port high level	
M67	Output No. 20 interrupt port low level	
M64	Output No. 21 interrupt port high level	
M65	Output No. 21 interrupt port low level	
M62	Output No. 22 interrupt port high level	

M63	Output No. 22 interrupt port low level	
M60	Output No. 23 interrupt port high level	
M61	Output No. 23 interrupt port low level	
M88 Pn Lm	Check if waiting for input IO (IN n) level signal is m (high/low)	
M89 Pn Lm Qt	Output OUT n, level is m, delay t ms before output	

4. CNC Program Structure

4.1 Program Structure

CNC processing program consists of the following parts:

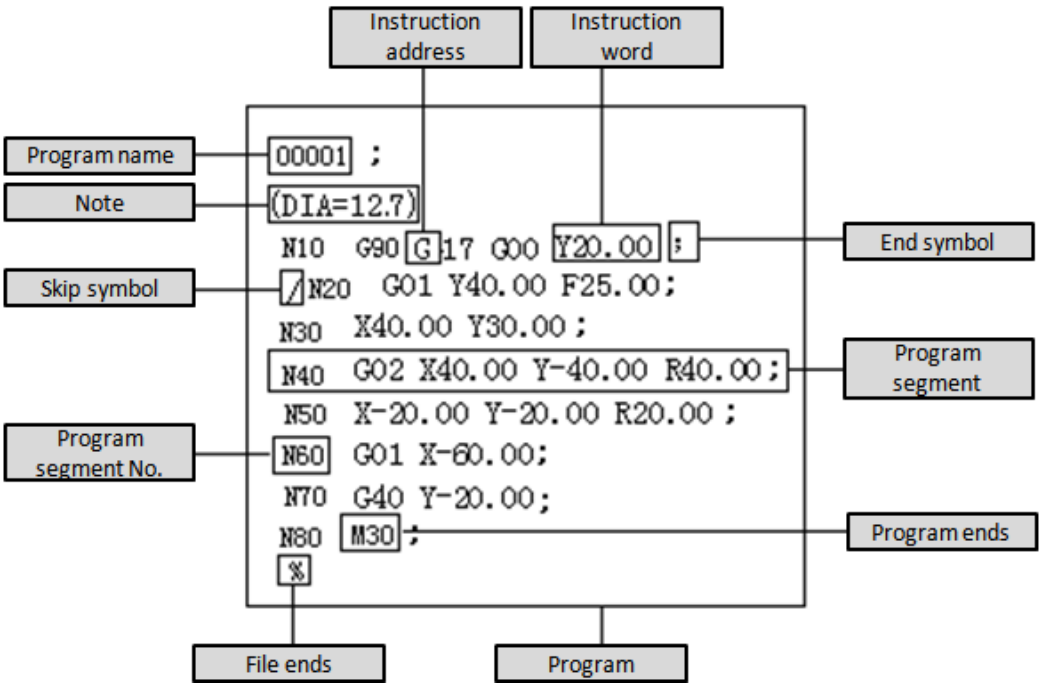


Fig. 4.1 CNC Program Structure Diagram

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
Principal axis 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	±99999.999mm	Coordinate position value
	R	±99999.999mm	Arc radius, corner radius
	I, J, K	±9999.9999mm	Arc center coordinate position value
Feeding rate	F	1~100,000 mm/min	Feeding rate
Principal axis rotation	S	1~4000rpm	Principal axis rotation
Select tool	T	0~99	Punch No.
Auxiliary function	M	0~99	Auxiliary function M code No.
Punch offset No.	H, D	1~200	Specify punch offset No.
Pause time	P, X	0~65sec	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 ±99999.999mm R is ±99999.999	Fixed cycle parameters

5. Position Instructions

5.1 Programming Mode Instructions

Function:

Punch 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 punch 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;

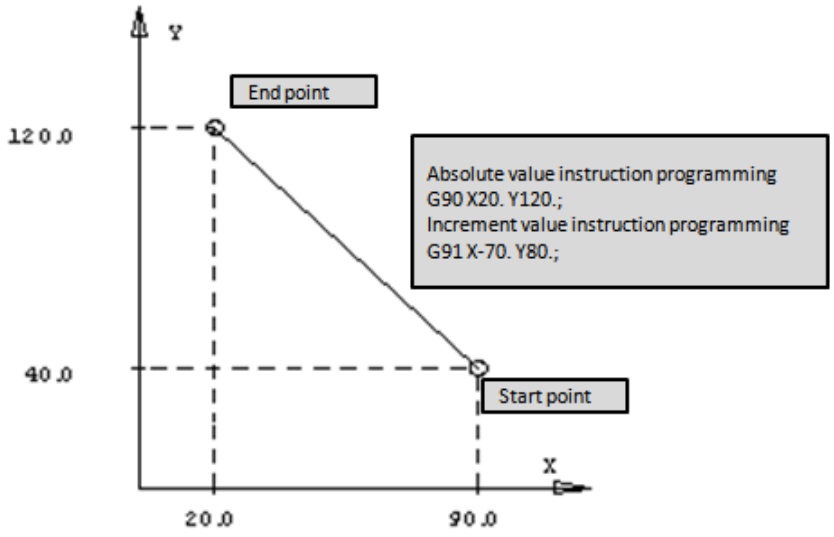


Fig. 1.1 Graphic Description Text

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.

6. Basic Script Processing Instruction Codes

G92.1 Origin Setting

After the machine tool is powered on, you must perform home operation automatically or manually and establish coordinate system basing on the reference point.

Function:

To specify the coordinates of processing position, you must decide the starting point of the coordinate system in advance; the specified values depend on the machine's specifications.

Format:

G92.1 X_ Y_ ;

Example:

If the X, Y stroke of the punch is 2500*1250, G92.1 X2500, Y1270;

According to the above instructions, the origin "X0, Y0" of X, Y coordinate system is decided.

All the absolute values in the associated program in the coordinate system are valid.

G90 Absolute Value, G91 Increment

Function:

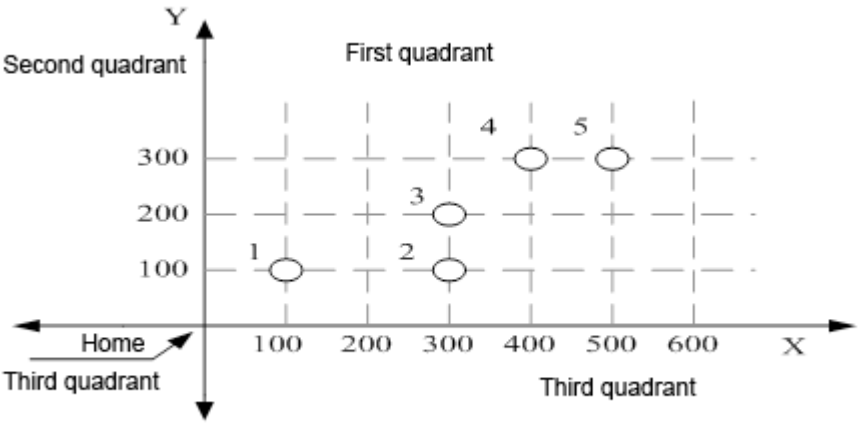
Absolute value (absolute coordinates) command and increment (relative coordinate, incremental coordinate) command are available to define the position of each axis. The position commands after absolute value command "G90" often designate the distance from the origin "X0, Y0".

The position commands after increment command "G91" specifies the movement of the block, i.e. the distance from the front position. "G90, G91" are also block headers.

Format:

G90 G91 X_ Y_ T_ ;

Example:



Position of Hole Coordinate System

	Absolute value		Increment	
	X	Y	X	Y
①	100.00	100.00	100.00 (absolute value)	100.00 (absolute value)
②	300.00	100.00	200.00	0
③	300.00	200.00	0.00	100.00
④	400.00	300.00	100.00	100.00
⑤	500.00	300.00	100.00	0

T Code: Mold Selection

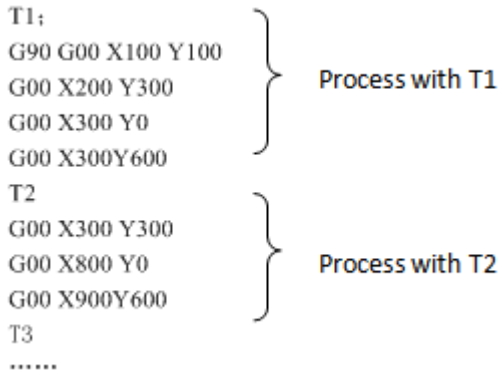
Function:

T-code 3-digit value specifies the mold position number. If the same mold is used continuously, the blocks do not need to repeat the instruction. Be sure to make a single block to instructions.

Format:

T_ (mold No.);

Example:



C Code: Sub-angle

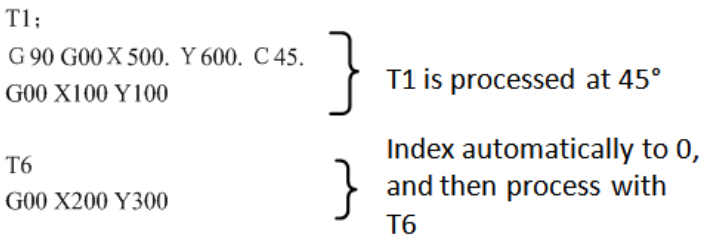
Function:

After X, Y, T command, subsequent C code value "±360.00" specifies indexing angle θ. Turning left (counterclockwise) is "+", and turning right (clockwise) is "-." Since this C code is also the "most common", each block doesn't need to instruct when continue processing with the same angle.

Format:

X_ Y_ C_ (Mold No.);

Example:



F Code: Shaft Speed Setting

Function:

The code setting X, Y-axis moving speed is the "most common". Speed command value is an integer. After the X, Y, C instruction, it also can be used as a separate block.

Format:

X_ Y_ C_ F_;

Example:

```
G 90 G54
G00 X 500.  Y 600.  C 45 F30000;   } Process at 30m/min
G00 X100 Y100;

G00 X200 Y300 F60000;              } Process at 60m/min
G00 X300;
```

Note

If the program doesn't have F command, it will process at default feeding speed;

G70: Stop Punching

Function:

If X and Y axes are moved only but not punched, do not interpolate while moving, and the speed is the rapid traverse speed of each axis. In the following cases, this instruction is used when the material and workpiece clamp are transferred to an appropriate location.

- When workpiece clamp approaches the mold, move to the next processing position;
- When the material is withdrawn from the workpiece clamping plate, try to relocate but still can't press down. "G70" is only valid for specified blocks.
- In addition, it can be used in the same block with "G90, G91"

Format:

G70 X_ Y_ ;

Example:

```
G90 G54
T1;
G00 X 500  Y 600;        ……Yes
G70 X100 Y100;           ……No
G00 X200 Y300 ;          ……Yes
```

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

G00: Single Punching

Function:

Achieve a punching action when the machine tool feeds to specified absolute position.

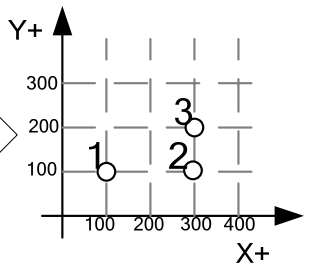
In addition, it can be used in the same block with "G90, G91"

Format:

G00 X_ Y_ ;

Example:

```
G90 G54
T1;
G00  100  100;
G00 X300 Y100;
G00 X300Y200;
M30;
```



Note:

G00 must be added before single punching;

G72: Mode Reference Point Setting

Function:

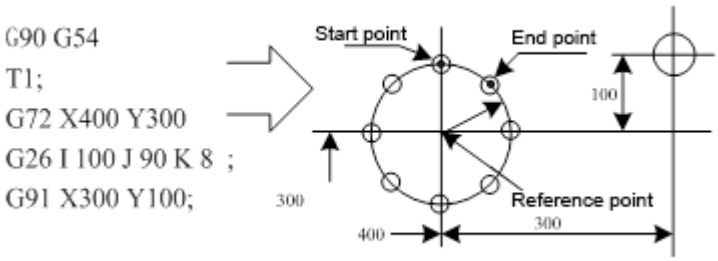
Set the reference point of the processing mode. Before specifying processing mode, please specify with separate block. If the "G72" command does not exist, use current position as the starting point to execute processing.

- Reference point position, either absolute value or increment.
- "G72" and "G90 (or G91)".
- Specify X, Y coordinates with "G72", does not determine the location or process.

Format:

G72 G90 (or G91) X_ Y_ ;

Example:



G26: Uniform Circular Hole

Function:

Divide the circumference equally and then process each point

Format:

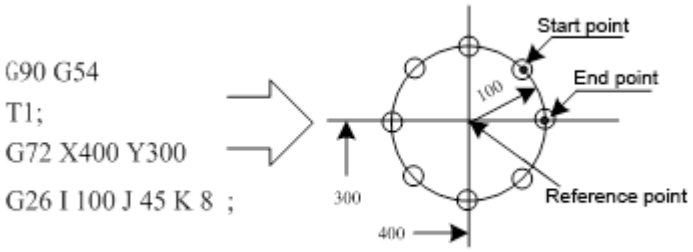
G26 I r J±θ K n

Ir: specify the circle radius r with a positive "mm" unit.

J±θ: specify the position of processing point with the angle ±θ against the X-axis direction. Counterclockwise rotation is "+", and CW rotation is "-".

K+n: Specify the number of processing points, i.e. specify the fraction + n.

Example:



G76: Linear Uniform Hole

Function:

Process several points equidistantly on the straight line

Format:

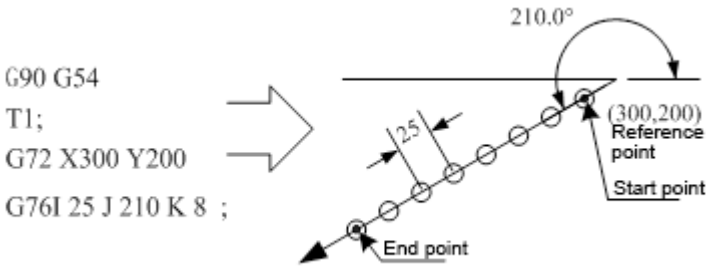
G76 I d Jθ K n

Id: Specify interval d (positive) of processing position with a positive "mm" unit.

Jθ: The angle between the straight line and +X-axis, in degrees. Angle is positive, and counterclockwise direction is positive.

Kn: Specify the number of processing points in positive number; do not include the mode reference point;

Example:



G77: Uniform Circular Hole

Function:

Process several points equidistantly on the straight line

Format:

G77I r JθPΔθK n

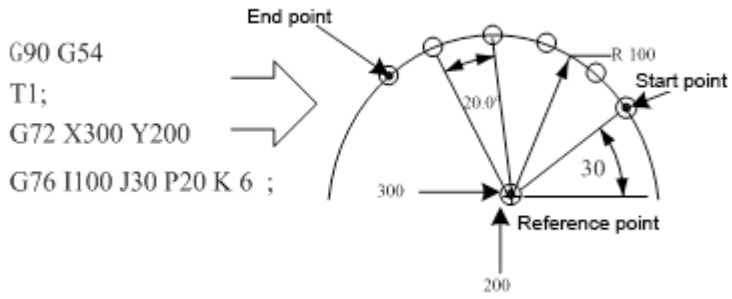
Ir: specify the circle radius with a positive "mm" unit.

J±θ: the angle between the first hole and +X-axis in the unit of degree. The counterclockwise is positive.

P ± Δθ: Specify the angular spacing; counterclockwise processing from the reference point is "+", and the clockwise processing is "-".

Kn: Specify the number of processing points in positive number; do not include the mode reference point;

Example:



G78: Grid-X

G79: Grid-Y

Function:

Process points in several rows equidistantly in X, Y direction. In "G78" and "G79", working directions are a different, but the ends are the same.

Format:

G78 I±d1 Pn1 J±d 2 Kn2

G79 I±d1 Pn1 J±d 2 Kn2

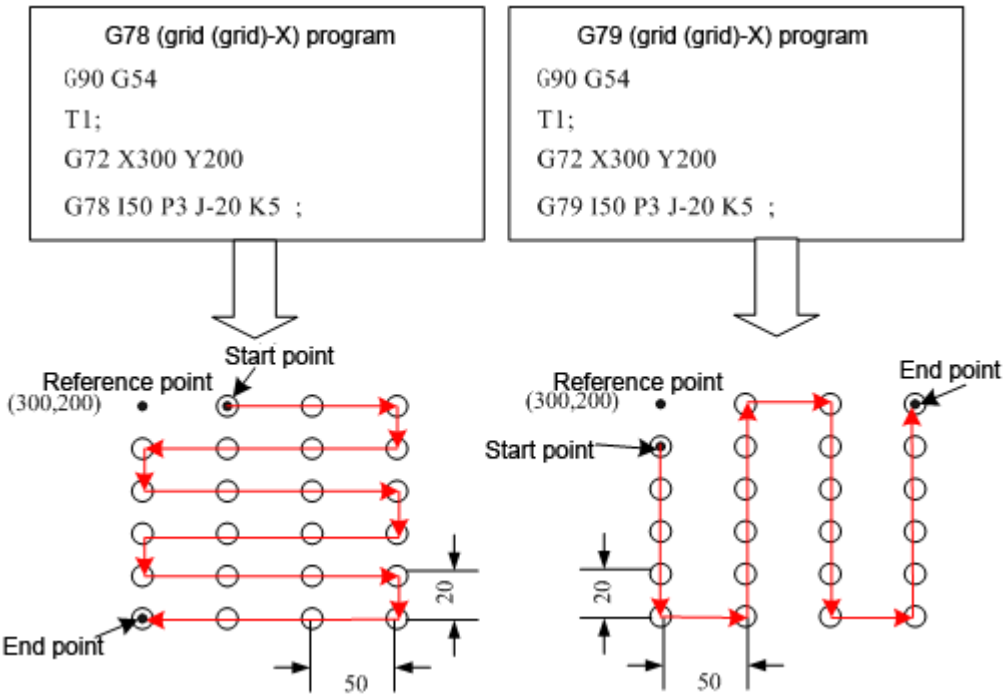
I±d1: specify with distance ±d1 in the X direction and "mm" unit. Processing from the reference point to +X direction is "+", and to -X direction is "-".

Pn1: specify the processing point n1 in X direction with positive number. Do not include reference point.

J±d2: specify with distance ±d2 in the Y direction and "mm" unit. Processing from the reference point to +X direction is "+", and to -X direction is "-".

Kn2: specify the processing point n2 in Y direction with positive number. Do not include reference point.

Example:



G86: Linear Punching

Function:

Supplement punching in the straight line with angle mold. The coordinate position is specified by punching shape rather than processing point (center of the mold).

Format:

G86 Iℓ J±θ P±w1 Q±w2 D±d

Iℓ: specify the punching length in "mm" unit on the straight line.

J±θ: specify the inclination of straight line with the angle θ against the X-axis direction. Counterclockwise rotation is "+", and CW rotation is "-".

P±w1: specify the mold punching amplitude w1 in "mm" unit (punch size in straight line direction is specified with "J±θ")

Processing the left side along the straight line from reference point is "+", and processing the right side is "-".

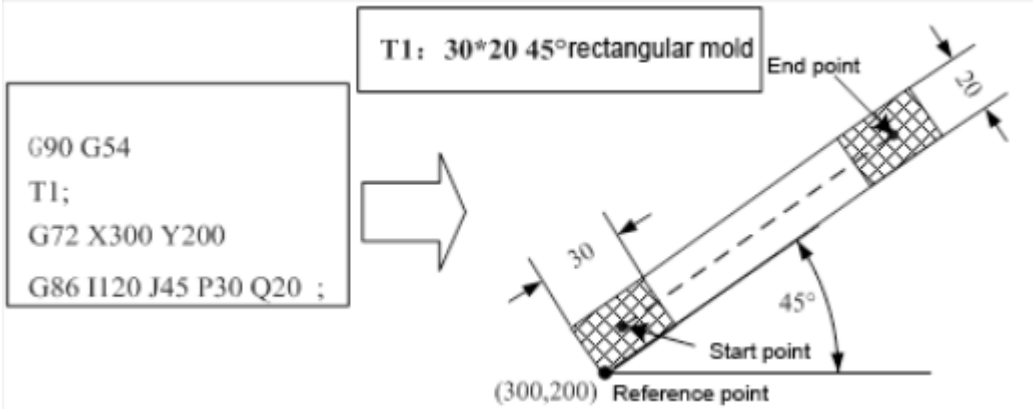
Q±w2: specify the mold punching amplitude w2 in "mm" unit (punch size in straight line and 90°direction is specified with "J±θ"). "±" must be same with w1.

Note:

w1 must have positive / negative sign; when the mold is a square, that is, w1 = w2, w2 can be omitted.

I ≥ 1.5W1

Example:



G87: Rectangle Command

Function:

Supplement punching rectangle in X, Y axis with square mold. The coordinate position is specified by punching shape rather than processing point (center of the mold). In addition, the supplementary punching pitch is set automatically according to the base of the mold size.

Format:

G87 I±ℓ 1 J±ℓ 2 P±w1 Q±w2 D±d

I±ℓ: specify the punching length ±ℓ1 in X-axis direction in "mm" unit. Processing from reference point in +X direction is "+", and processing in -X direction is "-".

J±ℓ: specify the punching length ±ℓ2 in Y-axis direction in "mm" unit. Processing from reference point in +Y direction is "+", and processing in -Y direction is "-".

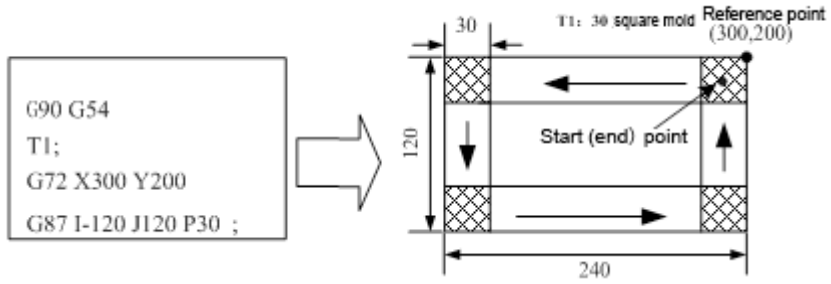
P±w1: specify the punching width (punch size) w1 in X-axis direction in "mm" unit;

Q±w2: specify the punching width (punch size) w2 in Y-axis direction in "mm" unit; in positive "mm units"; if "w1 = w2", the instruction can be omitted. Generally, Q code command is not executed because square punch is used.

Note:

“ℓ1 and ℓ2” must be at least three times of “w1 and w2”.

Example:



G88: Arc Nibbling

Function:

Punch specified angle equidistantly along with arc with round mold.

Format:

G88 Ir J±θ1 K±θ2 P±φ Qd

Ir: specify arc radius r in "mm" unit. Radius r must be greater than the punch diameter φ.

J±θ1: specify the position of initial processing point in the angle ±θ1 against the X-axis direction. The counterclockwise direction is "+", and the clockwise direction is "-".

K±θ2: Specify the arc angle ±θ2 of nibbling. Processing counterclockwise clockwise from the initial processing position is "+", and processing clockwise is "-".

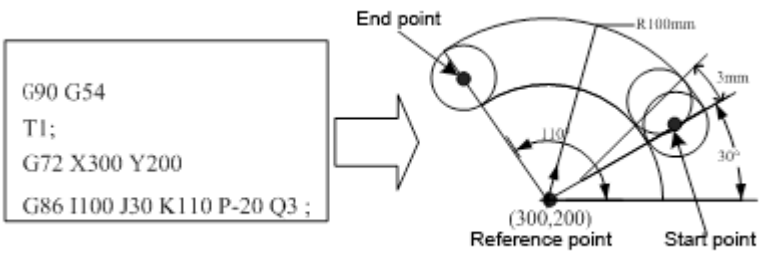
P±φ: specify punch diameter φ in "mm" unit. Processing the outside of the circle is "+", and processing the inside is "-". In addition, as "P0" command, process the punch center on the arc.

Qd: specify the equal space d on the arc in "mm" unit, which is limited by circular radius r.

Note:

"ℓ1 and ℓ2" must be at least three times of "w1 and w2".

Example:



G89: Linear Nibbling

Function:

Punch equidistantly for a given distance along the straight line

Format:

G89 Iℓ J±θ P±φ Qd

Iℓ: specify the distance ℓ from punch start to punch end for linear punching in "mm" unit. This is same to the distance between the reference points to the end point.

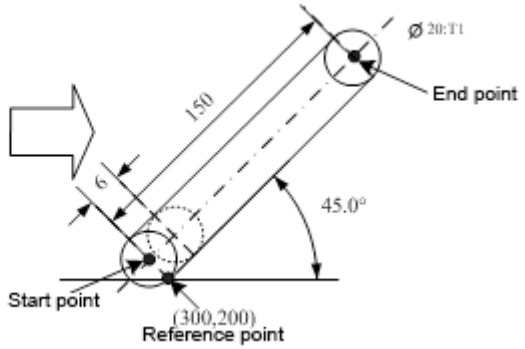
J±θ: specify the inclination of straight line with the angle ±θ against the X-axis direction. Counterclockwise rotation is "+", and CW rotation is "-".

P±φ: specify punch diameter φ in "mm" unit. Processing from the reference point to the punching direction is "+", and processing to the right is "-". In addition, as "P0" command, process the punch center on the arc.

Qd: specify the equidistance d on the straight line in "mm" unit.

Example:

```
G90 G54
T1;
G72 X300 Y200
G89 I150 J45 P20 Q6 ;
```



7. Relocation Command

To extend the range of processing in the X-axis direction, please use relocation (automatically change clamp) function. Change the clamp through the "G75" command. After changed, the processing position of X-axis becomes "command coordinates + offset".

G75: Relocation Command

Function:

Process the specified angle equidistantly along the arc with round mold.

Format:

G75X offset

Details:

The machine performs the following series of operations automatically in accordance with the instructions. Be sure to specify as a separate block.

- Relocation cylinder around the turret presses the material.
- When workpiece clamp opens, move away the material.
- Move back workpiece clamp (whole base) in Y direction for "2mm"; this parameter can be set by P1.097;
- Move the workpiece clamp (whole base) in the X direction for "X" amount.
- Move the workpiece clamp (whole base) forward in the Y direction for "2mm"; this parameter can be set by P1.098;
- Workpiece clamp closes and grasps the material.
- When the positioning cylinder lifts, move away the material.

Note

- The programming coordinates after relocated correspond with the actual board, and won't offset due to relocation; for example, to punch at 3000mm distance from the plate, it is necessary to relocate because it is beyond the range of processing, but the programming coordinate is still "X3000";
- If the movement is positive, X axis will move to the X-direction;
- Clamp forward/backward and X-axis travel speed are set by P1.121 parameter;
- The distance of clamp forward/backward is determined by parameter; if the clamp backward distance is greater than the forward distance, Y-axis direction will generate (backward - forward)

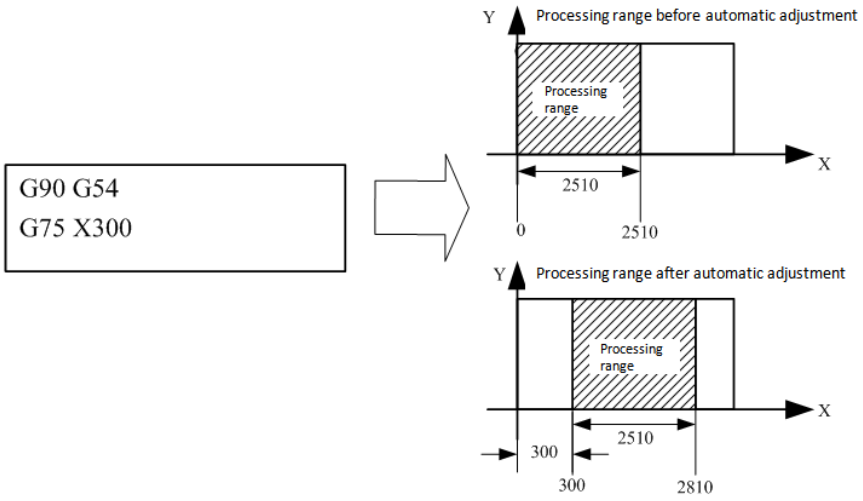
distance, and Y-axis command value can be entered as before clamp is changed because NC is calibrated automatically; but the Y-axis processing range will reduce the offset distance;

- Processing range after relocation:

The processing range after relocation only deviates for "X" amount in the X direction. Suppose a punch processing range is (0-2500), the processing range after running G75α is:

$(0+\alpha) \leq X \leq (2510+\alpha)$

Example:



Determination of movement:

Relocation movement is the largest X coordinates of the product minus the maximum movement of the machine in X direction.

Example:

The maximum X coordinate value of a product is "X3000", the X-axis maximum stroke is 2500, and the movement of relocating X-axis is:

3000-2500 = 500; "G27 X500" instruction. In this case, the processing range is: $(0 +500.) \leq X \leq (2510 +500)$

8. Coordinate System Setting Function (G52-G59, G591-G599, G92)

8.1 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.

8.2 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.

8.2.1 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)G92.1 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 punch 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 punch in original coordinate system and IP_ instruction value).

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

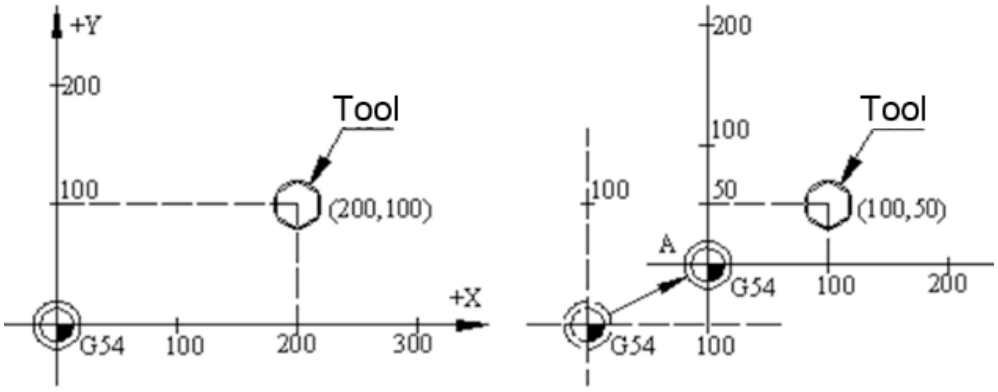


Fig. 8.1 G92 Instruction Function Diagram

8.2.2 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.

8.4 Operation 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.

8.4.1 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)

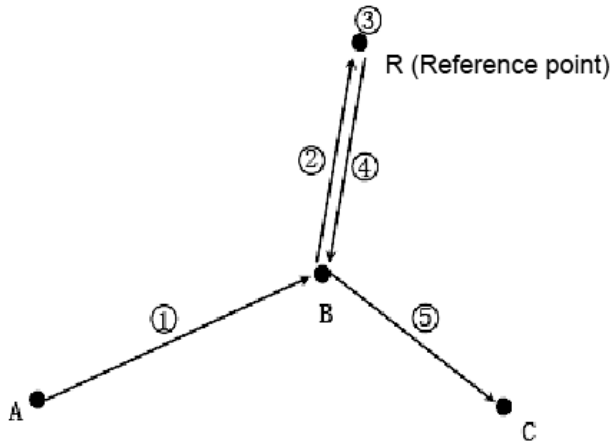


Fig. 8.2 Diagram of Automatically Returning to Reference Point

Note:

The coordinates of this center point are mainly used by G28 instruction.

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

8.4.2 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 Y Z 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:

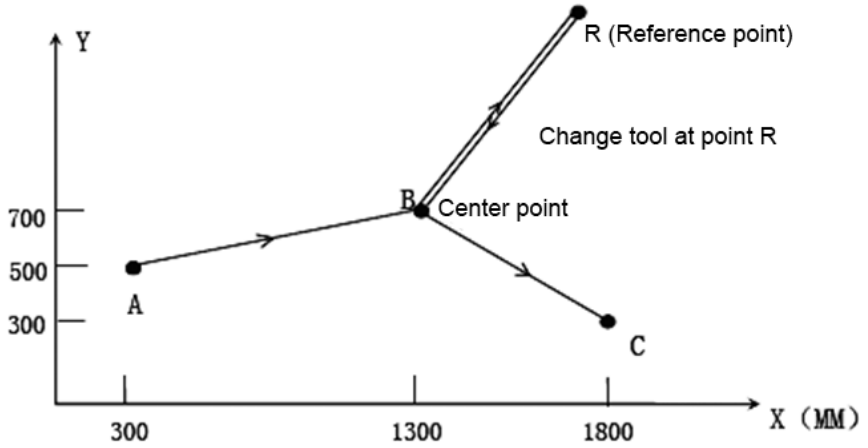


Fig. 8.3 G28, G29 Usage Diagram

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.

8.4.3 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 Y Z indicate 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.

9. CAM Compound Command Function

G800: Single positioning punch

Function:

Same as “G00”;

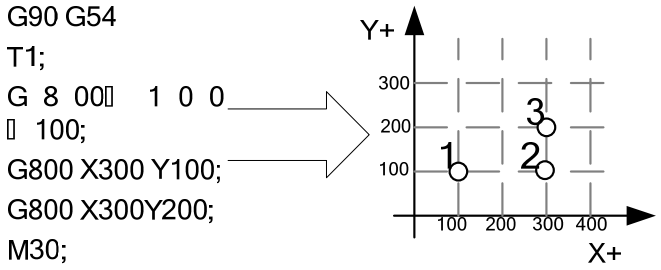
Achieve a punching action when the machine tool feeds to specified absolute position.

In addition, it can be used in the same block with "G90, G91"

Format:

G800 X_ Y_ ;

Example:



G801: Linear fixed number punching / fixed pitch punching

Function:

Punch equidistantly or in a set number from the starting point coordinates of a straight line to the ending point coordinates

In addition, it can be used in the same block with "G90, G91"

Format:

Set number straight punching: G801 X_Y_Q_;

Fixed pitch straight punching: G801 X_Y_D_;

Details:

X,Y- end position

Q- punching equal divisions Q>1

D- distance value of fixed pitch punching D>0

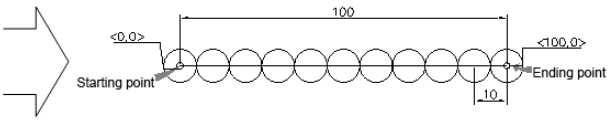
Note:

- Must have D or Q
- Starting point coordinate is the current coordinate position of the system;
- To specify the starting point coordinates, use G70 X_ Y_
- The first point is the punching point;
- Punch center is in the line of the starting point and ending point of the coordinates;

Example:

```
G70 X0 Y0;
G800 X 100 Y0 Q11;
M30;

G70 X0 Y0;
G800 X 100 Y0 D10;
M30;
```



G802/G803: Arc fixed angle punching

Function:

Punch equidistantly or in a set number along the arc direction

In addition, it can be used in the same block with "G90, G91"

Format:

Set number straight punching: G802 X_Y_R_Q_ or G802 X_Y_I_J_Q_

Fixed angle arc punching: G802 X_Y_R_D_ or G802 X_Y_I_J_D_

Details:

Set number arc punching: G802 X_Y_R_Q_ or

G802 X_Y_I_J_Q_

X, Y- end position

R- arc radius, R>0, program arc smaller than 180°, R < 0, program arc larger than 180°

I- start point X deviates X from the center

J- start point Y deviates Y from the center

Q- punching equal divisions

Fixed angle arc punching: G802 X_Y_R_D_ or

G802 X_Y_I_J_D_

X, Y- end position

R- arc radius, R>0, program arc smaller than 180°, R < 0, program arc larger than 180°

I- start point X deviates X from the center

J- start point Y deviates Y from the center

D- step value

Use G803 command for reverse circle; the format is same as “G802”;

Note:

- Must have D or Q
- Starting point coordinate is the current coordinate position of the system;
- To specify the starting point coordinates, use G70 X_ Y_
- The first point is the punching point;
- Punch center is in the arc of the starting point and ending point of the coordinates;

Example:

G804: Staggered punching compound command 1

G805: Arc punching compound command

G806: Linear punching compound command

G807: Peripheral circulation compound command

G808: Grid compound command Y

G809: Grid compound command X

G810: Staggered punching compound command 2

G811: Isosceles trapezoid processing compound command

G812: Rectangular trapezoid processing compound command

G813: Grid specific direction processing compound command

G814: Circumference uniform holes compound command

G815: M processing compound command

G816: Z processing compound command

G817: Single pitch processing compound command

G818: Double pitch processing compound command

10. Auxiliary Function

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 principal axis, 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……Mold pin in

M04……Mode pin out

M05……Clamp tight

M06……Clamp loose

M08……Main motor on

M09……Main motor off

M20……Lubrication on

M21……Lubrication off

M22……Clutch on (punch)

M23……Clutch off (stop at top dead center)

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, waits until IN0 is high voltage level, or else waits 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.

M00: Program stops

Function:

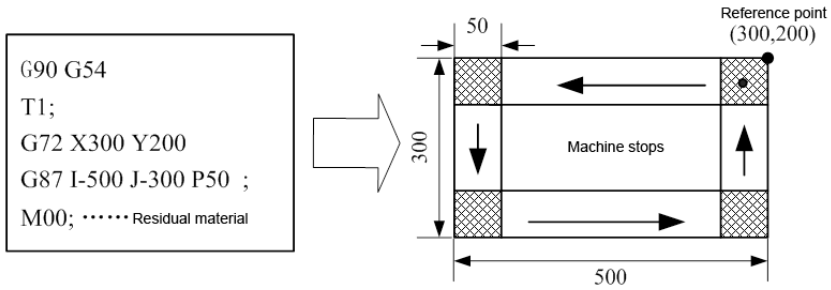
Pause the program instruction during processing; when running into "M00" command, the machine is paused; press the [Start] button to re-start the program;

Format:

M00

Example:

Supplement punching 300×500mm angle holes with 50×50mm punch; to pause the machine to remove residual material from the center:



M01: Program selection stops

Function:

Have the same functions as "M00", but this command is valid only when [M01] key on the additional panel is "ON".

[M01] key "ON" ····· reads "M01", and the machine stops.

[M01] key "OFF"·····ignore even reading "M01".

When "M01" is valid, the machine is paused; press [Start] button to re-start the program;

Format:

M01

M03: Mold pin in

Function:

When you run this command, the system outputs the mold insert signal to the port corresponding to P5.026 parameter;

M04: Mode pin out

Function:

When you run this command, the system outputs the mold insert signal to the port corresponding to P5.026 parameter;

M05: Clamp tight

M06: Clamp loose

Function:

When running "M05" command, the clamp will grip the plate;

When running "M06" command, the clamp is opened, and the plate can be moved;

Note:

- Clamp material corresponding output port: P5.016 parameter;

Example:

After processing and returning to the reference point, automatically open clamp and unload

G90 G54

T1 ……Use 1# mold

G 00 X100 Y100 ;…… Single punching

……

……

G91 G28 X0 Y0 . ……Return to reference point (mechanical home)

G04 P500 ……Delay 500ms

M06 ……Clamp opens and unloads

M30 ……Program ends and returns program header;

M08: Main motor on

M09: Main motor off

Function:

When running "M08" command, the main motor will start;

When running "M09" command, the main motor will shut down;

Note:

- Main motor corresponding output port: P5.022 parameter;

Example:

Program header automatically starts the main motor, and automatically turns off the main motor after processing;

G90 G54

M08 …… Start the main motor

G04 P3000 …… Delay 3 seconds to wait the motor for stable

T1 …… Use 1# mold

G 00 X100 Y100 ;……Single punching

……

……

G91 G28 X0 Y0 . ……Return to reference point (mechanical home)

G04 P500 ……Delay 500ms

M06 ……Clamp opens and unloads

M30 ……Program ends and returns program header;

M20: Lubrication on

M21: Lubrication off

Function:

When running "M20" command, the lubrication starts;

When running "M21" command, the lubrication stops;

Note:

- Lubrication corresponding output port: P5.024 parameter;

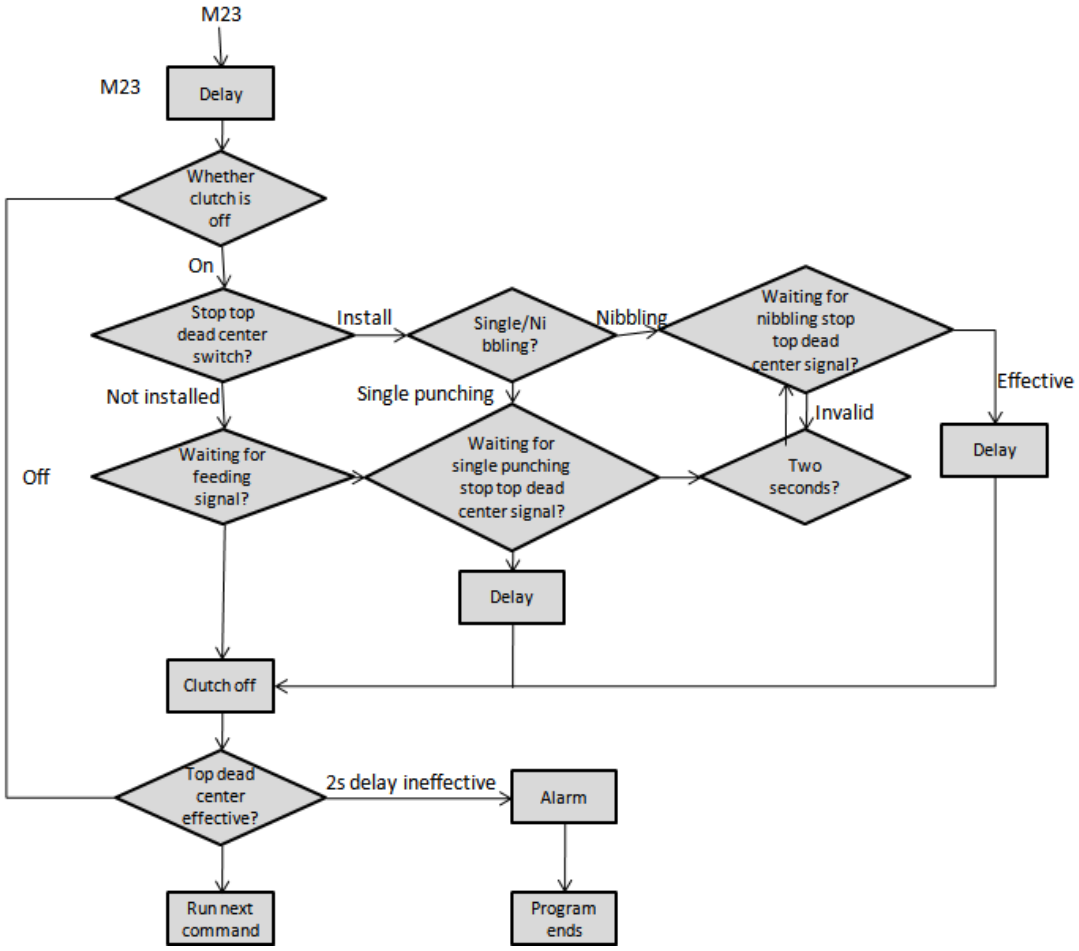
M22: Clutch on (punch)

M23: Clutch off (stop at top dead center)

Function:

When running "M22" command, clutch actuation signal outputs; flywheel and cam punch starts working;

When running "M23" command, the flow chart is:



Note:

- Clutch corresponding output port: P5.014 parameter;
- When port is assigned to P5.012 parameter, single punching stop top dead center switch is effective
- When port is assigned to P5.012 and P5.013 parameter, single punching and stop top dead center switch are effective
- If no port is assigned to P5.012 and P5.013 parameter, single punching and stop top dead center switch are ineffective, and the off clutch shares a signal with the feeding signal;

M30: Program ends and returns to the program header

Function:

At the end of the process, be sure to use a separate block. After executing the command, NC unit returns to the initial state.

Note:

This instruction cannot make each axis of the machine automatically returns to the reference point, Return to the reference point and end the program; may use the following commands;

G91 G28 X0 Y0 ..... machine tool returns to reference point

M30 ..... program ends and returns to the program header

At the end of the program, be sure to add the "M30" command;

Format:

M30

M88: input control

Function:

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.

Format:

M88 Pn Lm Qt

Details:

Check whether waiting for input IO (IN n) level signal is m (high/low) delay t ms

Pn: "n" corresponds to the input port number; 0: indicates IN0 corresponding input plate "1"

Lm: "m" represents the output level, 1: high level effective; 0: low level effective

Qt: "t" indicates the delay time; if effective signal is detected within this time, alarm and output;

Example:

M88 P0 L1	wait until IN0 is high voltage level, or else wait all along
M88 P0 L1 Q500	waiting IN0 is high; alarm if effective signal is detected within 500ms.

M89: output control

Function:

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.

Format:

M89 Pn Lm Qt

Details:

Output OUT n, level is m, delay t ms before output

Pn: "n" corresponds to the input port number; 0: indicates IN0 corresponding input plate "1"

Lm: “m” indicates output level; 1: high level effective, 0: low level effective

Qt: "t" indicates the delay time output; unit: ms

Example:

M89 P5 L0	specify OUT5 to output low voltage level immediately
M89 P5 L0 Q500	specify OUT5 to delay 500ms and output low level.

11. 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.

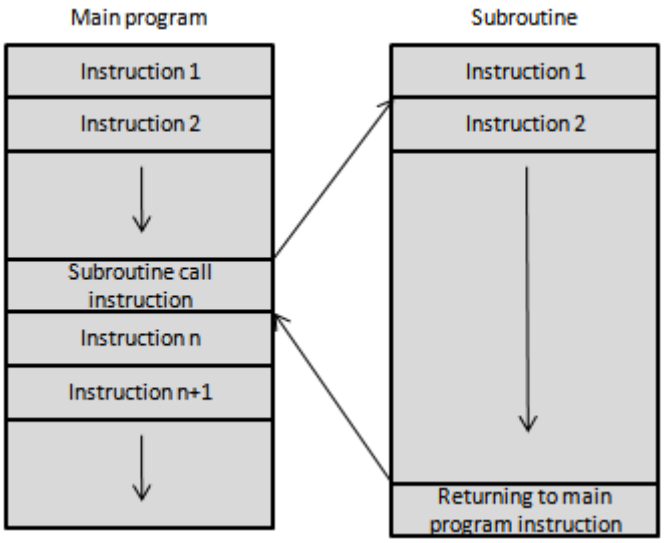


Fig. 4.2 Main Program and Subroutine

M99: Subroutine ends, returns / repeats

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.

M98: Call subroutine

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×××;

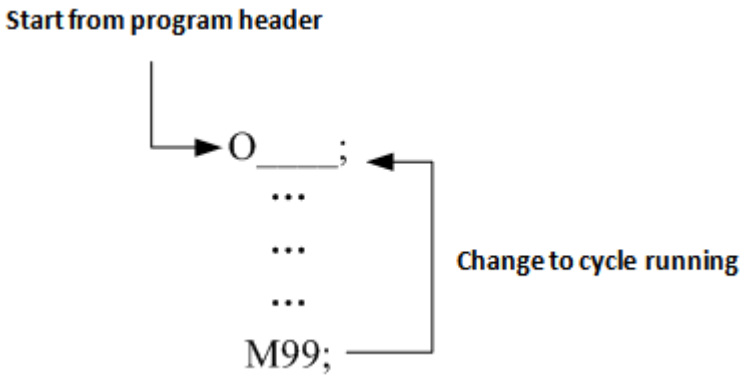
M98 P1234;…… …..P number not specified

M30;

This program only runs subroutine O1234 once;

Subroutine runs directly:

After the subroutine starts as is, it changes to cyclic operation; for safety, please make it stop in advance.

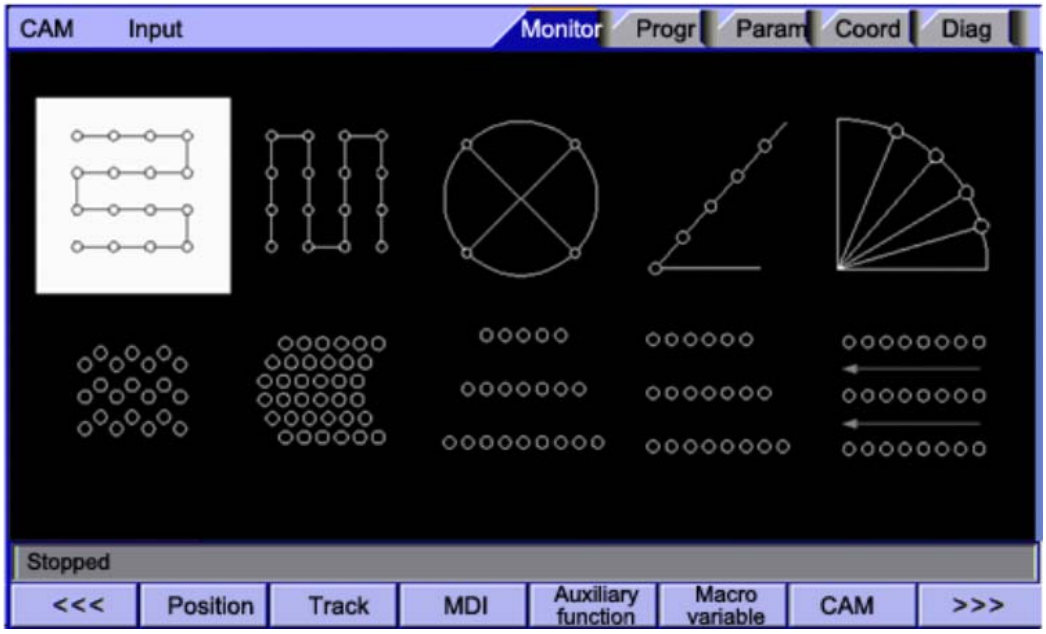


Note:

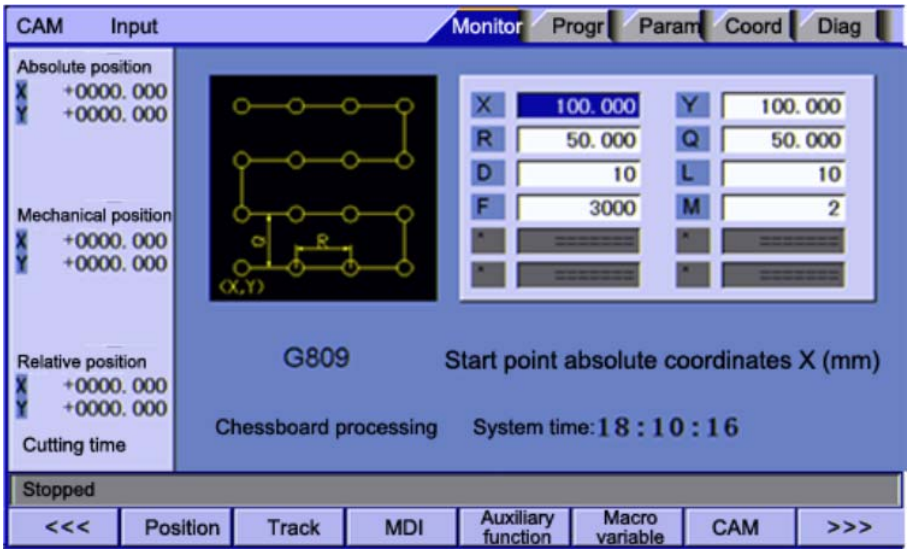
When performing subroutine directly (cyclic operation), please pay full attention to the processing position and circumambience conditions

12. CAM Function

System CAM function is intended for the processing of regularly distributed punching. The user only needs to set certain parameters, and the system can automatically calculate the position, generating the corresponding punching shape and reduce the input of each location. The main screen is as shown below



The graphics supported by the system include determinant punching (determinant also includes several type in accordance with the direction of the tool path and offset), equal round division, equal arc division, and equal line division. Enter the parameter setting screen by selecting the appropriate graphics; if the parameter is set correctly, the system can be started in automatic mode. The parameter setting of the first graph is as shown below:



Set starting point coordinates of graphics processing, step R in X direction, step Q in Y direction, punching number D in each line, total lines L, and home mode M, processing starts in the automatic mode.

13. CAD Function

CAD function enriches the programming of the system and improves the programming efficiency. The user only needs to design the graphics to be punched on PC with AutoCAD, and save as DXF file in the system, and then the system will convert DXF graphics file into G-code files for processing. G-code conversion is generated through the configuration file.

Before drawing, it is required to define the AUTOCAD processing layers. The layers other than ADTLAYER1 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:

(1) Point,

(2) Line, including straight line, line segment, regular polygon and rectangle;

(3) 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
M05           // Clamp open
[ADTLAYER 1 HEAD] // Layer 1 head
T1            // Tool change and other configuration
[POINT]             // Point configuration
G800X[X]Y[Y]     // Point punching
[LINE]                  // Line configuration
G801X[X]Y[Y]D2       // Straight line punching
[ARCW]                  // Forward arc configuration
G802X[X]Y[Y]I[I]J[J]D2
[ARCI]                  // Reverse arc configuration
G803X[X]Y[Y]I[I]J[J]D2
[CUTTERBACK]        // Tool jump configuration
```

G70X[X]Y[Y] // Quickly move to the starting position

[ADTLAYER 1 HEADEND] / Layer 1 end

G70X0Y0 // Resetting configuration //

[END] // Template end

G 91G28 X0 Y0

M06 //Clamp release

M30

%

Keywords in template:

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;

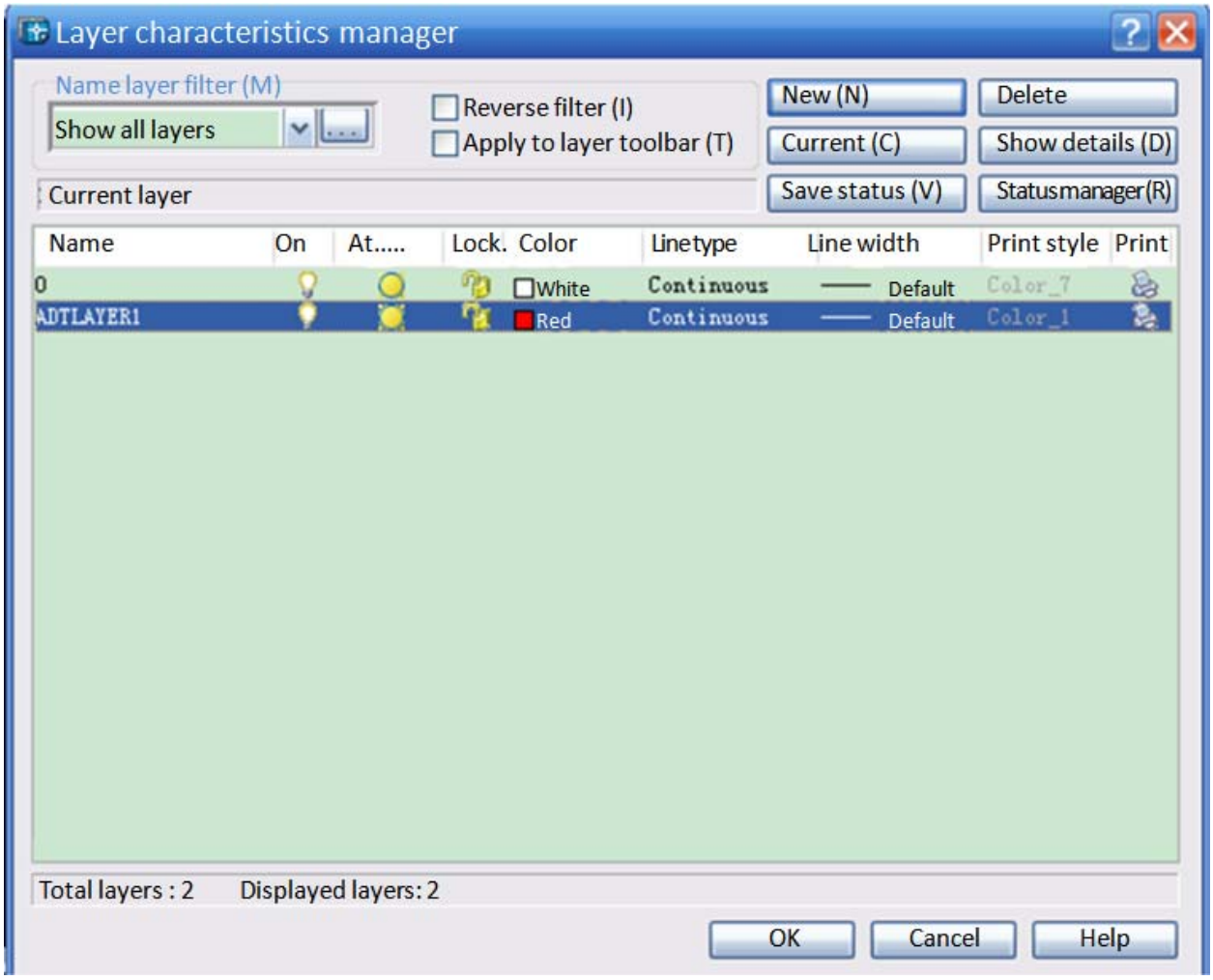
Layer configuration keyword:

[ADTLAYER 1 HEAD]: The head of layer 1, used to configure the initialization code of current level, such as tool change command;

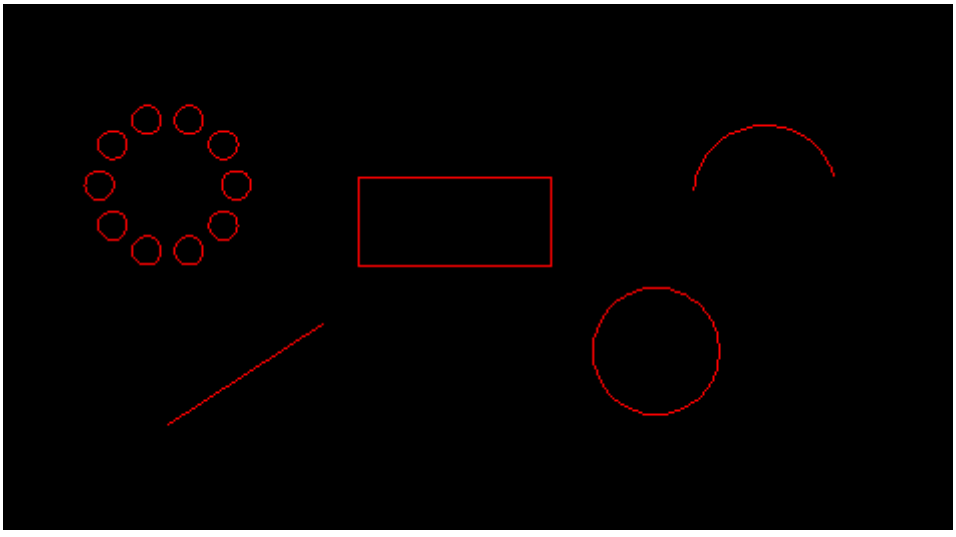
[ADTLAYER 1 HEADEND]: The end of layer 1, used to configure the end code of current layer;

CAD example

Start AUTO CAD and define the layer in layer management, as shown in the figure below. The layer name is defined as ADTLAYER1 and the color is set to red.



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.



Edit the configuration file; if current configuration is same as last configuration, it is not necessary to edit the template configuration file, and the system will save the last edited configuration file.

[HEADER] // Template header

%

```
O0001
G54G90G17
[ADTLAYER 1 HEAD] // Layer 1 head
T1M06                  // Tool change and other configuration
[POINT]                // Point configuration
G800X[X]Y[Y]           // Point punching
[LINE]                 // Line configuration
G801X[X]Y[Y]D2         // Straight line punching
[ARCW]                 // Forward arc configuration
G802X[X]Y[Y]I[I]J[J]D2    // Clockwise punching
[ARCI]                 // Reverse arc configuration
G803X[X]Y[Y]I[I]J[J]D2 // Counterclockwise punching
[CUTTERBACK]           // Tool jump configuration
G70X[X]Y[Y]            // Quickly move to the starting position
[ADTLAYER 1 HEADEND] / Layer 1 end
G70X0Y0                // Resetting configuration
[END]                  // Template end
M30
```

14. Category B Macro Function

14.1 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:

#△△△=○○○○○○○○○ or #△△△= [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]]

⑵Types of variables

Type	Variable address	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	

⑶ 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.

14.2 Macro Program Call

14.2.1 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).

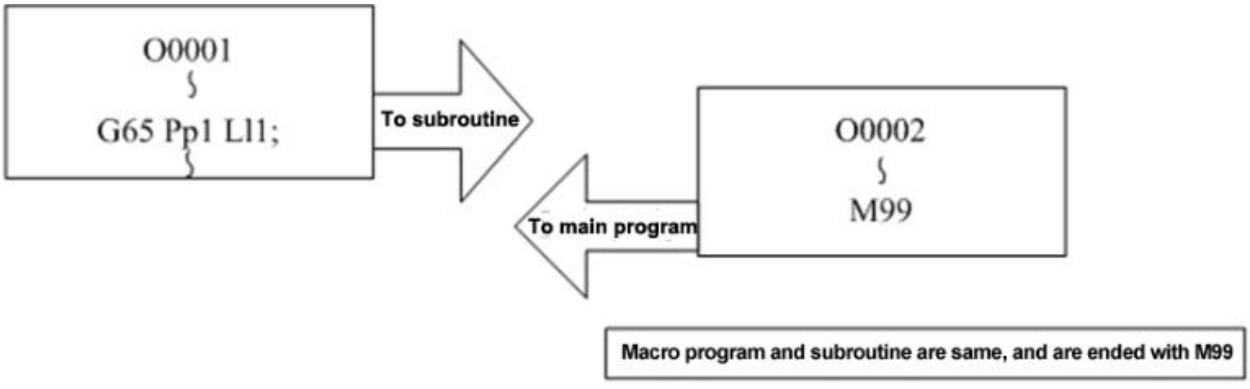
G66 (or G661) and G67 instructions must be used in pair in the same program.

14.2.2 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.

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:

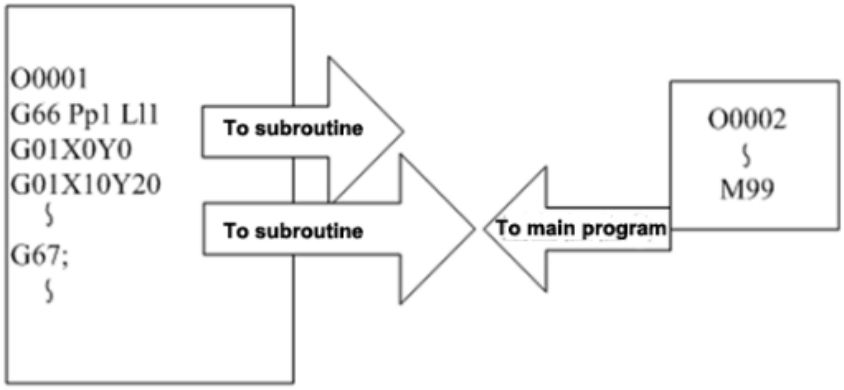
Table 12.2 Comparison between Argument Specified Bit Addresses and Local Variables

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

⑵ 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

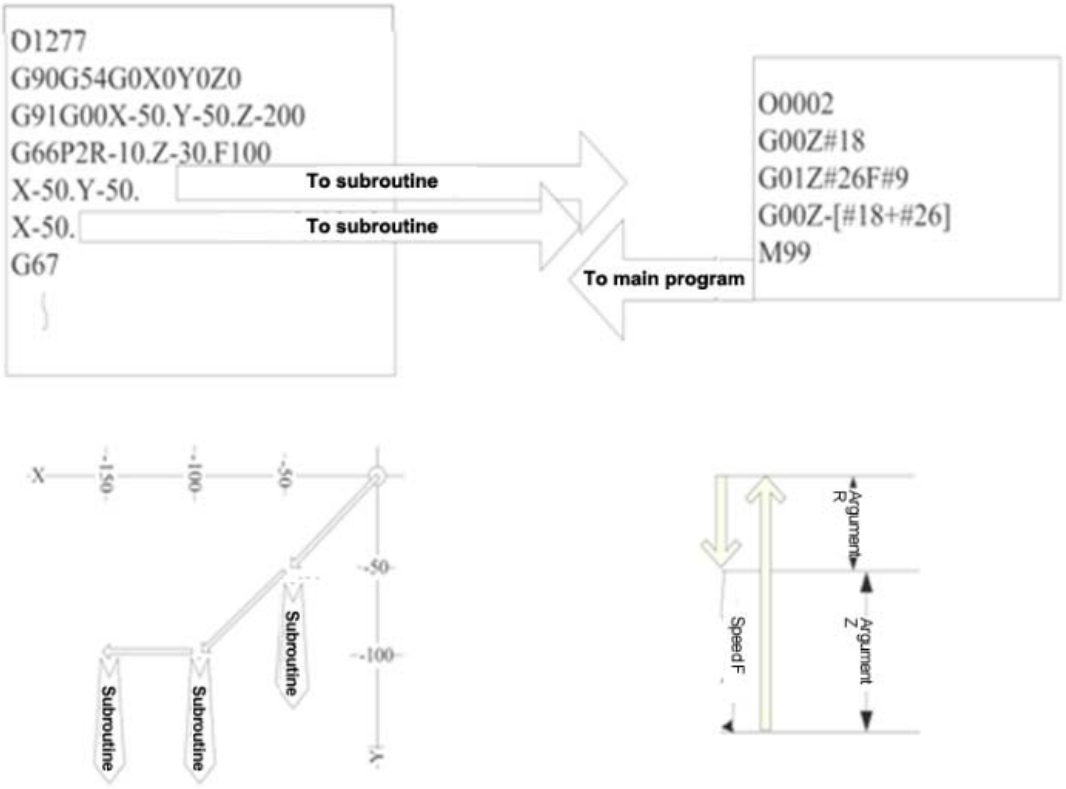
Details:

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 No.

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.

Example:

G661P0002N100G01G90X100.Y200.F400R1000,

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

14.3 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	According to #1=10,#[#[#1]]=#[#10]
#10=20	According to #10=20,#[#10]=#20
#20=30	Therefore, #5=#2 or #5=30
#5=#[#[#1]];	
#10=5	According to #1=10,#[#[#1]]=#[#10]
#10=20	According to #10=20,#[#10]=#20
#20=30	Therefore, #20=#5 or #20=1000
#5=1000	
#[#[#1]]=#5	

- Example of specifying multiple variables:

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

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

14.4 Types of variables

⑴ 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:

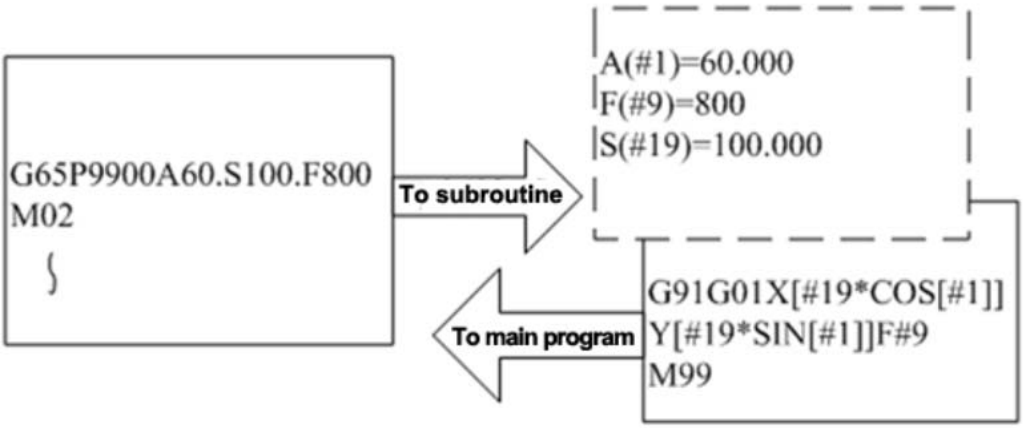
Address	Variable No.	Subroutine	Address	Variable No.	Subroutine
A	#1	○	N	×	×
B	#2	○	O	×	×
C	#3	○	P	×	×
D	#7	○	Q	#17	○
E	#8	○	R	#18	○
F	#9	○	S	#19	○
G	×	×	T	#20	○
H	#11	○	U	#21	○

I	#4	○	V	#22	○
J	#5	○	W	#23	○
K	#6	○	X	#24	○
L	×	×	Y	#25	○
M	#13	○	Z	#26	○

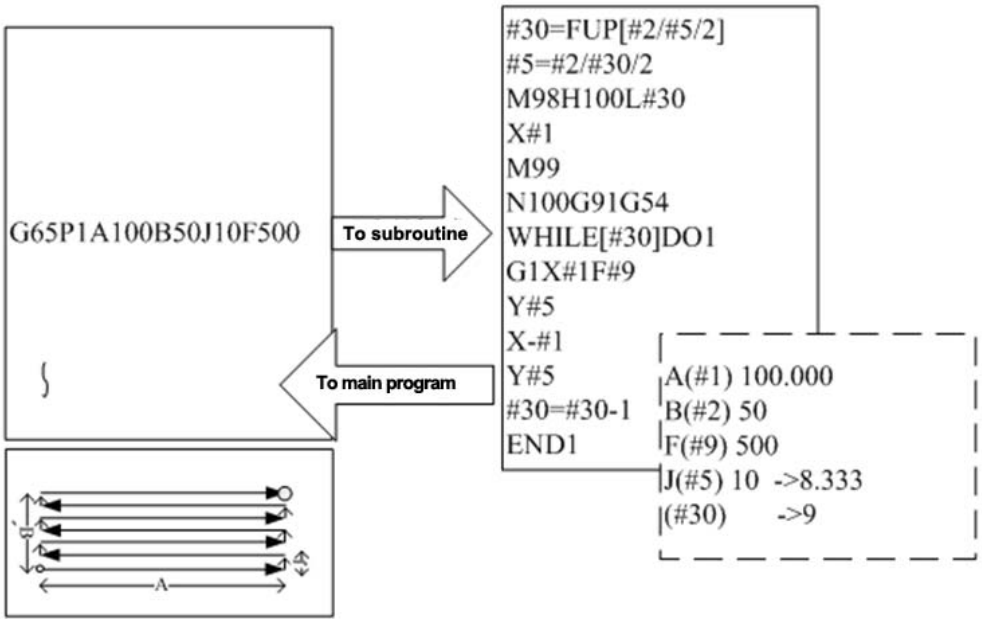
The argument addresses marked with "×" can't be used.

The argument 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.

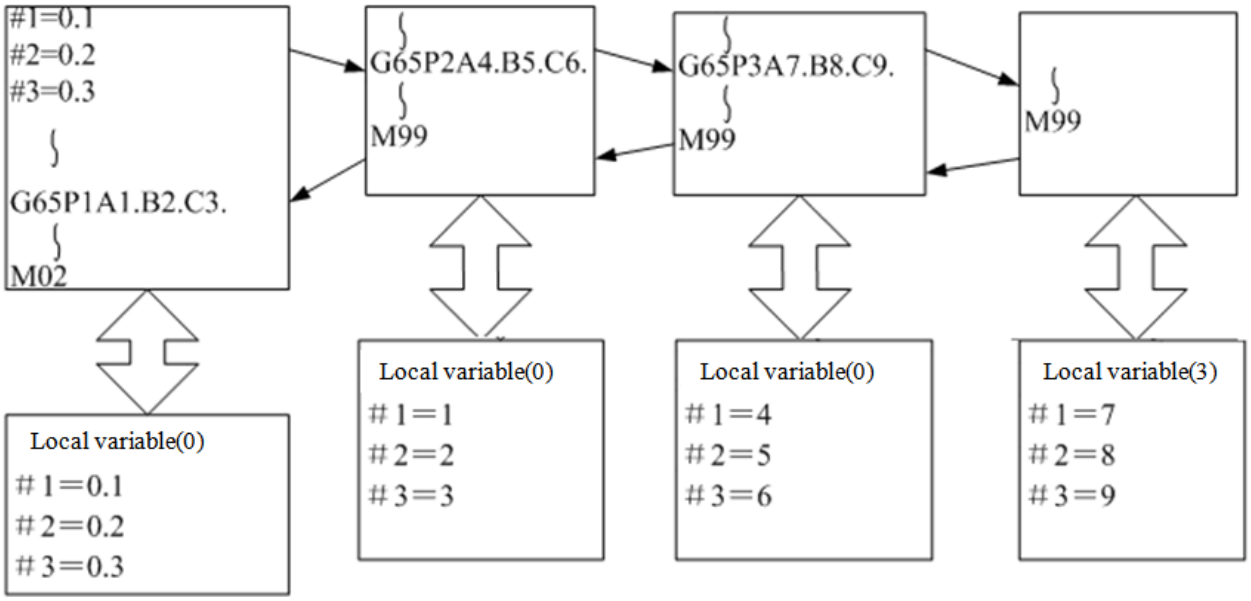


In face milling examples, argument J indicates that the spacing is 10mm during face milling; however, to ensure equal spacing processing, the spacing is changed to 8.333mm.

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.



14.5 Calculus Instruction

The variables allow various calculus expressions.

Format:

#i=[expression]

The expressions may be combinations of constants, variables, functions or subexpressions.

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

Calculation method	#i=#j	Definition/replacement
Addition and subtraction	#i=#j+#k #i=#j-#k	Addition Subtraction

	#i=#j OR #k or #i=#j|#k #i=#j XOR #k or #i=#j^#k	32-bit OR calculation (logical AND) 32-bit XOR calculation
Multiplication and division	#i=#j*#k #i=#j/#k #i=#j MOD #k #i=#j AND #k or #i=#j & #k	Multiplication Division Remainder 32-bit AND calculation (logical product)
Function calculation	#i=SIN[#k] #i=COS[#k] #i=TAN[#k] #i=ASIN[#k] #i=ATAN[#k] #i=ACOS[#k] #i=SQRT[#k] #i=ABS[#k] #i=ROUND[#k] #i=FIX[#k] #i=FUP[#k] #i=LN[#k] #i=EXP[#k]	Sine Cosine Tangent tanθ equals to sinθ/cosθ Arcsine Arctangent Arc cosine Square root Absolute value Rounding Abandon the decimal point Carry the decimal point Natural logarithm 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:

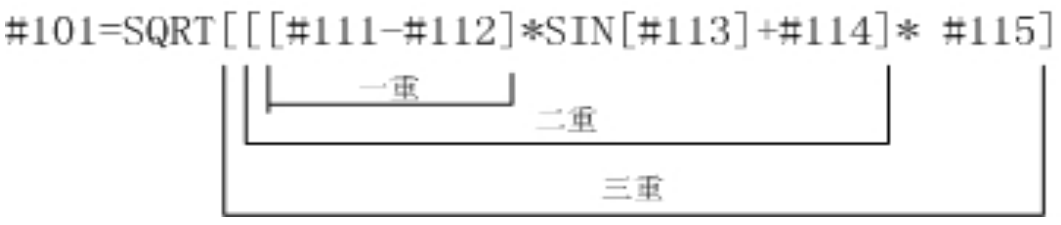
Smaller number indicates higher priority	Calculation symbol
1	#
2	[ ]
3	Function (SIN, COS, EXP…)
4	*,/,MOD
5	+, -
6	GE, GT, LE, LT
7	EQ, NE
8	AND, XOR, OR
9	=

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	#21 10000.000 #22 10000.000 #23 10000.000 #24 10000.000

	#25=100/100 #26=100./100. #27=100/100. #28=100./100. #29=#41*#101 #30=#41/#102	#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
(8) Logical product (AND)	#9 = 100 #10= #9 AND 15	#9 =01100100 15 =00001111 #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.]	#561 60.000 #562 60.000

	#563=ATAN[173.205/100000] #564=ATAN[173.205/100.] #565=ATAN[1.732]	#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 root (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 decimal point (FIX)	#21=FIX[14/3] #22=FIX[-14/3]	#21 4.000 #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 Error
(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.

14.6 Control Instruction

14.6.1 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 #i equals to #j
#i NE #j	≠ when #i doesn't equal to #j
#i GT #j	> when #i is larger than #j
#i LT #j	< when #i is smaller than #j
#i GE #j	≥ when #i is larger than or equals to #j
#i LE #j	≤ when #i is 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.

14.6.2 Cycle Conditional Instruction

Format:

WHILE[expression]DO m;(m=1,2,3…127)

…

END m;

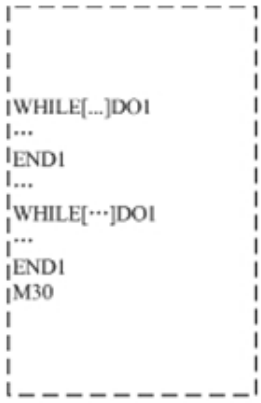
Details:

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

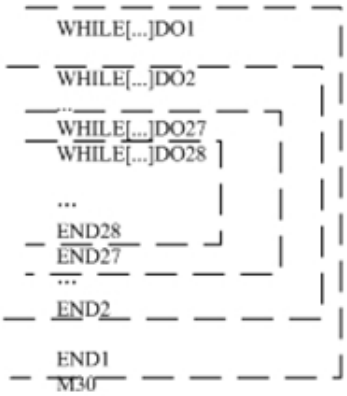
Correct

(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



Correct

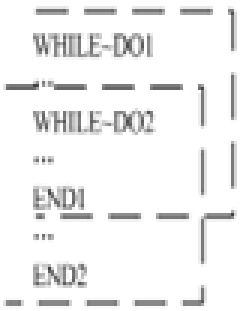
(5) WHILE~DO m must be specified before END m

False

(5、6) WHILE~DO m must be corresponded in one program

False

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

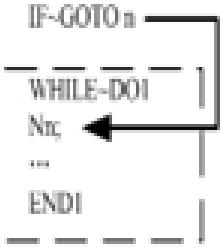


False

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

False

(9) GOTO can't jump into WHILE cycle

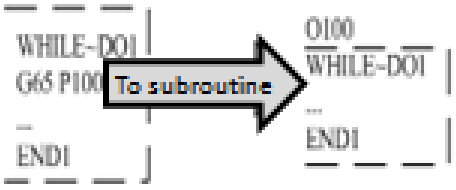


False

(10) GOTO can jump out of WHILE cycle

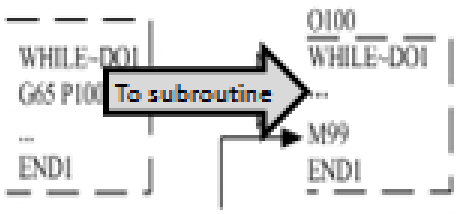
False

(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.

False



M99 causes that DO and END can't be matched

14.7 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.

15. Document Revision History

Date of Revision	Content	Revised by
2011-05-25	First version uploaded	Zhang Qinggang
2012-09-06	Documentation system uploaded	Zhang Qinggang
2012-09-28	Documentation system uploaded	Zhang Qinggang
2012-11-07	Documentation system uploaded	Zhang Qinggang