Pulser3
Programming Manual

HSC Kontrol Otomasyon

www.hsckontrol.com

HSC KONTROL OTOMASYON

Company Information

HSC Kontrol Otomasyon Sanayi Ve Dış Ticaret Limited Şirketi

Oruç Reis Mah. Giyimkent 16. Sok. No:103/A

Esenler, İstanbul / Türkiye

+90 212 544 03 20

www.hsckontrol.com

About the Document

- HSC Kontrol Otomasyon reserves the right to modify the content of this document without prior notice.
- This document may not be copied, reproduced, or distributed without the permission of HSC Kontrol Otomasyon.

Revisions		
Date	Revised by	Explanation
30/03/2023	Serkan Can	Document created
03/08/2024	Serkan Can	Detected spelling errors were corrected
18/12/2024	Yunus Emre TEKEOĞLU	Translated into English

CONTENT

1. General Information

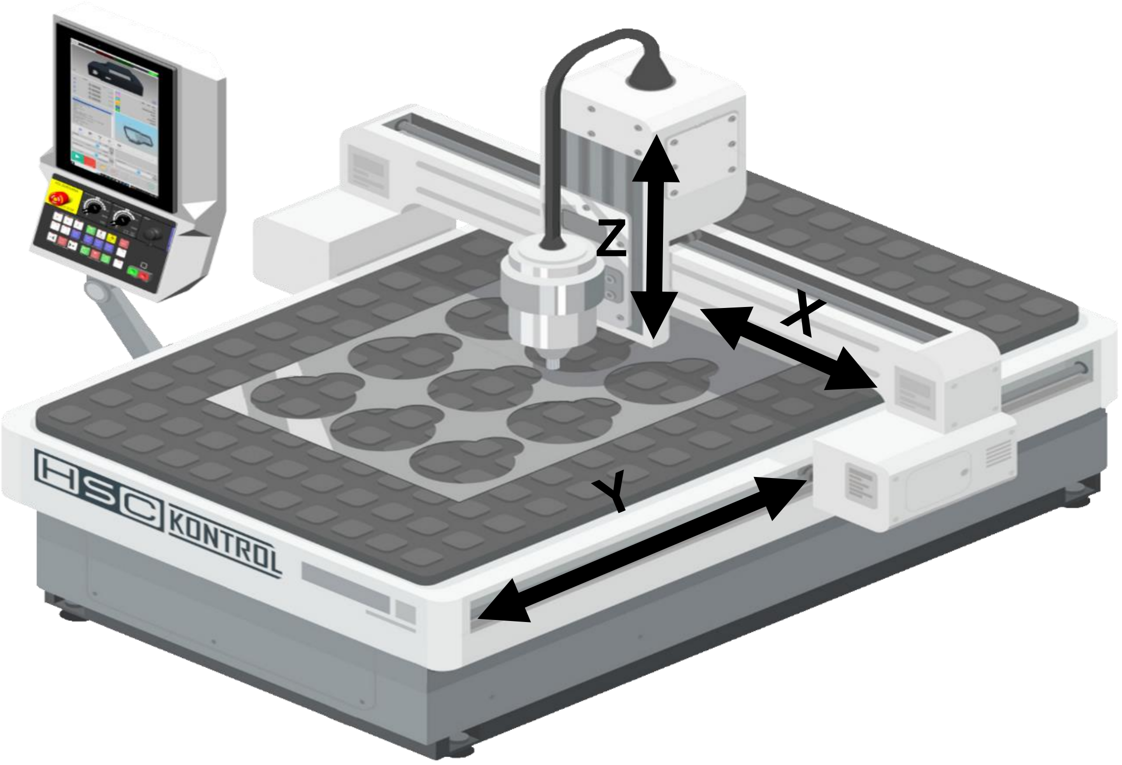
1.1. General Information

1.1.1. Controlled Axes

Axes are used to represent the linear or rotational electro-mechanical structure that generates movements such as left-right, forward-backward, or up-down in a CNC machine. Metric or inch systems can be preferred for axis control.

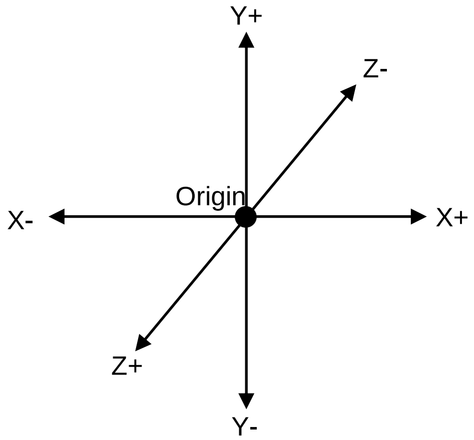
Pulser3 series control units can control up to 8 axes. The first three axes are defined as X, Y, and Z. Three-dimensional movements can be performed on these axes. The axes that rotate along the vertical lines of the X, Y, and Z axes are named A, B, and C, respectively. Additional axes parallel to or auxiliary to the X, Y, and Z axes are named U, V, and W, respectively. In lathe machines, if the U, V, and W axes are not present, these axis names are used as additive commands for the X, Y, and Z axes.

The unit of measurement for rotary axes is degrees.

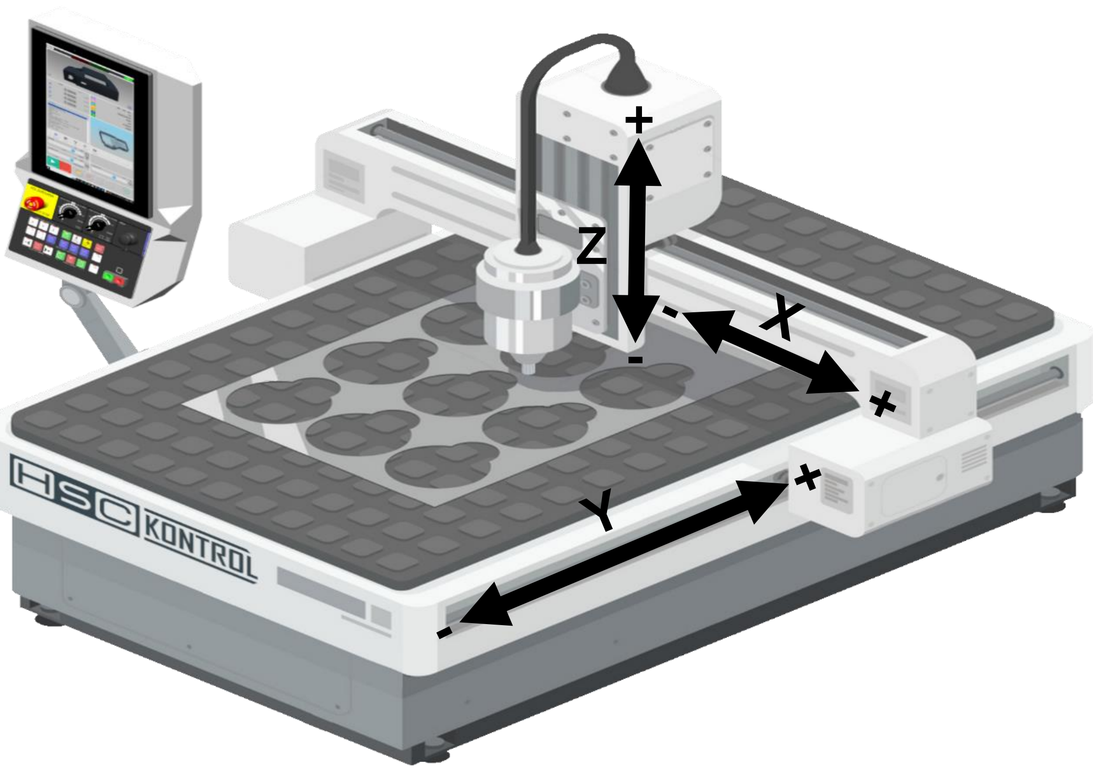


1.1.2. Coordinate System

Pulser3 series control units are built on the Cartesian coordinate system.



Axis names and movement directions should be adjusted according to the Cartesian coordinate system.



1.1.3. Reference Point

The reference point is the actual zero point of the machine. It cannot be altered by the user. For each axis, the position where the machine stops after the referencing process is the 0.0000 coordinate. All part offset values are entered relative to this reference point. When part offset values are left at 0, the part zero will coincide with the reference point.

1.1.4. Absolute and Incremental Programming

In CNC machines, target coordinates can be specified in two ways: absolute and incremental. In absolute programming, target coordinates are specified by giving the actual coordinate values where the tool tip is desired to move. In incremental programming, target coordinates are specified as the distance from the current position of the axes.

1.1.5. Spindle

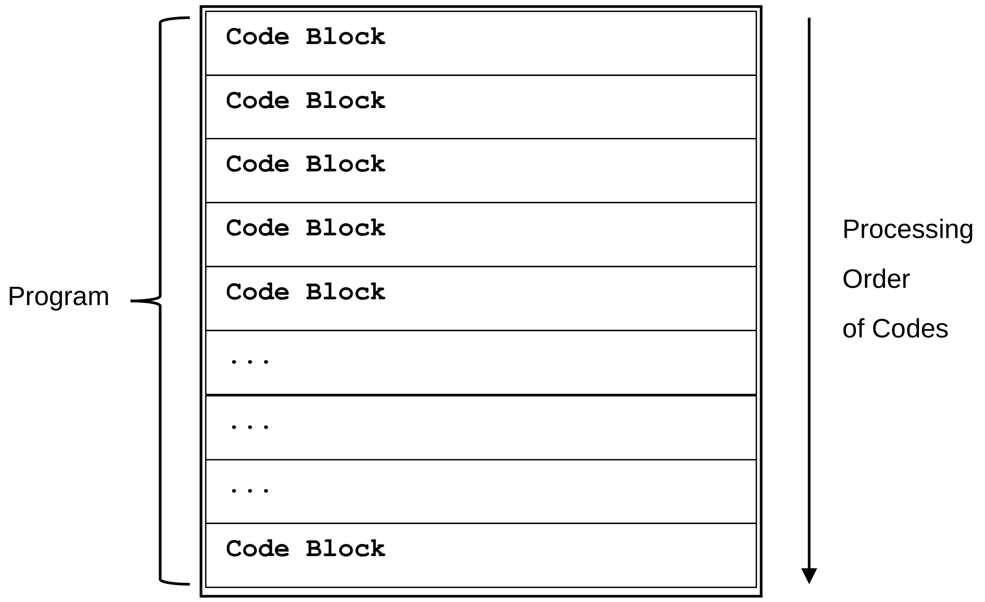
In milling machines, the electro-mechanical system that rotates the tool is called the spindle, while in lathe-type machines, the electro-mechanical system that rotates the workpiece is called the spindle. Generally, M codes are prepared for commands to rotate clockwise, counterclockwise, and to stop the spindle. For adjusting the spindle speed, the S code is used.

1.1.6. Tool

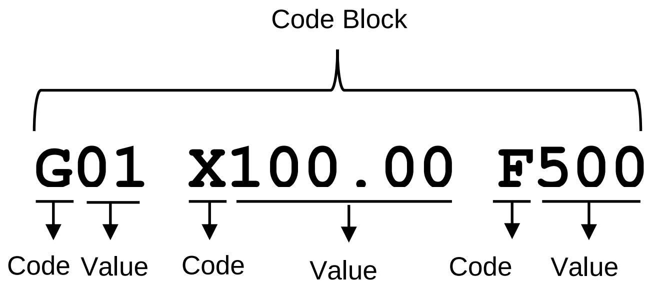
In milling machines, cutting tools attached to the spindle head and, in turning machines, cutting tools placed on the turret are referred to as tools. Many machines are equipped with automatic tool changers to accommodate the use of multiple tools. In milling machines, the automatic tool changing feature is called ATC (Auto Tool Changer). In lathe machines, the turret rotates to bring the required tool.

1.2. Programming Structure

Programs are a group of commands executed to control the machine by the CNC control unit. These commands are written in sequence, line by line, to be executed in order. Each line contains commands for axis movement, operation of peripheral units, tool changes, etc. These lines are also referred to as G-code blocks. Each G-code line (block) must be terminated with a line end character.



CNC codes are created by a command indicated by a letter, followed by a numerical value. Each letter (code) has a specific meaning.



The functions of the codes are as follows:

Code	Function	Alternative Function	Type
A	A-axis movement		Decimal
B	B-axis movement		Decimal
C	C-axis movement		Decimal
D	Tool radius compensation number		Integer
E	Extruder axis movement		Decimal
F	Cutting feedrate		Decimal
G	Preparation codes		Decimal
H	Tool length compensation number		Integer
I	Distance to the center of the arc on the X axis		Decimal
J	Distance to the center of the arc on the Y axis		Decimal
K	Distance to the center of the arc on the Z axis		Decimal
L	Subprogram repetition count		Integer
M	General purpose use codes		Integer
N	Line number		Integer
O	Program number		Integer
P	Subprogram number / Waiting time	Auxiliary integer	Integer
Q	Auxiliary decimal number		Decimal
R	Radius value	Auxiliary decimal number	Decimal
S	Spindle rotation RPM		Integer
T	Tool number		Integer
U	U-axis movement	X axis incremental target (Lathe)	Decimal
V	V-axis movement	Y axis incremental target (Lathe)	Decimal
W	W-axis movement	Z axis incremental target (Lathe)	Decimal
X	X-axis movement	Wait value	Decimal
Y	Y-axis movement		Decimal
Z	Z-axis movement		Decimal

The codes must be written adjacent to their corresponding values. Two codes do not have to be adjacent; a space can be left between them. A dot (.) cannot be added to the value next to integer codes. Characters like "+", "-", or "." cannot be added before the codes.

Incorrect Writing Examples:

X 100   (There is a space between X and the value 100)

G04 P20. (A . (dot) has been added next to a code of integer type)

-X100   (A – (minus sign) has been added before the X code)

Correct Writing Examples:

```
X100
G04 P20
X-100
```

If a code of decimal number type does not have a "." (dot) at the end, the decimal part of the number is processed as .0000. Similarly, if only the dot is added without any value following it, this value will also be processed as .0000. Similarly, any digit not added after the dot will be processed as 0.

`X100` => X100.0000 will be processed

`X100.` => X100.0000 will be processed

`X100.2` => X100.2000 will be processed

A code can be followed by either a fixed number or a variable. Variables are indicated with the "#" prefix.

#0 = 2000000

`X#0` => X200.0000 will be processed

`X-#0` => X-200.0000 will be processed

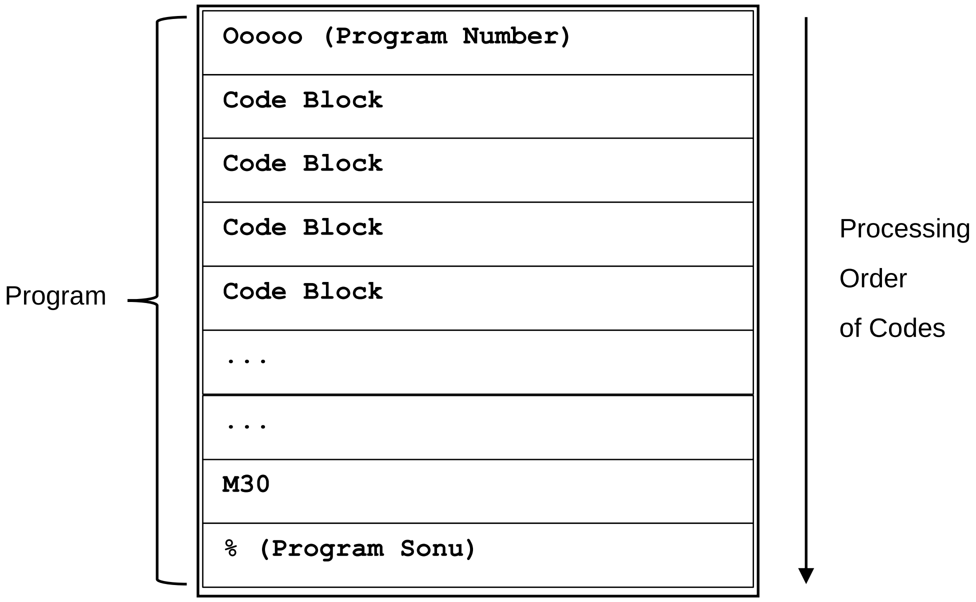
A value or variable number must always be specified next to a code. Each line must end with a line-ending character. A ";" (semicolon) can be added at the end of the line, but it is not mandatory. It is recommended to add the "%" character at the end of the program.

In the program, comments can be written between the characters "(" and ")". The content within these characters will be ignored and not processed.

```
(COMMENT)
X100. (COMMENT NEXT TO THE CODE)
```

The codes can be specified in either uppercase or lowercase. A code block cannot exceed 63 characters in length.

A program template can be created as follows:



1.3. Library

Multiple programs can be stored in the library and selected for use at different times. The names of the programs cannot exceed 63 characters in length.

Supported file extensions: **.nc .cnc .tap .ngc .anc .gcode .txt**

In the library, subprograms can be stored alongside programs. Subprograms must follow specific naming rules. A subprogram should start with a capital "O" followed by a 4-digit program number. The file extension must be **.cnc**.

Example: **O9001.cnc**

A program file can be copied to the library via the USB port on the HMI or through FTP from a PC.

2. G Code List

Function	Group	Milling	Lathe
Rapid positioning	1	G00	G00
Ping pong rapid positioning	1	G00.1	G00.1
Linear interpolation	1	**G01**	**G01**
Circular interpolation(Clockwise)	1	G02	G02
Circular interpolation(Counter clockwise)	1	G03	G03
Dwell	0	G04	G04
Programmable data input	0	G10	G10
Cartesian coord. system selection	17	**G15**	**G15**
Polar coordinate system selection	17	G16	G16
XY Plane selection	2	**G17**	G17
ZX Plane selection	2	G18	**G18**
YZ Plane selection	2	G19	G19
Inch system selection	6	**G20**	**G20**
Millimeter system selection	6	G21	G21
Axis limit setting On	4	-	G22
Axis limit setting Off	4	-	**G23**
Move to reference point	0	G28	G28
Move to the 2. 3. 4. ref.point	0	G30	G30
Motion skip (SKIP) function	0	G31	G31
Constant-Pitch threading	1	G33	G33
Constant-Pitch threading with rotary axis	1	G33.1	G33.1
Variable-Pitch threading	1	G34	G34
Variable-Pitch threading with rotary axis	1	G34.1	G34.1
Tool radius compensation Off	7	**G40**	**G40**
Tool radius compensation left	7	G41	G41
Tool radius compensation right	7	G42	G42
Tool length offset compensation(+)	8	G43	-
RTCP On	8	G43.4	-
Tool length offset compensation(-)	8	G44	-
Tool length compensation + RTCP Off	8	**G49**	-
At the end of motion, motion signal is active	0	G50.1	G50.1
At the end of motion, motion signal is inactive	0	G50.2	G50.2
Temporary coordinate system	0	G52	G52
Machine coordinate system	0	G53	G53
1. Work offset selection	14	**G54**	**G54**
2. Work offset selection	14	G55	G55
3. Work offset selection	14	G56	G56
4. Work offset selection	14	G57	G57
5. Work offset selection	14	G58	G58
6. Work offset selection	14	G59	G59
7. Work offset selection	14	G59.1	G59.1
8. Work offset selection	14	G59.2	G59.2
9. Work offset selection	14	G59.3	G59.2
10. Work offset selection	14	G59.4	G59.4

Continuation of the G-Code List

Function	Group	Milling	Lathe
Exact stop check	15	G61	G61
Continuous cutting mode	15	**G64**	**G64**
Macro command	0	G65	G65
Macro modal call	12	G66	G66
Macro modal call cancel	12	**G67**	**G67**
Rotate coordinate system	0	G68	-
Turn Off coord. system rotation	0	G69	-
Turret mirroring On	0	-	G68
Turret mirroring Off	0	-	**G69**
Laser On (Piercing)	0	G70	-
Laser Lead in	0	G70.1	-
Lazer cutting	0	G70.2	-
Lazer lead out	0	G70.3	-
Laser Off	0	G71	-
Laser height calibration	0	G72	-
Laser single shot test	0	G72.1	-
Laser gas test	0	G72.2	-
Laser gas test cancel		G72.3	-
Peck drilling cycle	9	G73	-
Reverse tapping cycle for milling	9	G74	-
Fine boring cycle for milling	9	G76	-
Cancel canned cycle	9	**G80**	-
Simple drilling cycle	9	G81	-
Drilling cycle/reverse boring	9	G82	-
Peck drilling cycle	9	G83	-
Tapping cycle	9	G84	-
Boring cycle	9	G85	-
Boring cycle	9	G86	-
Boring cycle, backboring	9	G87	-
Boring cycle	9	G88	-
Boring cycle	9	G89	-
Absolute programming	3	**G90**	**G90**
Incremental programming	3	G91	G91
Coord. system/Spindle max. speed	0	G92	G92
Feedrate per minute	5	G94	G94
Feedrate per revolution	5	G95	G95
Fixed surface speed control On	13	-	G96
Fixed surface speed control Off	13	-	**G97**
Return to Z point in canned cycle	10	**G98**	**G98**
Return to R point In canned cycle	10	G99	G99

G codes active at startup are highlighted in bold. G codes are divided into two groups: one-shot and modal G codes. Group 0 G codes are one-shot and lose their validity after the line is completed. All other G codes remain in memory until another G code from the same group is issued.

3. M Code List

Code	Function	Note
M00	Program stop	<CNC>
M01	Optional program stop	<CNC>
M02	End of program	<CNC>
M03	Spindle start clockwise	
M04	Spindle start counterclockwise	
M05	Spindle stop	
M06	Tool change	It is generally used in milling models
M07	Coolant ON – Mist coolant/Coolant thru spindle	It is generally used in milling models
M08	Coolant ON – Flood coolant	
M09	Coolant OFF	
M10	Milling: Lock the table	Lathe: Chuck clamp
M11	Milling: Unlock the table	Lathe: Chuck unclamp
M13	Spindle2 start clockwise	
M14	Spindle2 start counterclockwise	
M15	Spindle 2 stop	
M19	Spindle orientation	It is generally used in milling models
M20	Spindle orientation cancel	It is generally used in milling models
M21	Tool clamp	It is generally used in milling models
M22	Tool unclamp	It is generally used in milling models
M25	C/Spd axis spindle control	
M26	C/Spd axis C-axis control	
M30	End of program (Return to start)	<CNC>
M98	Subprogram call	<CNC>
M99	Return from subprogram	<CNC>

Note: M codes are divided into two groups: those directly interpreted by the CNC system and those left to the user. The M00, M01, M02, M30, M98, and M99 codes are interpreted by the CNC system and are not recommended to be modified. All other M codes are assigned by the machine manufacturer. A list of commonly used M codes is provided above. However, these M codes may be assigned to different functions by the machine manufacturer. You can obtain the appropriate list of M codes for your machine from your machine manufacturer.

4. G-Code Descriptions

4.1. G00: Rapid positioning

The axes are moved quickly to the specified target. Only the axes with codes added next to the G00 code will move. The coordinate each axis will move to is specified by the value next to the corresponding axis code. When the absolute distance command type (G90) is selected, the target coordinate is given as an absolute value. In incremental command mode (G91), the axes move by the specified value from their current position. The movement speeds of the axes depend on the PRM48-PRM53 parameters. These speeds can also be scaled using the ROV function (%0, %25, %50, %100). When the FOV ratio is set to 0%, the movement is paused.

Format:	**G00 X_ Y_ Z_ A/B/C_ U/V/W_**

X: X axis target coordinate

Y: Y axis target coordinate

Z: Z axis target coordinate

A/B/C: A/B/C axis target coordinate

U/V/W: U/V/W axis target coordinate (Incremental X/Y/Z target in lathe model)

In the milling model, the U/V/W codes specify the target coordinates of auxiliary axes. In the lathe model, the U/V/W commands specify the incremental targets of the X/Y/Z axes.

4.2. G00.1: Rapid positioning with arc movement (Ping-Pong)

It is used to move the axes rapidly to the specified target by drawing an arc. Only the axes with codes added next to the G00.1 code will move along with the Z axis. The coordinate each axis will move to is specified by the value next to the corresponding axis code. When the absolute distance command type (G90) is selected, the target coordinate is specified as an absolute value. In the incremental command type (G91), the axes move by the specified value from their current position. The movement speeds of the axes depend on the PRM48-PRM53 parameters. Additionally, these speeds can be scaled with the ROV function (%0, %25, %50, %100). When the FOV ratio is set to 0%, the movement is paused.

Format:	**G00.1 X_Y_ Z_**

X: X axis target coordinate

Y: Y axis target coordinate

Z: Z axis target coordinate

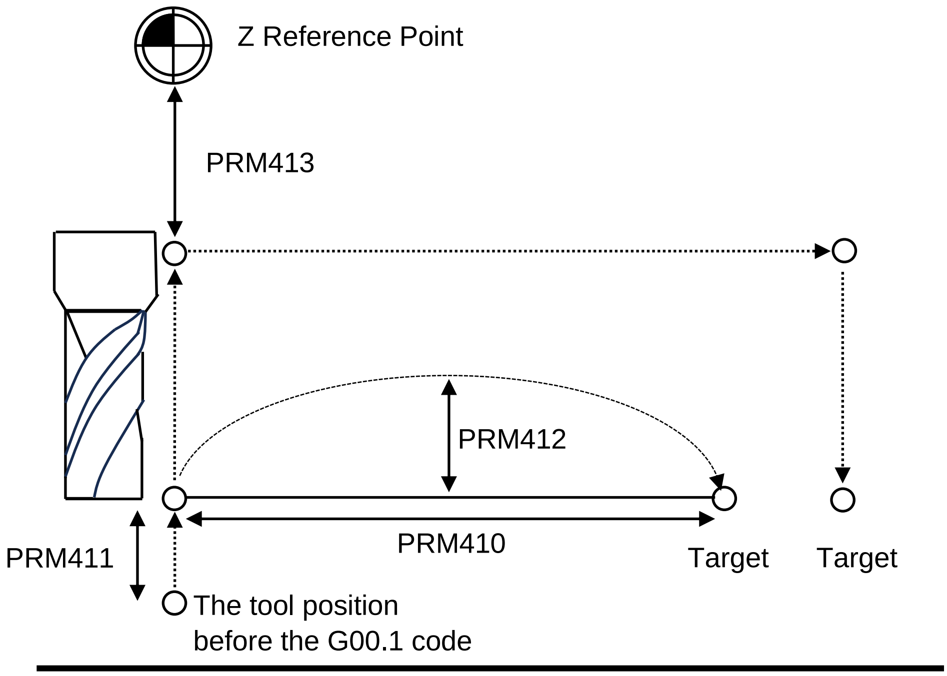
This code is typically used in laser and plasma cutting machines. For the code to function properly, the relevant parameters must be configured. Before using this code, please consult your machine manufacturer/system integrator to ensure the necessary settings have been made.

When a target is given with the G00.1 code, the vector length from the current position of the axes to the target is considered. If this length is equal to or shorter than the distance specified by the PRM410 parameter:

1- The Z axis is raised by the distance specified in the PRM411 parameter from its current position.
2- The movement is performed in such a way that an arc is formed on the Z axis towards the target coordinate. The peak of the arc, as an incremental value, must be specified by PRM412 in relation to the retraction point.
3- The Z axis is moved to the target coordinate.

When a target is given with the G00.1 code, and the vector distance from the current position of the axes to the target is longer than the distance specified by the PRM410 parameter:

1- The Z axis is moved to the position specified by PRM413.
2- Movement is performed on the X and Y axes to the target coordinates.
3- The Z axis is moved to the target coordinates.



Example:

```
M3 S1000              (SPINDLE CW ROTATION)
G90 G00 X100. Y100.   (RAPID MOVEMENT TO X100 Y100 POINT)
G90 G00 Z10.          (RAPID MOVE TO Z SAFE POSITION)
G01 Z-1. F500         (ENTER THE WORKPIECE AT F500)
X200. F1000.          (PERFORM CUTTING AT F1000 UNTIL X200.)
G00 Z10.              (MOVE TO THE SAFE Z POSITION)
G00.1 Y200. Z10.      (PERFORM AN ARC MOTION TO Y200)
G01 Z-1. F500         (ENTER THE WORKPIECE AT F500)
X100. F1000.          (PERFORM CUTTING AT F1000 UNTIL X100.))
G00 Z10.              (MOVE TO THE SAFE Z POSITION)
G53 Z0.               (GO TO Z AXIS REFERENCE)
M5                    (SPINDLE STOP)
```

4.3. G01: Linear Interpolation

This is used to move the axes along a linear line towards a specified target while cutting. Only the axes with codes added next to the G01 code will move. The coordinate each axis will move to will be specified next to the code for that axis. When the absolute distance command type (G90) is selected, the target coordinate is specified as an absolute value. In the incremental command type (G91), the axes move by the specified value from their current position. The movement speeds of the axes are specified with the value added next to the F code. This speed will remain valid until a new F code is specified. Additionally, the cutting feed can be scaled with the FOV function (%0~%150). When the FOV ratio is set to 0%, movement is paused. The specified cutting feed command is processed according to the status of the G94 (units per minute) / G95 (units per revolution) codes.

Format:	G01 X_ Y_ Z_ A/B/C_ U/V/W_ F_

X: X axis target coordinate

Y: Y axis target coordinate

Z: Z axis target coordinate

A/B/C: A/B/C axis target coordinate

U/V/W: U/V/W axis target coordinate (Incremental X/Y/Z target in lathe model)

F: Cutting feedrate

In the milling model, U/V/W codes indicate the target coordinates of auxiliary axes. In the lathe model, however, U/V/W commands specify the incremental targets for the X/Y/Z axes.

G01 Xα Yβ Zγ Aδ Ff

<u>The movement speeds of the axes:</u>

$$\alpha: F\alpha = \frac{\alpha}{L} * f \qquad \beta: F\beta = \frac{\beta}{L} * f$$

$$\gamma: F\gamma = \frac{\gamma}{L} * f \qquad \delta: F\delta = \frac{\delta}{L} * f$$

<u>The length of the movement:</u>

$$L = \sqrt{\alpha^2 + \beta^2 + \gamma^2 + \delta^2}$$

4.4. G02/G03: Circular interpolation

The axes are used to move to the specified target along a circular path while cutting. When the absolute distance command type (G90) is selected, the target coordinates are specified as absolute values. In incremental command type (G91), the axes move by the specified value from their current position. The I/J/K commands should always be given relative to the starting position of the arc. The I/J/K and R commands should not be given in the same line. The movement feedrate of the axes are specified by the value added next to the F code. This feedrate remains valid until a new F code is specified. Additionally, the cutting speed can be adjusted using the FOV function (%0~%150). When the FOV ratio is set to 0%, the movement is ed. The given cutting speed command is processed according to the status of the G94 (unit/minute) or G95 (unit/rev) codes.

Format:	**G17/G18/G19 G02/G03 X_ Y_ Z_ I/J/K/R_ F_**

G17: The arc motion will be performed in the XY plane

G18: The arc motion will be performed in the ZX plane

G19: The arc motion will be performed in the YZ plane

X: X axis target coordinate

Y: Y axis target coordinate

Z: Z axis target coordinate

I: Distance to the center of the arc on the X axis

J: Distance to the center of the arc on the Y axis

K: Distance to the center of the arc on the Z axis

R: Radius of the arc

F: Cutting feedrate

The arc motion is performed in the plane specified by G17/G18/G19.

In the milling model, the U/V/W codes specify the target coordinates of the auxiliary axes. In the lathe model, the U/V/W commands specify the incremental targets of the X/Y/Z axes.

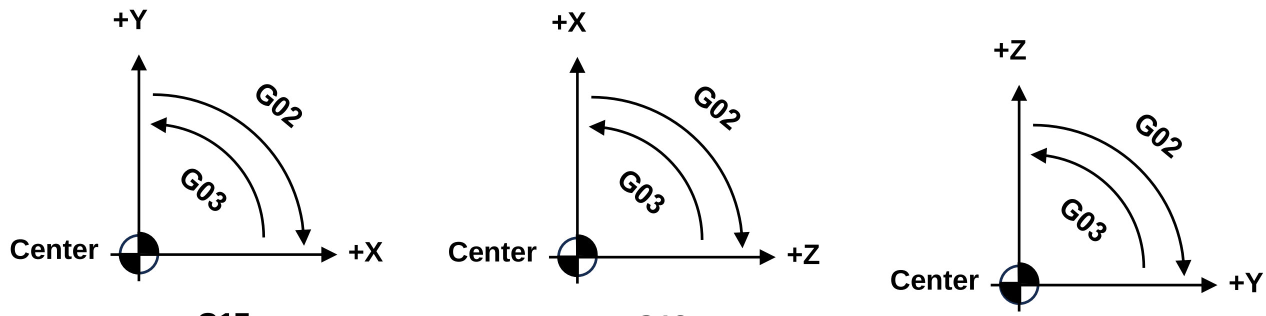
In the lathe model, the parameter PRM430 allows you to select how the target value is specified, either in radius programming or diameter programming.

PRM430:

0: Radius programming

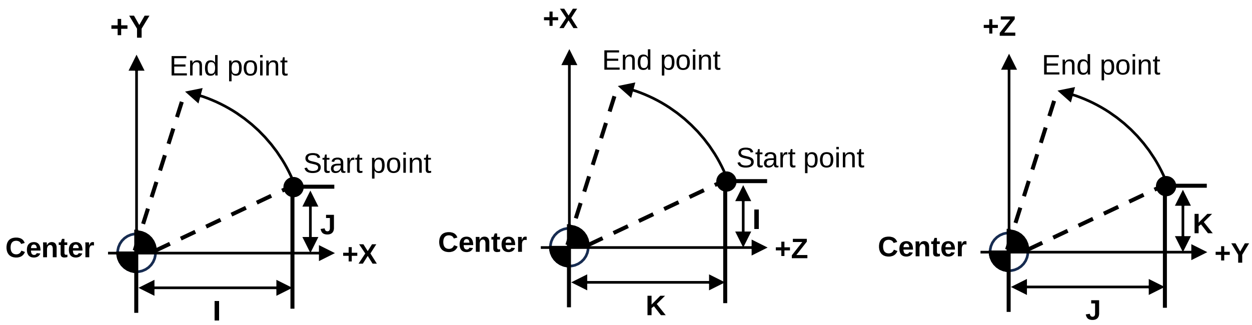
1: Diameter programming

Arc direction:



G17 G18 G19

Arc center:



G17 G18 G19

Programming the arc by specifying the radius:

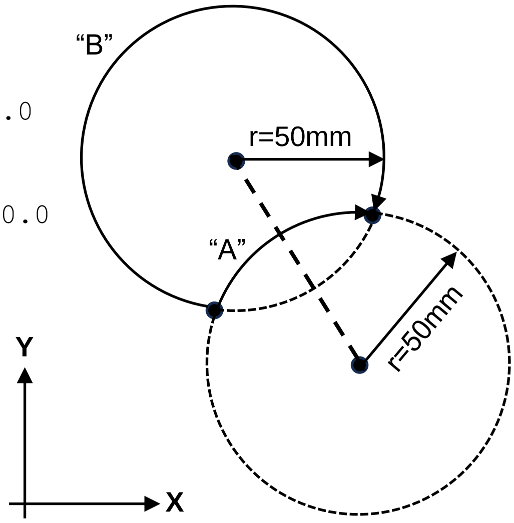
Arc motion can also be specified by using the R code instead of the I/J/K codes. The value specified alongside the R code represents the radius of the arc. When using the R code for arc motion, the arc movement should be either greater than or less than 180 degrees. If an arc movement greater than 180 degrees is required, the radius value should be specified as negative.

For an arc '**A**' smaller than 180 degrees**:**

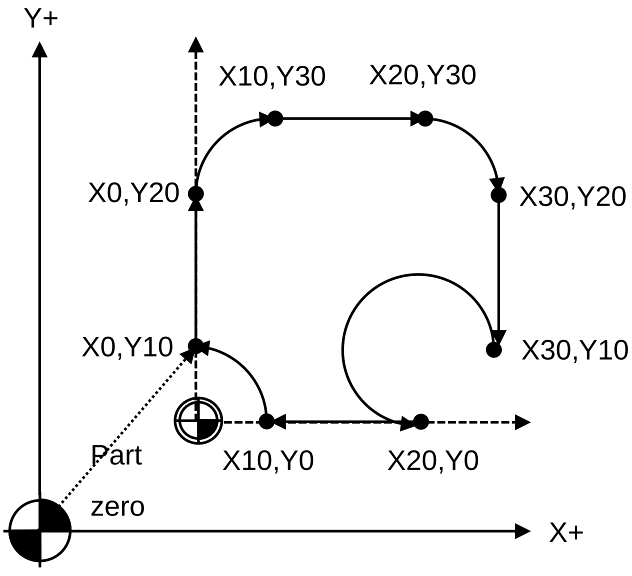
```
G91 G02 X60.0 Y20.0 R50.0 F300.0
```

For an arc '**B**' greater than 180 degrees**:**

```
G91 G02 X60.0 Y20.0 R-50.0 F300.0
```



Example:



```
M3 S1000              (SPINDLE CW ROTATION)
G90 G00 X0. Y10.      (RAPID MOVEMENT TO X0 Y10 POINT)
G90 G00 Z10.          (RAPID MOVE TO Z SAFE POSITION)
G01 Z-1. F500         (ENTER THE WORKPIECE AT F500)
X0. Y20. F1000.       (PERFORM CUTTING AT F1000 UNTIL Y20.)
(EXAMPLE OF CLOCKWISE ARC MOTION USING I AND J)
G17 G02 X10. Y30. I10. J0.
G01 X20.
(EXAMPLE OF AN ARC MOTION LESS THAN 180 DEGREES USING R)
G02 X30. Y20. R10.
G01 Y10.
(EXAMPLE OF AN ARC MOTION GREATER THAN 180 DEGREES USING R)
G03 X20. Y0. R-10.
G01 X10.
(EXAMPLE OF A COUNTERCLOCKWISE ARC MOTION USING IJ)
G03 X0. Y10. I-10. J0.
G00 G53 Z0.           (Z REFERENCE)
M5                    (SPINDLE STOP)
```

4.5. G04: Dwell

It is used to make a pause during the operation. Either X or P code can be specified next to the G04 code as the dwell time.

Format:	**G04 X_/P_**

X: Dwell time (seconds)

P: Dwell time (miliseconds)

Example:

```
G04 X1.5   (DWELL FOR 1.5 SECONDS)
G04 P2000  (DWELL FOR 2 SECONDS)
G04        (EXACT STOP)
```

4.6. G10: Programmable Data Input

G10 code is used to modify tool length and workpiece origin settings. The values provided in the command are stored permanently. In milling and turning models, it is used differently for tool offset adjustments. The general structure of the code is as follows.

Format:	**G10 L_ P_**

L: Data group to be adjusted

- **1**: Adjusting tool offset values
- **2:** Adjusting workpiece origin offset values
- **3**: Technology block selection in laser machines
- **4**: Introducing the workpiece size to the system

P: The sequence number of the data to be adjusted

The given axis values are considered as absolute or incremental according to the G90 and G91 commands in the milling model. In the lathe model, X/Y/Z/A/B/C commands are considered as absolute or incremental based on the G90 and G91 commands, while U/V/W commands are always interpreted as incremental..

4.6.1. Tool Length Adjustment (Milling)

Format:	**G10 L1 P_ R_**

P: The sequence number of the data to be adjusted

- **1~100**: Adjusts the tool length compensation value
- **101~200**: Adjusts the tool radius compensation value
- **201~300**: Adjusts the tool length compensation wear values
- **301~400**: Adjusts the tool radius compensation wear values

R: The value to be assigned to the area specified by the **P** code

4.6.2. Tool Length Adjustment (Lathe)

Format:	**G10 L1 P___ X(U)_ Y(V)_ Z(W)_ Q_ R_**

P: The sequence number of the data to be adjusted

1~50: Adjusts the geometry offset values

51~100: Adjusts the geometry offset values

X: Adjusts the X-axis geometry offset value (Absolute or Incremental)

U: Adjusts the X-axis geometry offset value (Incremental)

Y: Adjusts the Y-axis geometry offset value (Absolute or Incremental)

V: Adjusts the Y-axis geometry offset value (Incremental)

Z: Adjusts the Z-axis geometry offset value (Absolute or Incremental)

W: Adjusts the Y-axis geometry offset value (Absolute or Incremental)

Q: Adjusts the tool type

R: Adjusts the tool radius value (Incremental)

Only one of the X and U codes should be written.

Only one of the Y and V codes should be written.

Only one of the Z and W codes should be written.

4.6.3. Adjusting Workpiece Offset Values (Milling)

Format:	**G10 L2 P_ X_ Y_ Z_ A/B/C_ U/V/W_**

P: The sequence number of the data to be adjusted

- **0**: Adjusts the G92 (All offsets) values
- **1**: Adjusts the G54 values
- **2**: Adjusts the G55 values
- **3**: Adjusts the G56 values
- **4**: Adjusts the G57 values
- **5**: Adjusts the G58 values
- **6**: Adjusts the G59 values
- **7**: Adjusts the G59.1 values
- **8**: Adjusts the G59.2 values
- **9**: Adjusts the G59.3 values
- **10**: Adjusts the G59.4 values

X: Adjusts the X axis value of the offset group specified by the **P** code

Y: Adjusts the Y axis value of the offset group specified by the **P** code

Z: Adjusts the Z axis value of the offset group specified by the **P** code

A/B/C: Adjusts the A/B/C axis value of the offset group specified by the **P** code

U/V/W: Adjusts the U/V/W axis value of the offset group specified by the **P** code

4.6.4. Adjusting Workpiece Offset Values (Lathe)

Format:	**G10 L2 P_ X(U)_ Y(V)_ Z(W)_ A/B/C_**

P: The sequence number of the data to be adjusted

- **0**: Adjusts the G92 (All offsets) values
- **1**: Adjusts the G54 values
- **2**: Adjusts the G55 values
- **3**: Adjusts the G56 values
- **4**: Adjusts the G57 values
- **5**: Adjusts the G58 values
- **6**: Adjusts the G59 values
- **7**: Adjusts the G59.1 values
- **8**: Adjusts the G59.2 values
- **9**: Adjusts the G59.3 values
- **10**: Adjusts the G59.4 values

X: Adjusts the X axis value of the offset group specified by the **P** code(Absolute or Incremental)

U: Adjusts the X axis value of the offset group specified by the **P** code(Incremental)

Y: Adjusts the X axis value of the offset group specified by the **P** code(Absolute or Incremental)

V: Adjusts the Y axis value of the offset group specified by the **P** code(Incremental)

Z: Adjusts the Z axis value of the offset group specified by the **P** code(Absolute or Incremental)

W: Adjusts the Z axis value of the offset group specified by the **P** code(Incremental)

A/B/C: Adjusts the A/B/C axis value of the offset group specified by the **P** code

Only one of the X or U codes should be written.

Only one of the Y or V codes should be written.

Only one of the Z or W codes should be written.

4.6.5. Technology block selection in laser machines i

Format:	**G10 L3 P_**

P: The sequence number of the selected technology block

A total of 10 technology blocks are reserved for use in laser cutting machines. The value written next to the P code should be between 0 and 9. P0 code selects the first technology block.

4.6.6. Introducing the workpiece size to the system

Format:	**G10 L4 X_ Y_ Z_**

X: The X dimension of the workpiece

Y: The Y dimension of the workpiece

Z: The Z dimension of the workpiece

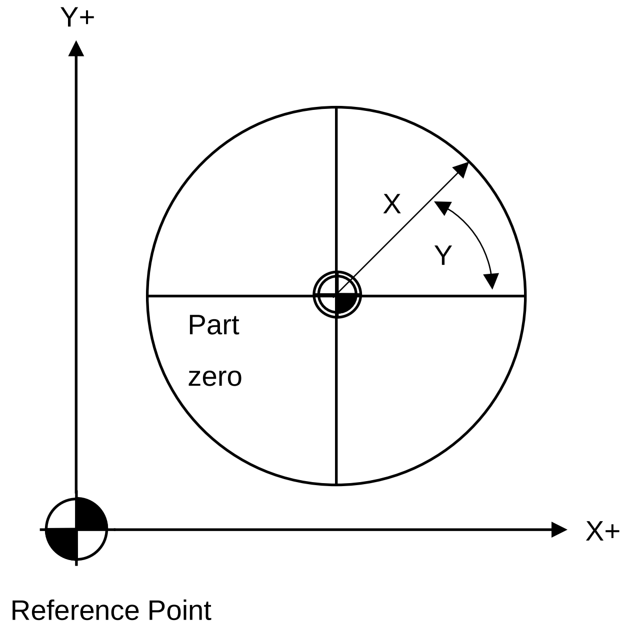
4.7. G15/G16: Coordinate system selection

These codes are used to change the coordinate system in use. The Cartesian coordinate system is the default coordinate system. In the polar coordinate system, X represents the radius and Y represents the angle. When the angle specified by Y is positive, it represents counterclockwise direction; when negative, it represents clockwise direction. It is commonly used for drilling holes on a circle. The polar coordinate system can only be used for target values in G00, G01, and canned cycles.

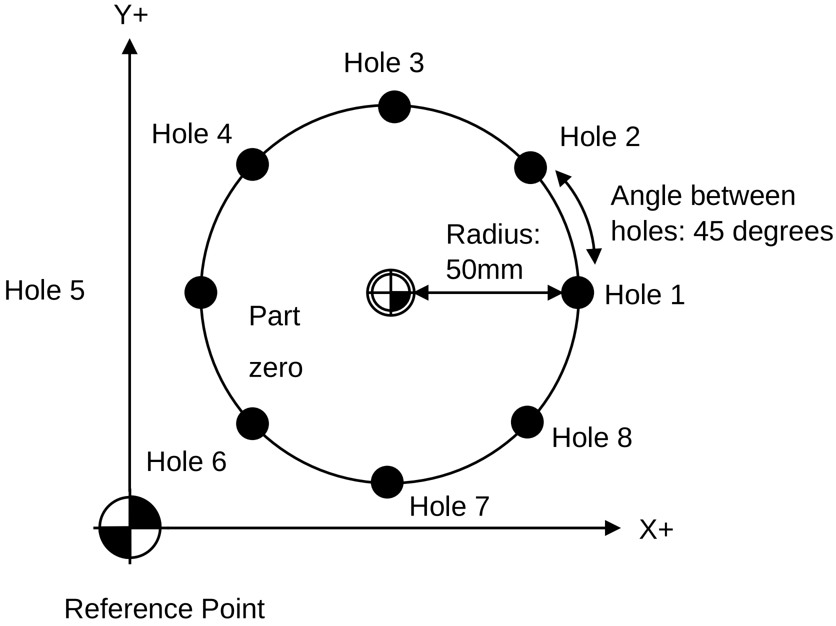
Format:	**G15/G16**

G15: Cartesian coordinate system selection

G16: Polar coordinate system selection



Example:




```
G00 G53 Z0.            (Z AXIS REFERENCE)
M3 S1000               (SPINDLE CW ROTATION)
G16                    (POLAR COORDINATE SYSTEM ENABLED)
G90 G00 X50. Y0.       (RAPID MOVEMENT TO X60 Y0 POINT)
G90 G00 Z10.           (MOVE TO THE SAFE Z POSITION)
G90 G99 G81 X50. Y0. Z-10. R5. F120.
Y45.                   (HOLE 2)
Y90.                   (HOLE 3)
Y135.                  (HOLE 4)
Y180.                  (HOLE 5)
Y225.                  (HOLE 6)
Y270.                  (HOLE 7)
Y315.                  (HOLE 8)
G80 G00 G53 Z0.        (Z AXIS REFERENCE)
G15                    (CARTESIAN COORDINATE SYSTEM ENABLED)
M5                     (SPINDLE STOP)
```

<u>Example 2 (Programming the same holes using G91):</u>

```
G00 G53 Z0.          (Z AXIS REFERENCE)
M3 S1000             (SPINDLE CW ROTATION)
G16                  (POLAR COORDINATE SYSTEM ENABLED)
G90 G00 X50. Y0.     (RAPID MOVEMENT TO X50 Y0 POINT)
G90 G00 Z10.         (MOVE TO THE SAFE Z POSITION)
G90 G99 G81 X50. Y0. Z-10. R5. F120.
G91 Y45. L7
G90 G80 G00 G53 Z0.  (Z AXIS REFERENCE)
G15                  (CARTESIAN COORDINATE SYSTEM ENABLED)
M5                   (SPINDLE STOP)
```

4.8. G17/G18/G19: Plane Selection

Arc motion, tool offsets, and canned cycles are applied to the selected axis group. The axis for which the tool length offset will be applied is determined by PRM320. In the milling model, G17 is selected by default at startup, whereas in the turning model, G18 is the default selection at startup.

<u>PRM320:</u>

0: Tool length offset is always applied to the Z axis

1: Tool length offset is applied to the X, Y, and Z axes according to the selected plane

Format:	**G17/G18/G19**

G17: XY Plane selection

G18: ZX Plane selection

G19: YZ Plane selection

4.9. G20/G21: Unit Selection

These codes are used to change the unit of measurement.

If the system encounters one of these codes and detects that the current unit needs to be changed, a "restart the system" alarm will be generated. After restarting the system, the relevant parameters and offset values will be automatically adjusted to the new unit by the system. However, user variables will not be adjusted.

For this reason, ensure that subprograms such as tool measurement or tool changing, which utilize user variables, are programmed by the machine manufacturer to function properly in both measurement units..

Format:	**G20**

Format:	**G21**

G20: Inch system selection

G21: Millimeter system selection

4.10. G22/G23: Registered Stroke Control

It is used to limit the movement range of the axes. The maximum values of the area in which the axes can move should be specified with X, Y, and Z. The minimum values of the area the axes can move within should be specified with I, J, and K. Registered stroke control is applied to the axes specified with the G22 command. Limit values should be specified in groups, such as X with I, Y with J, and Z with K. These values should be programmed as machine coordinates. When the axes are intended to move outside the specified area, a software limit alarm for the corresponding axis will occur. It can be selected to prohibit movement either inside or outside the area specified by PRM337. These limits are not cleared with the reset button. To disable the limits, the G23 command must be issued.

Format:	**G22 X_ Y_ Z_ I_ J_ K_**

Format:	**G23**

G22: Axis limit setting On

G23: Axis limit setting Off

X: Maximum limit value of the X axis

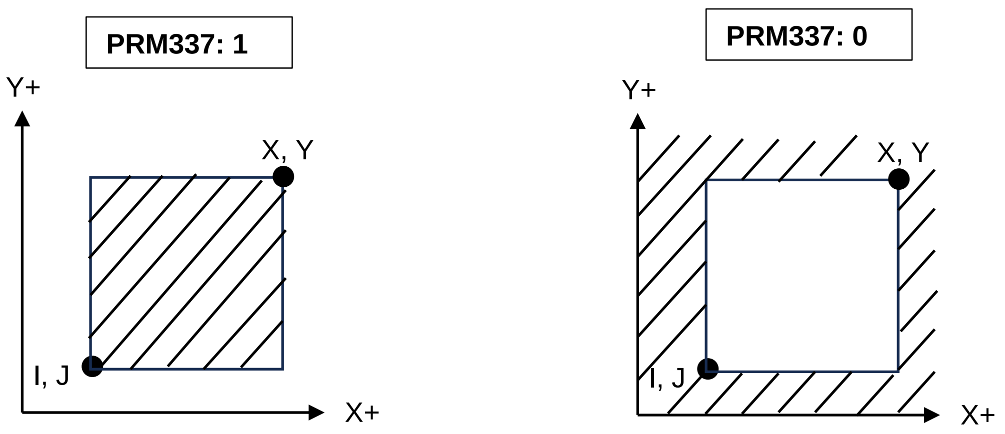
Y: Maximum limit value of the Y axis

Z: Maximum limit value of the Z axis

I: Minimum limit value of the X axis

J: Minimum limit value of the Y axis

K: Minimum limit value of the Z axis



Shaded area (///): restricted area

4.11. G28: Move to Reference Point

The axes first move to the specified center point and then to the 1st reference point (home position). For accurate movement to the correct point, the axes must be sent to the reference.

Format:	**G28 X_ Y_ Z_ A/B/C_ U/V/W_**

G28: 1. Reference point movement command

X: X position to move to before reaching the reference point

Y: Y position to move to before reaching the reference point

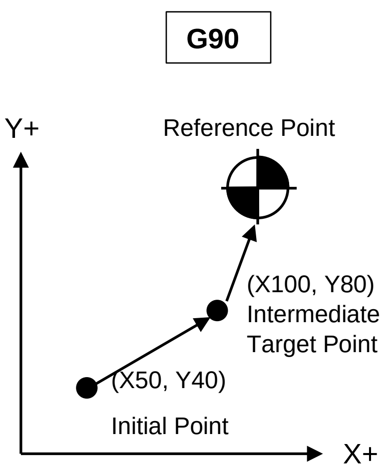
Z: Z position to move to before reaching the reference point

A/B/C: A/B/C position to move to before reaching the reference point

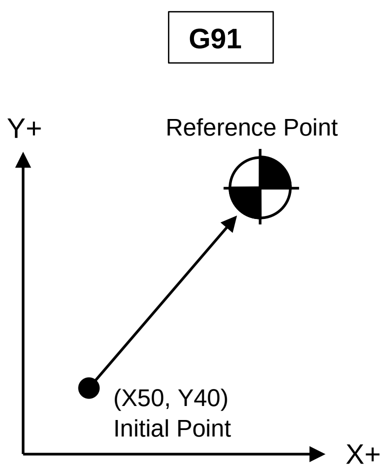
U/V/W: U/V/W position to move to before reaching the reference point

If an axis code is specified in the command, that specific axis or axes will move. If no axis command is added to the G28 command, all defined axes in the machine will move. In incremental mode, if the target value for an axis is specified as 0, the machine will move directly to the reference point.

__Example:__




```
G90 G28 X100. Y80.
```




```
G91 G28 X0. Y0.
```

4.12. G30: Move to 2nd/3rd/4th Reference Point

The axes are moved first to the specified center point, and then to the 2nd/3rd/4th reference point. For accurate movement to the correct position, the axes must be referenced first.

Format:	**G30 P2/P3/P4 X_ Y_ Z_ A/B/C_ U/V/W_**

G30: Reference point move command

P2/P3/P4: Reference point selection

X: X position to be moved to before reaching the reference point

Y: Y position to be moved to before reaching the reference point

Z: Z position to be moved to before reaching the reference point

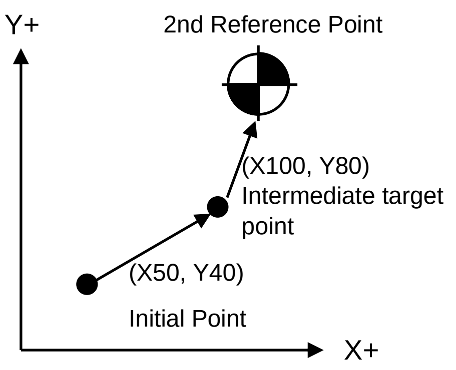
A/B/C: A/B/C position to be moved to before reaching the reference point

U/V/W: U/V/W position to be moved to before reaching the reference point

In the axis codes, the axis or axes that are specified as a command will move accordingly. If no axis commands are added next to the G30 command, all axes defined in the machine will move. In incremental mode, when the axis target value is set to 0, the machine will directly move to the reference point. The 2nd/3rd/4th reference points of the axes can be set using parameters PRM168~PRM191.

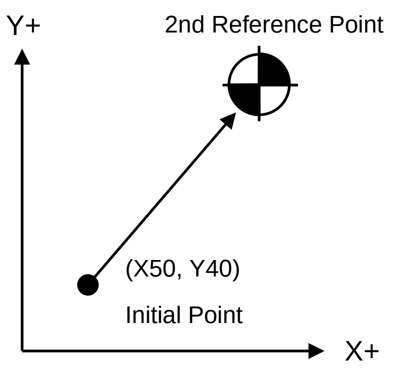
Example:

G90




```
G90 G30 P2 X100. Y80.
```

G91




```
G91 G30 P2 X0. Y0.
```

4.13. G31: Motion Skip (SKIP) Function

G31 allows for the cancellation of a linear movement command before completion through an external signal. For the G31 command to function, the necessary software and settings must be configured by the machine manufacturer. It is used for purposes such as measuring a part with a probe, automatic tool radius measurement, and automatic tool length measurement.

Format:	**G31 X_ Y_ Z_ A/B/C_ U/V/W_**

G31: If the external SKIP signal is active, skip this line without completing it

X: X axis target coordinate

Y: Y axis target coordinate

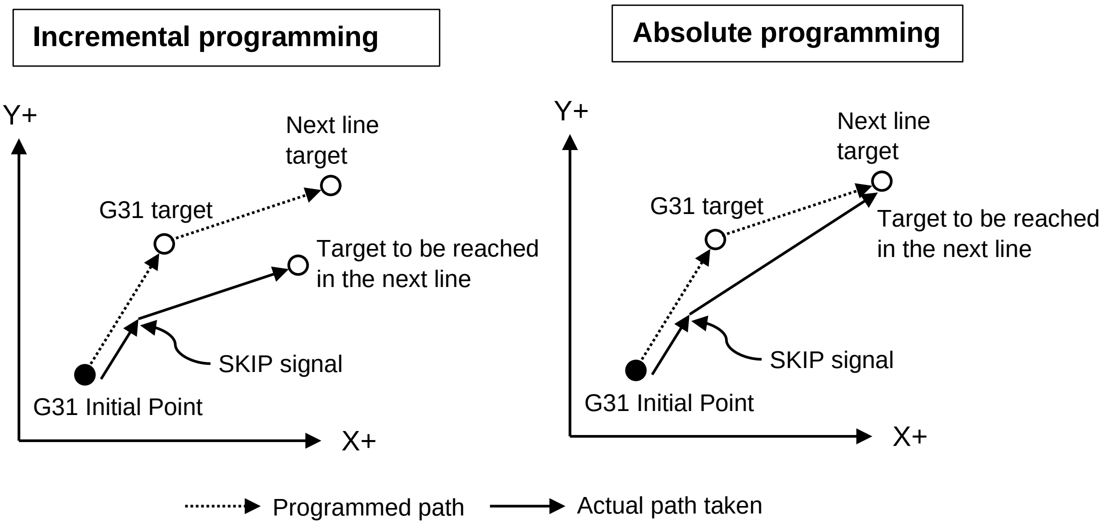
Z: Z axis target coordinate

A/B/C: A/B/C axis target coordinate

U/V/W: U/V/W axis target coordinate

In the axis codes, the axis or axes specified as a command will move accordingly. The machine coordinate values at the moment the SKIP signal is detected are stored in the variables #5060~#5067.

Example:



4.14. G33/G33.1: Thread cutting

It is used for thread cutting and is typically required in lathe models. The F command specifies the thread profile, and the Q command specifies the thread entry angle. While this command is being executed, the axis movements are synchronized with the turret spindle angle. The part is approached from the same angle in each pass.

Format:	**G33/G33.1 X_ Y_ Z_ F_ Q_**

G33: Constant-Pitch threading with external encoder

G33.1: Constant-Pitch threading with rotary axis

X: X axis target coordinate

Y: Y axis target coordinate

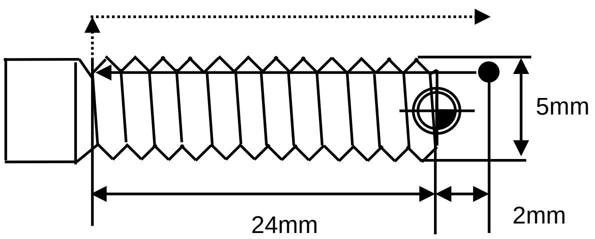
Z: Z axis target coordinate

F: Thread pitch

Q: Thread initial angle

In the axis codes, the axis or axes specified as a command will move accordingly. For the thread cutting command to function, an encoder must be directly connected to the spindle, and the necessary settings must be configured by the machine manufacturer. The recommended resolution is 1024 pulses/rev. If a servo motor is connected to the spindle, thread cutting can be performed without an external encoder using the G33.1 code. Similarly, the required settings for this command must also be configured by the machine manufacturer.

Example:



```
      G00 X4.5 Z2.    (RAPID MOVEMENT TO THE STARTING POINT)
      G33 Z-24. F2.   (THREAD CUTTING COMMAND)
      G0 X10.         (RAPID RETRACTION ON THE X AXIS)
      G0 Z2.          (RAPID MOVEMENT TO THE STARTING POINT ON THE
Z AXIS)
```

4.15. G34/G34.1: Variable-Pitch threading

It is used for variable-pitch threading and is typically required in lathe models. The thread pitch is specified with the F command, and the thread entry angle is defined with the Q command. Depending on the selected plane, the I/J/K commands define the pitch variation per revolution. A positive value results in an increasing pitch, while a negative value results in a decreasing pitch. During the execution of this command, axis movements progress in synchronization with the spindle angle. The part is approached from the same angle in each pass.

Format:	**G34/G34.1 X_ Y_ Z_ F_ Q_ I/J/K_**

G34: Variable-pitch threading command with external encoder

G34.1: Variable-pitch threading command with servo turret spindle

X: X axis target coordinate

Y: Y axis target coordinate

Z: Z axis target coordinate

F: Thread pitch

Q: Thread initial angle

I/J/K: Pitch variation per revolution

In the axis codes, the axis or axes specified as a command will move accordingly. For the thread cutting command to function, an encoder must be directly connected to the turret spindle, and the necessary settings must be configured by the machine manufacturer. The recommended resolution is 1024 pulses/rev. If a servo motor is connected to the turret spindle, thread cutting can be performed without the need for an external encoder using the G34.1 code. Similarly, the required settings for this command must also be configured by the machine manufacturer.

4.16. G40/G41/G42: Tool radius compensation

It is used to compensate for the radius of the cutting tool. A new path is created either inside or outside the programmed tool path, and movement is performed along this new path.

Format:	**G41/G42 D_**

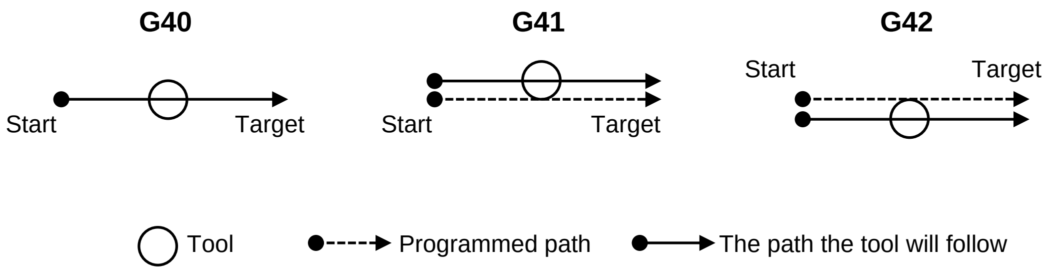
Format:	**G40**

G41: Tool radius compensation left

G42: Tool radius compensation right

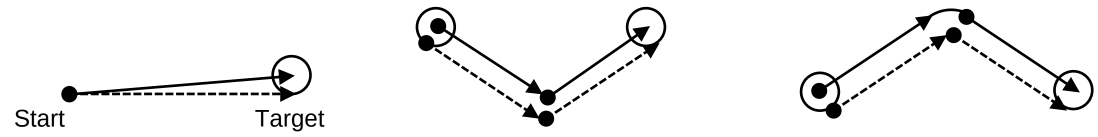
D: The offset number where the radius compensation value to be applied is written

G40: Tool radius compensation Off



When tool radius compensation is applied, the consecutive movements within the program are pre-read, interpreted, and a new vector path is generated. The tool is then moved along this new path. To achieve correct results while using this function, the number of lines between two movements should be no less than 10. During the creation of the new path, if the target coordinates of the next line cannot be found, the tool is positioned 90 degrees to the left or right of the target of the processed line. If the angle between two consecutive movements is less than 180 degrees, trimming is performed at the end of the processed movement and the beginning of the next movement. If the angle between two consecutive movements is greater than 180 degrees, an arc is added connecting the starting points of both movements.

When the tool radius compensation command is given, the following first movement must be linear. Tool radius compensation should not be initiated with an arc movement.



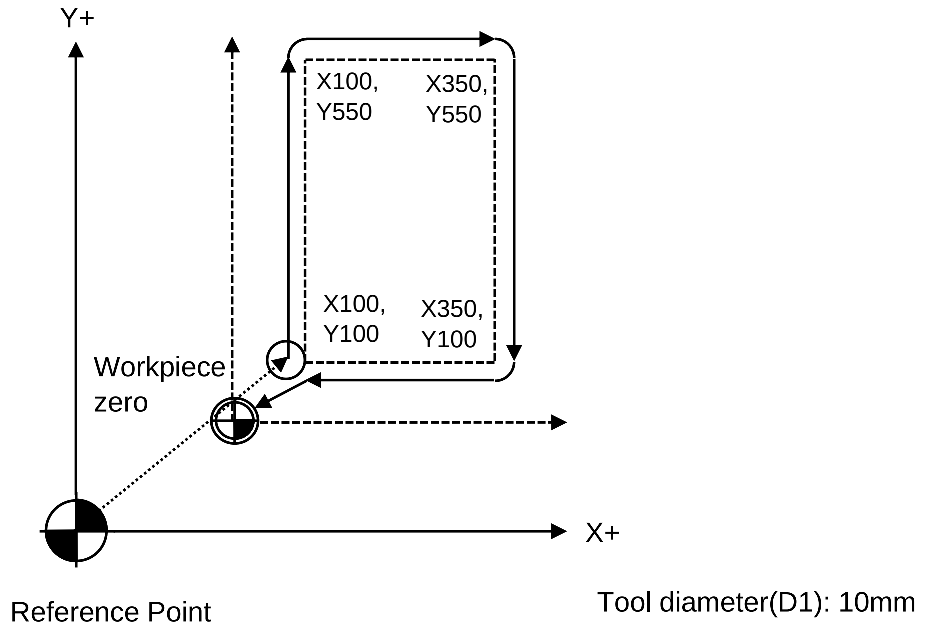
The next movement is unknown

The angle is less than 180 degrees.

The angle is greater than 180 degrees.

Tool Programmed path The path the tool will follow

Example:




```
G00 G53 Z0.           (Z REFERENCE)
M3 S1000              (SPINDLE CW ROTATION)
G41 D1                (RADIUS COMPENSATION LEFT)
G90 G00 X100. Y100.   (RAPID MOVEMENT TO X100 Y100 POINT)
G90 G00 Z10.          (MOVE TO THE SAFE Z POSITION)
G01 Z-1. F500         (ENTER THE WORKPIECE AT F500)
Y550. F1000.          (PERFORM CUTTING AT F1000 UNTIL Y550.)
X350.                 (PERFORM CUTTING AT F1000 UNTIL X350.)
Y100.                 (PERFORM CUTTING AT F1000 UNTIL Y100.)
X100.                 (PERFORM CUTTING AT F1000 UNTIL X100.)
G40                   (RADIUS COMPENSATION OFF)
G00 G53 Z0.           (Z REFERENCE)
G00 X0. Y0.           (MOVE THE TOOL AWAY FROM THE CUTTING
AREA)
M5                    (SPINDLE STOP)
M30
%
```

4.17. G43/G43.4/G44/G49: Tool Length Compensation (RTCP)

It is used to compensate for the length difference between tools, allowing all tools to be programmed with the same Z-axis position. Each tool has a dedicated length compensation value, which must be entered by the user.

Format:	G43/G44 H_

Format:	G49

G43: Tool length compensation positive (+) direction

G44: Tool length compensation negative (-) direction

H: The offset number where the tool length compensation value to be applied is written

G49: Tool length compensation off

Tool length compensation values can always be applied to the Z-axis, or they can be distributed to the X, Y, and Z axes depending on the selected plane. This selection can be adjusted using PRM320.

PRM320:

0: Tool length compensation is always applied to the Z-axis.

1: Tool length compensation is applied to the X, Y, and Z axes according to the selected plane.

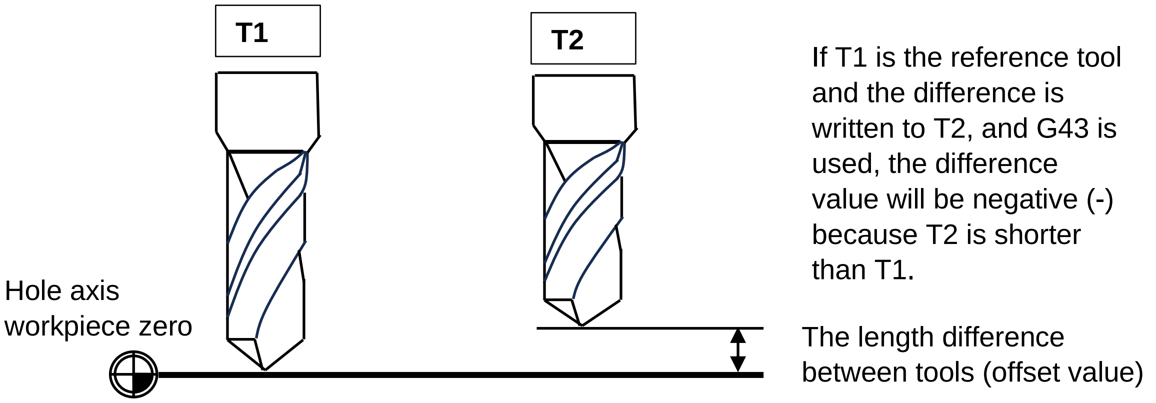
Format:	G43.4

G43.4: It has been allocated. Please contact HSC Control.

Writing of compensation values:

The tool length offset value can be used in different ways. The most common method of usage is as follows.

1- T1 is considered the reference tool.
2- The Z axis (or hole axis according to the plane) is touched to the surface of the part.
3- At this point, the Z axis workpiece zero is set. (The Z absolute position becomes 0.0000).
4- T2 is selected.
5- The Z axis (or hole axis according to the plane) is touched to the surface of the part.
6- The value of the Z axis absolute position is written as compensation to H2.
7- For each tool, the process is repeated starting from step 4.



If T1 is the reference tool and the difference is written to T2, and G43 is used, the difference value will be negative (-) because T2 is shorter than T1.

Example:

```
M6 T1              (TAKE T1)
M3 S2000           (SPINDLE CW ROTATION)
G43 H1             (APPLY H1 AS TOOL LENGTH COMPENSATION)
(DRILL TO THE POINT X100 Y-250.)
(RETURN TO THE R POINT AFTER THE DRILLING OPERATION)
G90 G99 G81 X100. Y-250. Z-150. R10. F120.
G80 G53 G00 Z0. (MOVE TO THE SAFE Z POSITION)
G49                (TOOL LENGTH COMPENSATION OFF)
M6 T2              (TAKE T1)
M3 S2000           (SPINDLE CW ROTATION)
G43 H2             (APPLY H2 AS TOOL LENGTH COMPENSATION)
(DRILL TO THE POINT X200 Y-250.))
G81 X200. Y-250. Z-150. R10. F120.
G80 G53 G00 Z0. (GO TO Z AXIS REFERENCE)
G49                (TOOL LENGTH COMPENSATION OFF)
M5                 (SPINDLE STOP)
```

4.18. G52: Temporary Coordinate System

It is used to set the axes to the specified values. With this command, the shifted coordinate system shifts all workpiece zeros (G54 to G59.4). The shift values are not permanent and will be canceled when one of the G54 to G59.4 commands is given.

Format:	**G52 X_ Y_ Z_ A/B/C_ U/V/W_**

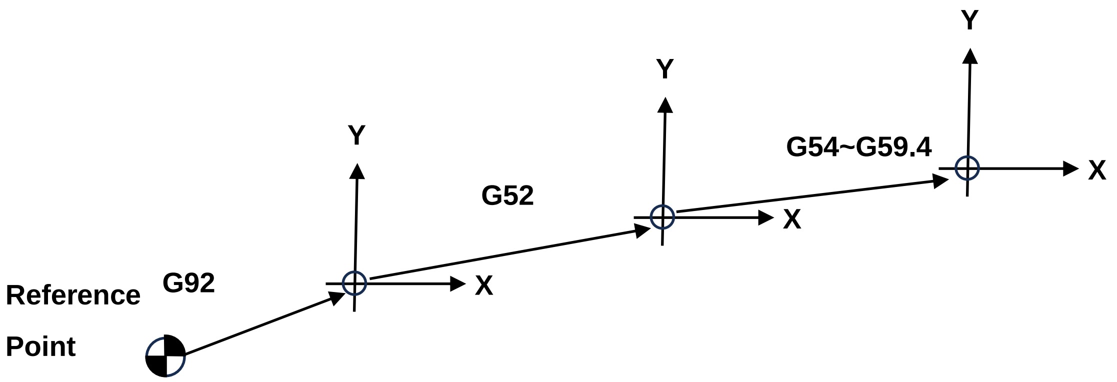
X: The absolute coordinate to which the X axis is intended to be set

Y: The absolute coordinate to which the Y axis is intended to be set

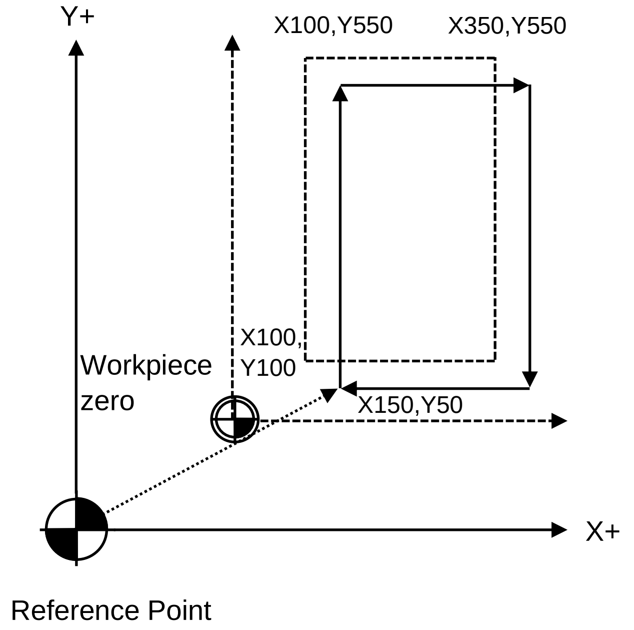
Z: The absolute coordinate to which the Z axis is intended to be set

A/B/C: The absolute coordinate to which the A/B/C axis is intended to be set

U/V/W: The absolute coordinate to which the U/V/W axis is intended to be set



Example:




```
G00 G53 Z0.            (Z AXIS REFERENCE)
M3 S1000               (SPINDLE CW ROTATION)
G90 G00 X150. Y50.     (RAPID MOVEMENT TO X150 Y150 POINT)
G52 X100. Y100.        (SHIFT THE COORDINATE SYSTEM)
G90 G00 Z10.           (MOVE TO THE SAFE Z POSITION)
G01 Z-1. F500          (ENTER THE WORKPIECE AT F500)
Y550. F1000.           (PERFORM CUTTING AT F1000 UNTIL Y550.)
X350.                  (PERFORM CUTTING AT F1000 UNTIL X350.)
Y100.                  (PERFORM CUTTING AT F1000 UNTIL Y100.)
X100.                  (PERFORM CUTTING AT F1000 UNTIL X100.)
G00 G53 Z0.            (Z AXIS REFERENCE)
M5                     (SPINDLE STOP)
```

4.19. G53: Machine Coordinate System

It is used to program the axes according to the machine coordinates. The zero point of the machine coordinates is the reference point of the axes and cannot be changed.

Format:	**G53 X_ Y_ Z_ A/B/C_ U/V/W_**

X: The target value of the X axis according to the machine coordinates

Y: The target value of the Y axis according to the machine coordinates

Z: The target value of the Z axis according to the machine coordinates

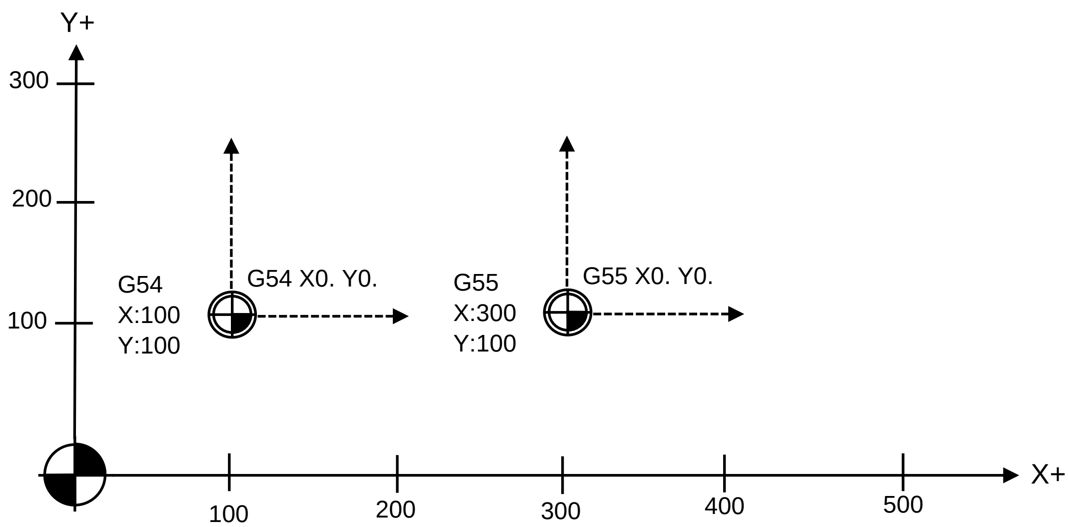
A/B/C: The target value of the A/B/C axis according to the machine coordinates

U/V/W: The target value of the U/V/W axis according to the machine coordinates

4.20. G54~G59.4: Workpiece Zero Selection

It selects the workpiece zero to be used. A total of 10 workpiece zeros are available. G54 is automatically selected at startup. These workpiece zeros are permanent and cannot be canceled. They can only be temporarily disabled using the G53 command.

Format:	**G54/G55/G56/G57/G58/G59/G59.1/G59.2/G59.3/G59.4**



Reference Point

4.21. G61/G64: Movement Type Selection

It changes the behavior of the axes during line transitions. In precision machining, the exact stop mode is used to verify the position of the axes at each line transition, maintaining accuracy during transitions. However, position verification at the end of each line requires all axes to stop, which results in time loss. In continuous cutting mode, position verification is not performed at the end of the lines. The system calculates a connection speed and acceleration that does not exceed the oscillation and acceleration values of each axis, and the transition is made according to these values. The part is completed faster compared to the exact stop mode, but deviations may occur during transitions.

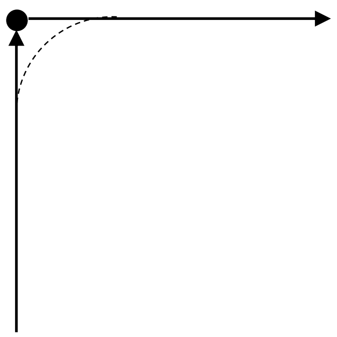
Format:	G61

Format:	G64

G61: Exact stop check

G64: Continuous cutting mode

Example:



——— The tool path generated with G61(Exact Stop mode)

- - - - The tool path generated with G64 (Continuous Cutting mode)

```
G61 G90 G01 Y100. F1000.   (EXACT STOP MODE)
X100
G64                        (CONTINUOUS CUTTING MODE)
```

4.22. G65: Single Macro Command Execution

Executes the macro commands provided by the system. Operations such as mathematical, trigonometric, and conditional branching can be performed.

Format:	G65 L_ P_ Q_ R_

L: Reference point selection

P: Macro command 1st value

Q: Macro command 2nd value

R: Macro command 3rd value

For detailed usage of macro commands, refer to the **Subprograms and Macro Commands** section.

4.23. G66/G67: Sequential Macro Execution On/Off

Redirects to the specified subprogram without processing all commands given in the main program. These commands must be executed within the subprogram by the user.

Format:	G66 P_

Format:	G67

G66: Sequential macro execution enabled

G67: Sequential macro execution disabled

P: Subprogram number (0~9999)

For detailed usage of macro commands, refer to the Subprograms and Macro Commands section.

4.24. G68/G69: Coordinate System Rotation

Rotates the coordinate system by the specified angle. If the value given with the R command is negative, the rotation occurs clockwise; if positive, the rotation occurs counterclockwise.

Format:	**G68 R_**

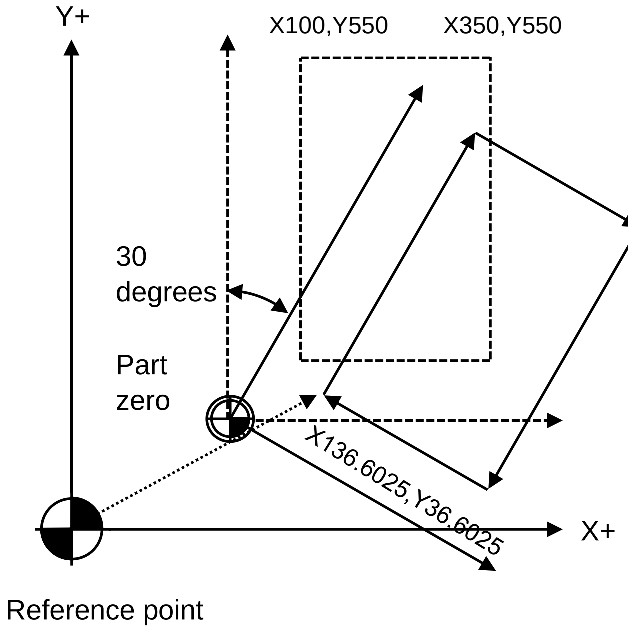
Format:	**G69**

G68: Coordinate system rotation enabled

G69: Coordinate system rotation disabled

R: Coordinate system rotation angle

Example:



```
G00 G53 Z0.          (SPINDLE CW ROTATION)
G68 R-30             (ROTATE THE COORDINATE SYSTEM BY 30
DEGREES)
M3 S1000             (SPINDLE CW ROTATION)
G90 G00 X100. Y100.  (RAPID MOVEMENT TO X100 Y100 POINT)
G90 G00 Z10.         (MOVE TO THE SAFE Z POSITION)
G01 Z-1. F500        (ENTER THE WORKPIECE AT F500)
Y550. F1000.         (PERFORM CUTTING AT F1000 UNTIL Y550.)
X350.                (PERFORM CUTTING AT F1000 UNTIL X350.)
Y100.                (PERFORM CUTTING AT F1000 UNTIL Y100.)
X100.                (PERFORM CUTTING AT F1000 UNTIL X100.)
G00 G53 Z0.          (Z AXIS REFERENCE)
G69                  (COORDINATE SYSTEM ROTATION DISABLED)
M5                   (SPINDLE STOP)
```

4.25. G73: Peck Drilling Cycle

This cycle performs high-speed and stepped drilling operations. It gradually descends to the bottom of the hole to evacuate the chips.

Format:	**G73 X_ Y_ Z_ R_ Q_ F_ K_**

X: Hole position X-axis coordinate

Y: Hole position Y-axis coordinate

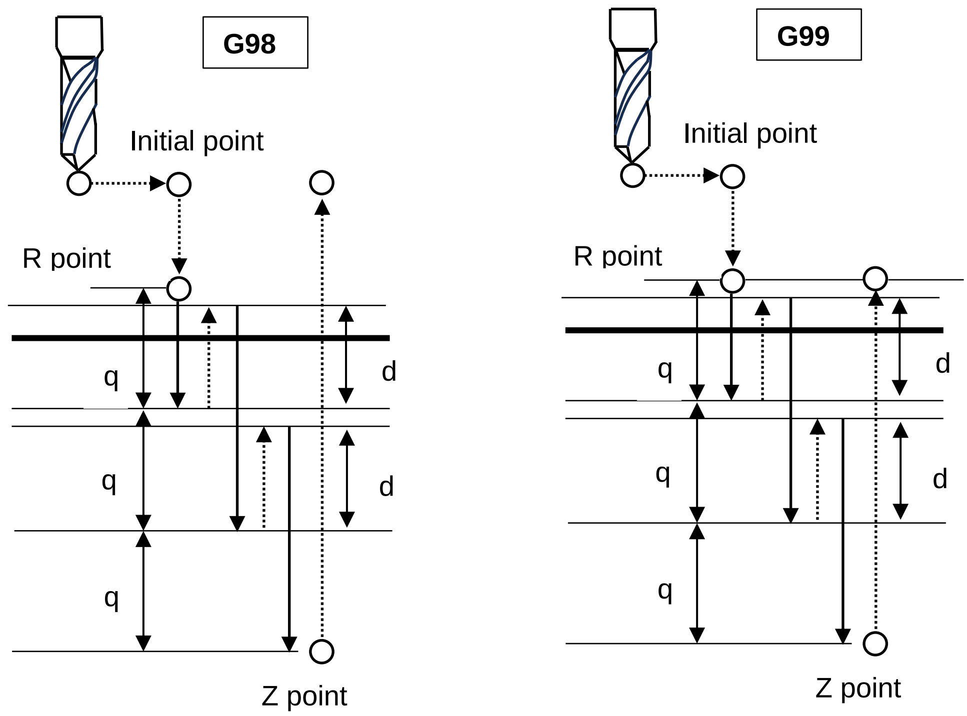
Z: Hole bottom coordinate

R: Safe Z coordinate to descend rapidly

Q: The distance to be drilled in each cutting movement

F: Cutting feedrate

K: Number of repetitions



The retract amount can be adjusted using the PRM312 parameter. After the tool operation is completed, a rapid retract is performed. Before issuing the G73 command, the spindle rotation should be activated with an M code. The M code given in the same line as the G73 command is processed once during the first positioning. However, the drilling operation proceeds without waiting for the completion of the M code. When multiple holes are drilled with the same command, the M code given in the same line as the G73 command is only executed once during the first operation.

In order to perform the drilling operation, one of the axes such as X, Y, Z, R, or other axes must be specified in the command.

When programming the G73 code, the R and Q values must be specified in the first line of consecutive G73 commands. In the subsequent drilling lines, these values are not mandatory.

The G73 code cannot be executed while the tool radius compensation is active. Use G40 to cancel the tool radius compensation before using the G73 code. Tool length compensation commands can be used with the G73 code.

Canned cycle commands should not be programmed on the same line as Group 1 G-codes (G0, G1, G2, G3). If they are, the canned cycle commands will be canceled.

Example:

```
M3 S2000 (SPINDLE CW ROTATION)
(DRILL A HOLE AT X100 Y-250.)
(RETURN TO THE R POINT AFTER COMPLETING THE DRILLING OPERATION)
G90 G99 G73 X100. Y-250. Z-80. R10. Q30. F120.
X200.        (2ND DRILLING OPERATION)
Y0.          (3RD DRILLING OPERATION)
X100.        (4TH DRILLING OPERATION)
G98 Y250.    (5TH DRILLING OPERATION)
G80 G53 Z0. ( GO TO Z AXIS REFERENCE)
M5           (SPINDLE STOP)
```

4.26. G74: Reverse Tapping Cycle for Milling

This cycle performs reverse tapping. When the bottom of the hole is reached, the spindle begins to rotate in the reverse direction.

Format:	**G74 X_ Y_ Z_ R_ P_ F_ K_**

X: Hole position X-axis coordinate

Y: Hole position Y-axis coordinate

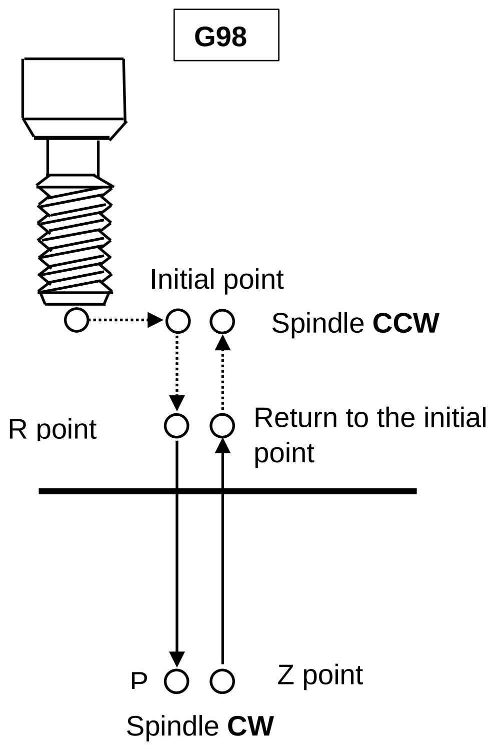
Z: Hole bottom coordinate

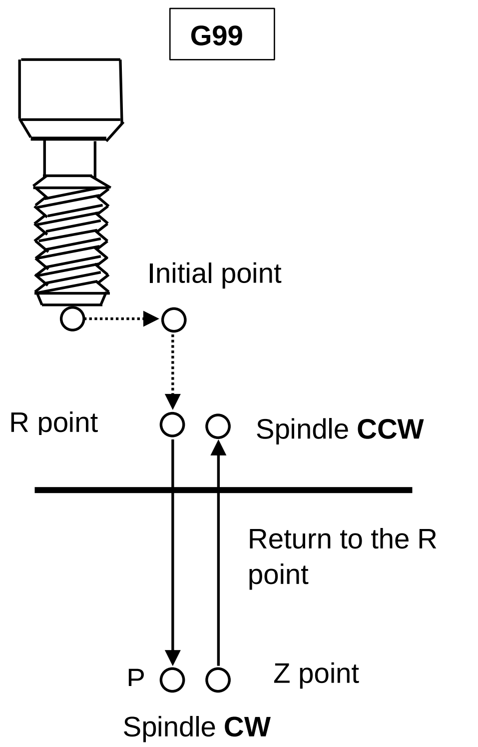
R: The Z safe rapid descent coordinate

P: The dwell time at the hole bottom

F: Cutting feed rate

K: Number of repetitions





The tapping operation is performed by rotating the spindle counterclockwise. When the hole depth is reached, the spindle is rotated clockwise for the retraction process. This creates the reverse thread.

During this process, the cutting feed rate setting cannot be adjusted. Program pause is disabled until the operation is completed.

Before issuing the G74 command, the spindle rotation must be enabled with an M code. The M code given on the same line as G74 is executed once during the initial positioning. However, the drilling operation proceeds before the M code is completed. If multiple taps are performed with the same command, the M code given on the same line as G74 is only executed once during the first operation.

One of the axes, such as X, Y, Z, R, or others, must be specified as a command for the drilling operation to be performed.

When programming the G74 code, the R value must be specified in the first line of consecutive G74 commands. It is not required in subsequent tapping lines.

While tool radius compensation is enabled, the G74 code cannot be executed. Before using the G74 code, disable tool radius compensation with G40. Tool length compensation commands can be used with G74.

Canned cycle commands should not be programmed on the same line as Group 1 G-codes (G0, G1, G2, G3). If they are, the canned cycle commands will be canceled.

Example:

```
M4 S100 (SPINDLE CCW ROTATION)
 (PERFORM THREAD CUTTING AT THE X100 Y-250 POINT)
 (RETURN TO THE R POINT AFTER THE DRILLING OPERATION)
G90 G99 G74 X100. Y-250. Z-150. R10. F120.
X200.        (2ND DRILLING OPERATION)
Y0.          (3RD DRILLING OPERATION)
X100.        (4TH DRILLING OPERATION)
G98 Y250.    (5TH DRILLING OPERATION)
G80 G53 Z0.  (GO TO Z AXIS REFERENCE)
M5           (SPINDLE STOP)
```

4.27. G76: Fine Boring Cycle for Milling

This cycle performs a precision boring operation. Upon reaching the bottom of the hole, the spindle stops at a predefined angle, and the tool tip is retracted from the machined surface before the retract operation is executed.

Format:	**G76 X_ Y_ Z_ R_ Q_ P_ F_ K_**

X: Hole position X-axis coordinate

Y: Hole position Y-axis coordinate

Z: Hole bottom coordinate

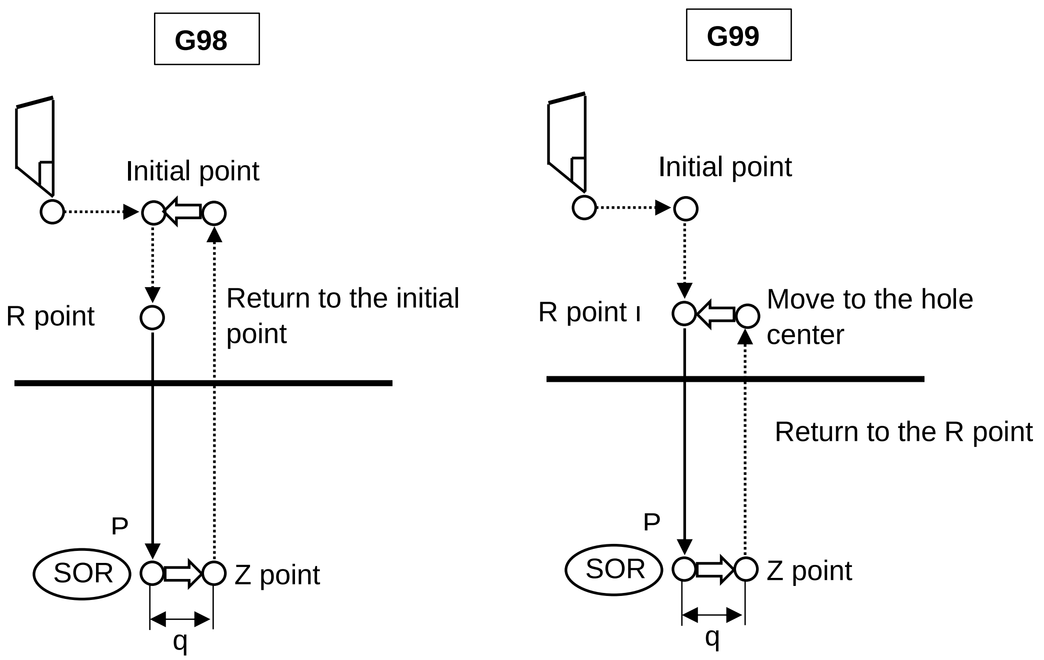
R: The Z safe rapid descent coordinate

Q: Retraction distance from the surface at the bottom of the hole

P: Dwell time at the bottom of the hole (unit: ms)

F: Cutting feedrate

K: Number of repetitions



Before programming the G76 command, spindle rotation must be activated with an M code.

The M code specified on the same line as the G76 command is executed once during the initial positioning. However, the drilling operation begins without waiting for the completion of the M code. If multiple boring operations are to be performed with the same command, the M code given with G76 is executed only once during the first operation.

To perform the boring operation, one of the X, Y, Z, R, or other axes must be specified as a command.

When programming the G76 code, the R and Q values must be specified in the first line of consecutive G76 commands. These are not mandatory in subsequent boring lines. Since the Q value in the G73 and G83 cycles is used for pecking depth, it should be used carefully when working with the G76 cycle. The previously specified Q value is retained in memory.

When tool radius compensation is active, the G76 code cannot be executed. Before using the G76 code, turn off tool radius compensation with G40. Tool length compensation commands can be used with G76.

Canned cycle commands should not be programmed on the same line as Group 1 G-codes (G0, G1, G2, G3). If they are, the canned cycle commands will be canceled.

For this command to function properly, spindle orientation parameter settings and ladder program implementation must be done. The axis and direction of the retract movement at the hole bottom can be adjusted with PRM314.

Example:

```
M3 S500 (SPINDLE CW ROTATION)
 (PERFORM BORING OPERATION AT X100 Y-250)
 (RETURN TO R POINT AFTER BORING OPERATION)
G90 G99 G76 X100. Y-250. Z-150. R10. Q5. P1000 F120.
X200.        (2ND DRILLING OPERATION)
Y0.          (3RD DRILLING OPERATION)
X100.        (4TH DRILLING OPERATION)
G98 Y250.    (5TH DRILLING OPERATION)
G80 G53 Z0.  (GO TO Z AXIS REFERENCE)
M5           (SPINDLE STOP)
```

4.28. G80: Cancel Canned Cycle

This command disables canned cycle commands. The given coordinates will be processed according to the selected last movement mode.

4.29. G81: Simple Drilling Cycle

This cycle performs normal drilling operation. Cutting motion is carried out with the given cutting feed rate until the hole bottom is reached.

Format:	**G81 X_ Y_ Z_ R_ F_ K_**

X: Hole position X-axis coordinate

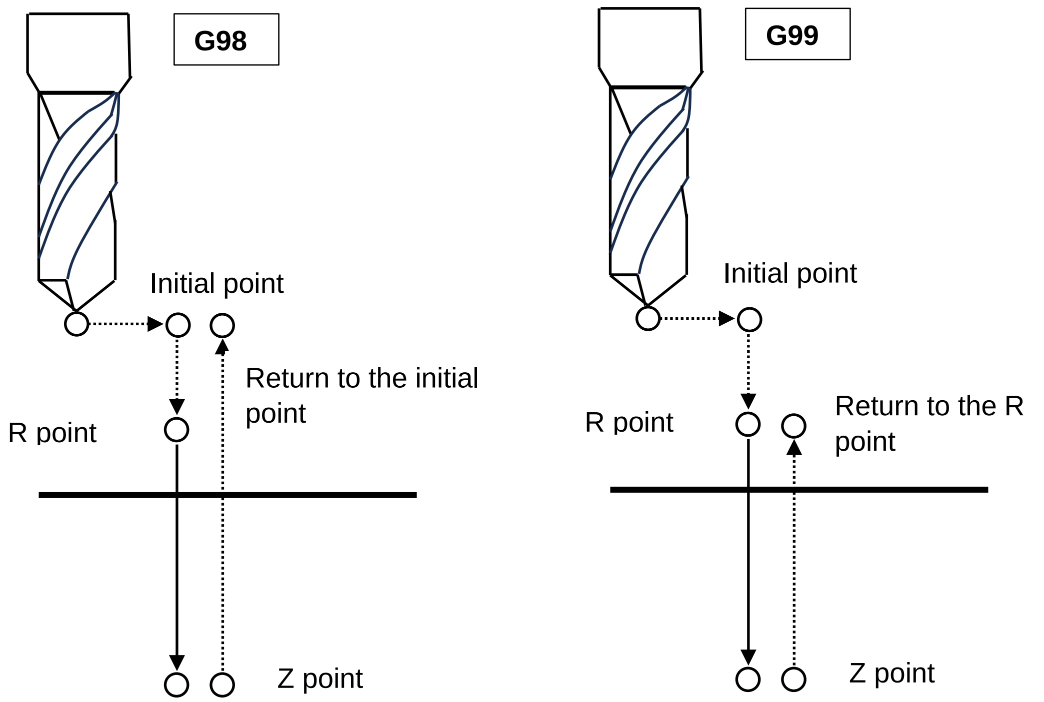
Y: Hole position Y-axis coordinate

Z: Hole bottom coordinate

R: The Z safe rapid descent coordinate

F: Cutting feedrate

K: Number of repetitions



Before executing the G81 command, spindle rotation must be enabled with an M code. The M code given in the same line as G81 is executed once during the first positioning. However, the drilling operation begins without waiting for the M code to complete. If multiple drilling operations are performed with the same command, the M code given in the same line as G81 is executed only once during the first operation.

To perform the boring operation, one of the X, Y, Z, R, or other axes must be specified as a command.

When programming the G81 code, the R value must be provided in the first line of consecutive G81 commands. It is not mandatory in subsequent drilling lines.

When tool radius compensation is active, the G81 code cannot be executed. Before using the G81 code, disable tool radius compensation with the G40 command. Tool length compensation commands can be used with G81.

Canned cycle commands should not be programmed on the same line as Group 1 G-codes (G0, G1, G2, G3). If they are, the canned cycle commands will be canceled.

Example:

```
M3 S2000 (SPINDLE CW ROTATION)
(DRILL A HOLE AT X100 Y-250)
(RETURN TO POINT R AFTER THE DRILLING OPERATION)
G90 G99 G81 X100. Y-250. Z-150. R10. F120.
X200.        (2ND DRILLING OPERATION)
Y0.          (3RD DRILLING OPERATION)
X100.        (4TH DRILLING OPERATION)
G98 Y250.    (5TH DRILLING OPERATION)
G80 G53 Z0.  (GO TO Z AXIS REFERENCE)
M5           (SPINDLE STOP)
```

4.30. G82: Drilling Cycle/Reverse Boring

This cycle performs a standard drilling operation. The cutting motion is executed at the specified cutting feed rate until reaching the bottom of the hole. After waiting for the duration specified by P at the bottom, the tool retracts with a rapid motion.

Format:	**G82 X_ Y_ Z_ R_ P_ F_ K_**

X: Hole position X-axis coordinate

Y: Hole position Y-axis coordinate

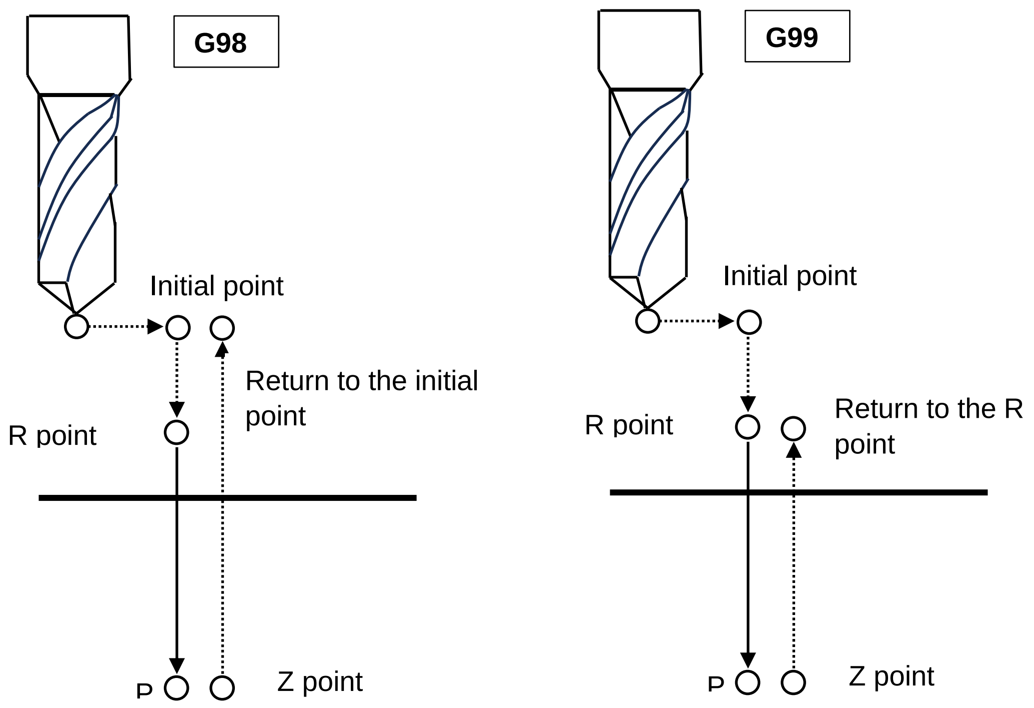
Z: Hole bottom coordinate

R: The Z safe rapid descent coordinate

P: Dwell time at the bottom of the hole (unit: ms)

F: Cutting feedrate

K: Number of repetitions



Before programming the G82 command, the spindle rotation must be activated using an M code.

The M code specified on the same line as the G82 command is executed once during the initial positioning. However, the drilling operation starts without waiting for the completion of the M code. If multiple drilling operations are to be performed with the same command, the M code specified with G82 is executed only once during the first operation.

To perform the drilling operation, one of the axes, such as X, Y, Z, R, or any other, must be specified as a command.

When programming the G82 code, the R value must be specified in the first line of consecutive G82 commands. It is not mandatory in the subsequent drilling lines.

The G82 code cannot be executed while the tool radius compensation is active. Disable the tool radius compensation with G40 before using the G82 code. Tool length compensation commands can be used with the G82 code.

Canned cycle commands must not be programmed on the same line as group 1 G codes (G0, G1, G2, G3). If programmed, the repetitive cycle commands will be canceled.

Example:

```
M3 S2000 (SPINDLE CW ROTATION)
(DRILL A HOLE AT X100 Y-250)
(RETURN TO POINT R AFTER THE DRILLING OPERATION)
G90 G99 G82 X100. Y-250. Z-150. R10. P1000 F120.
X200.        (2ND DRILLING OPERATION)
Y0.          (3RD DRILLING OPERATION)
X100.        (4TH DRILLING OPERATION)
G98 Y250.    (5TH DRILLING OPERATION)
G80 G53 Z0.  (GO TO Z AXIS REFERENCE)
M5           (SPINDLE STOP)
```

4.31. G83: Peck Drilling Cycle

This cycle performs a high-step drilling operation. It descends to the bottom of the hole incrementally to eject the chips.

Format:	**G83 X_ Y_ Z_ R_ Q_ F_ K_**

X: Hole position X-axis coordinate

Y: Hole position Y-axis coordinate

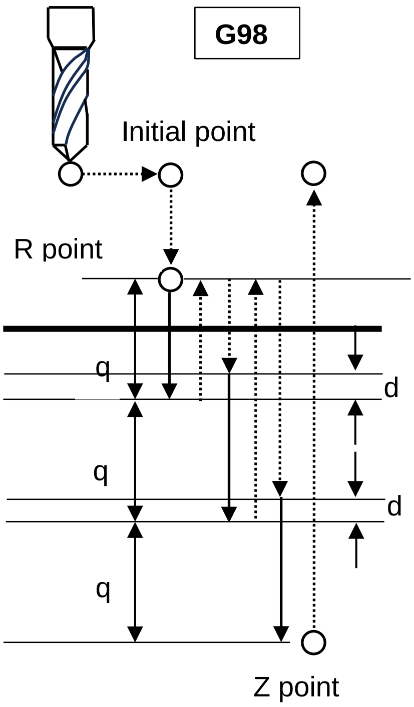
Z: Hole bottom coordinate

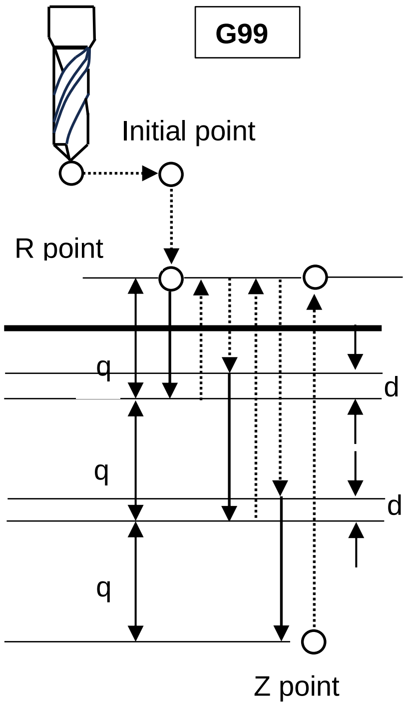
R: The Z safe rapid descent coordinate

Q: The distance to be drilled with each cutting movement

F: Cutting feedrate

K: Number of repetitions





The distance for the rapid approach of the tool can be adjusted using the PRM313 parameter. When a value of 1 mm is set for this parameter, the tool will rapidly move 1 mm above the final drilled coordinate. After completing the operation, the tool will quickly retract. Before executing the G83 command, the spindle rotation must be activated using an M code. The M code given in the same line as G83 is executed only once during the first operation. However, the drilling operation proceeds without waiting for the completion of the M code. If multiple holes are drilled with the same command, the M code in the same line as G83 will only be executed during the first operation.

One of the axes (X, Y, Z, R, or others) must be specified as a command for the drilling operation to be performed.

When programming the G83 code, the R and Q values must be specified in the first line of consecutive G83 commands. These values are not required in subsequent drilling lines.

When tool radius compensation is active, the G83 code cannot be executed. To use G83, deactivate tool radius compensation with the G40 command. Tool length compensation commands can be used with G83.

Canned cycle commands must not be programmed on the same line as group 1 G codes (G0, G1, G2, G3). If programmed, the repetitive cycle commands will be canceled.

Example:

```
M3 S2000 (SPINDLE CW ROTATION)
 (DRILL A HOLE AT X100 Y-250)
 (RETURN TO POINT R AFTER THE DRILLING OPERATION)
G90 G99 G83 X100. Y-250. Z-80. R10. Q30. F120.
X200.        (2ND DRILLING OPERATION)
Y0.          (3RD DRILLING OPERATION)
X100.        (4TH DRILLING OPERATION)
G98 Y250.    (5TH DRILLING OPERATION)
G80 G53 Z0.  (GO TO Z AXIS REFERENCE)
M5           (SPINDLE STOP)
```

4.32. G84: Tapping Cycle

This cycle performs tapping operations. Once the hole bottom is reached, the spindle starts rotating in the reverse direction.

Format:	**G84 X_ Y_ Z_ R_ P_ F_ K_**

X: Hole position X-axis coordinate

Y: Hole position Y-axis coordinate

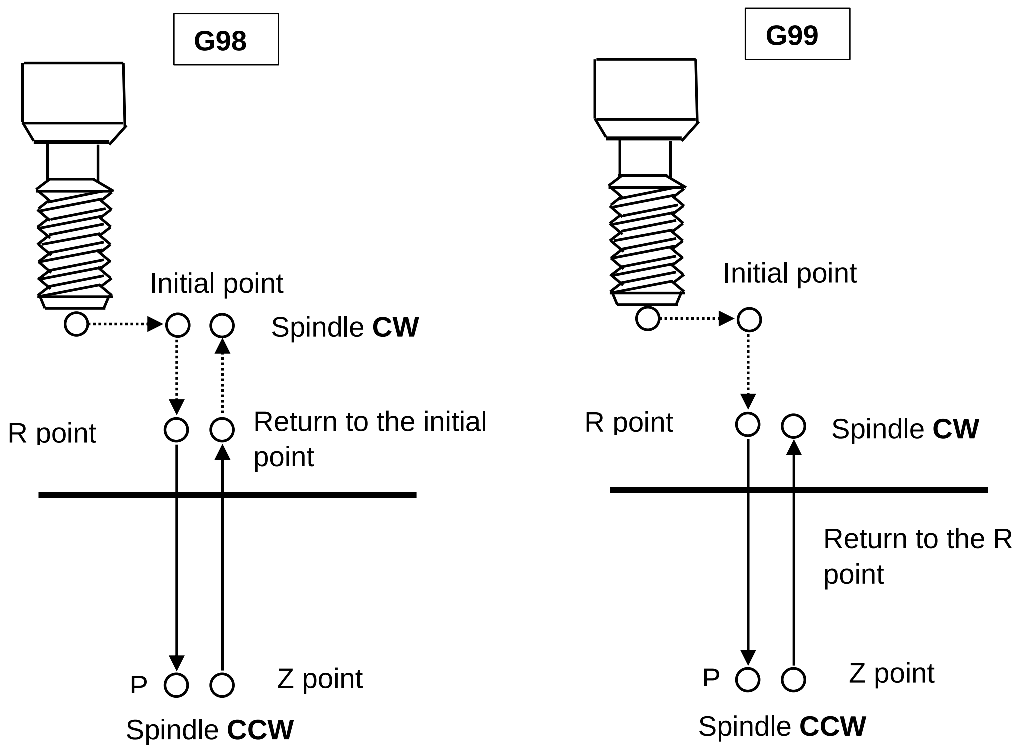
Z: Hole bottom coordinate

R: The Z safe rapid descent coordinate

P: Dwell time at the bottom of the hole (unit: ms)

F: Cutting feedrate

K: Number of repetitions



The tapping operation is performed by rotating the spindle in the clockwise direction. When the hole bottom is reached, the spindle is rotated counterclockwise for the retraction movement, thus completing the tapping process.

During this process, the cutting feed rate cannot be adjusted. Program pause is disabled until the operation is completed.

Before issuing the G84 command, the spindle rotation must be set with an M code. The M code provided on the same line as G84 will be processed once during the first positioning. However, the hole-making process will begin without waiting for the M code to complete. If multiple taps are performed with the same command, the M code given on the same line as G84 will only be executed once during the first operation.

To perform the hole drilling operation, one of the coordinates X, Y, Z, R, or other axes must be specified as a command.

When programming the G84 code, the R value must be specified in the first line of successive G84 commands. For subsequent tapping lines, it is not mandatory.

G84 code cannot be executed when tool radius compensation is enabled. Before using G84, turn off tool radius compensation with the G40 code. Tool length offset commands can be used with G84.

Canned cycle commands must not be programmed on the same line as group 1 G codes (G0, G1, G2, G3). If programmed, the repetitive cycle commands will be canceled.

Example:

```
M3 S100 (SPINDLE CW ROTATION)
 (PERFORM TAPPING OPERATION AT X100 Y-250)
 (RETURN TO R POINT AFTER OPERATION)
G90 G99 G84 X100. Y-250. Z-150. R10. P1000 F120.
X200.        (2ND DRILLING OPERATION)
Y0.          (3RD DRILLING OPERATION)
X100.        (4TH DRILLING OPERATION)
G98 Y250.    (5TH DRILLING OPERATION)
G80 G53 Z0.  (GO TO Z AXIS REFERENCE)
M5           (SPINDLE STOP)
```

4.33. G85: Boring Cycle

This cycle performs boring. When the bottom of the hole is reached, cutting feed is used for retraction.

Format:	**G85 X_ Y_ Z_ R_ F_ K_**

X: Hole position X-axis coordinate

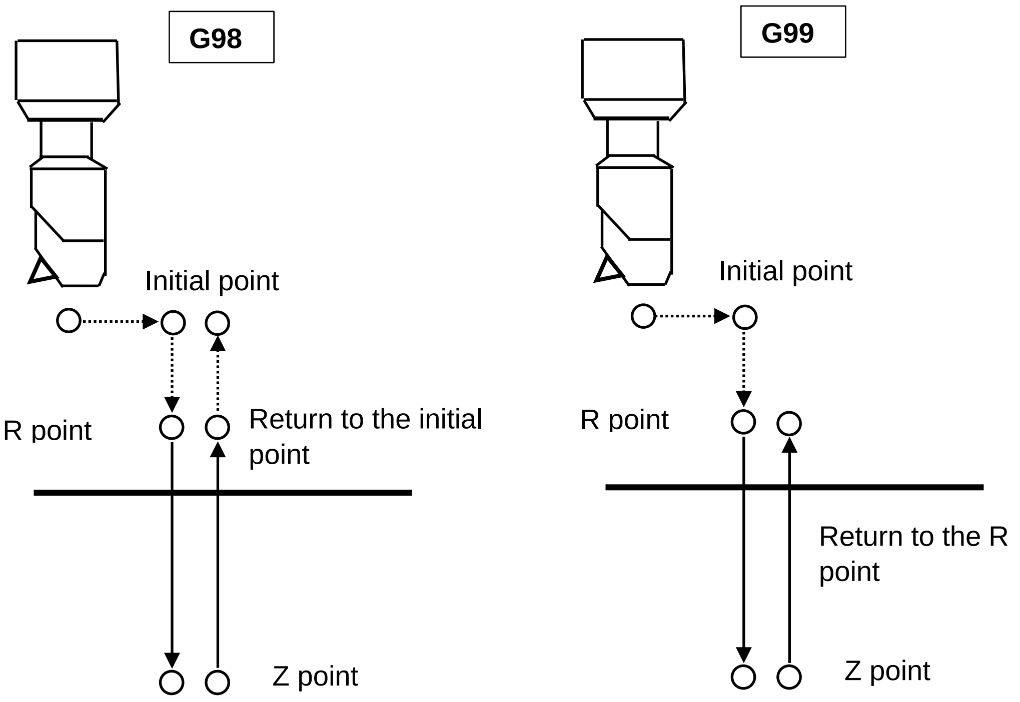
Y: Hole position Y-axis coordinate

Z: Hole bottom coordinate

R: The Z safe rapid descent coordinate

F: Cutting feedrate

K: Number of repetitions



Before issuing the G85 command, the spindle rotation must be activated with an M code. The M code given on the same line as G85 is processed once during the first positioning. However, the drilling operation proceeds without waiting for the completion of the M code. If multiple threading operations are performed with the same command, the M code given on the same line as G85 is executed only once during the first operation.

For the boring operation to be carried out, one of the axes, X, Y, Z, R, or others, must be specified as a command.

When programming the G85 code, the R value must be specified in the first line of consecutive G85 commands. It is not mandatory to provide the R value in the following boring lines.

When tool radius compensation is enabled, the G85 code cannot be executed. Before using the G85 code, disable tool radius compensation by using the G40 command. Tool length compensation commands can be used with G85.

Canned cycle commands must not be programmed on the same line as group 1 G codes (G0, G1, G2, G3). If programmed, the repetitive cycle commands will be canceled.

Example:

```
M3 S100 (SPINDLE CW ROTATION)
(PERFORM BORING OPERATION AT X100 Y-250)
(RETURN TO THE R POINT AFTER THE OPERATION)
G90 G99 G85 X100. Y-250. Z-150. R10. F120.
X200.       (2ND DRILLING OPERATION)
Y0.         (3RD DRILLING OPERATION)
X100.       (4TH DRILLING OPERATION)
G98 Y250.   (5TH DRILLING OPERATION)
G80 G53 Z0. (GO TO Z AXIS REFERENCE)
M5          (SPINDLE STOP)
```

4.34. G86: Boring Cycle

This cycle performs the boring operation. When the hole bottom is reached, the spindle stops, and the retraction is carried out in this way.

Format:	**G86 X_ Y_ Z_ R_ F_ K_**

X: Hole position X-axis coordinate

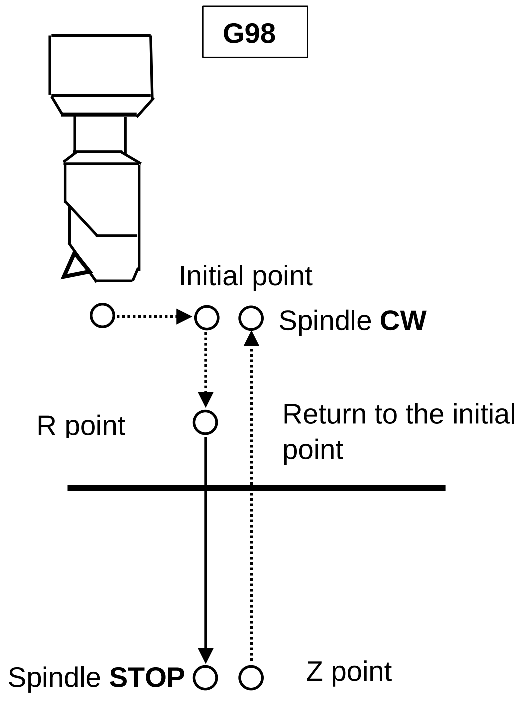
Y: Hole position Y-axis coordinate

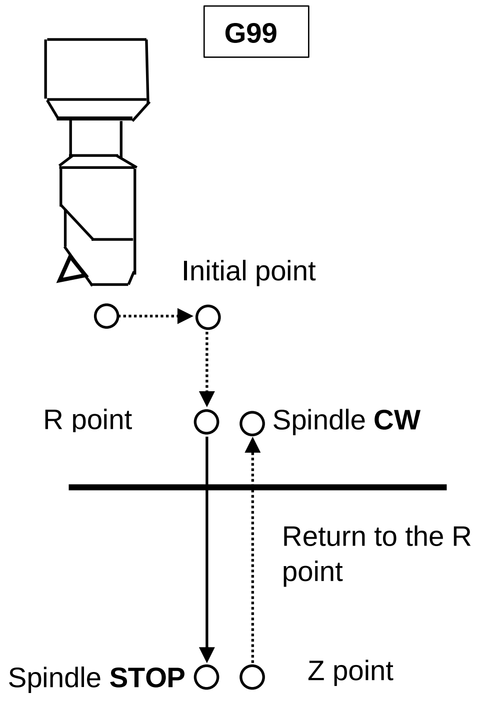
Z: Hole bottom coordinate

R: The Z safe rapid descent coordinate

F: Cutting feedrate

K: Number of repetitions





Before issuing the G86 command, the spindle rotation must be provided with an M code. The M code given on the same line as G86 is processed once during the initial positioning. However, the drilling operation begins without waiting for the completion of the M code. If multiple thread cutting operations are performed with the same command, the M code given with G86 on the same line will only be executed once during the first operation.

To perform the boring operation, one of the axes (X, Y, Z, R, or other axes) must be specified in the command.

When programming the G86 code, the R value must be specified in the first line of consecutive G86 commands. It is not required in subsequent boring lines.

The G86 code cannot be executed when tool radius compensation is enabled. Before using the G86 code, disable tool radius compensation with G40. Tool length offset commands can be used with G86.

Canned cycle commands must not be programmed on the same line as group 1 G codes (G0, G1, G2, G3). If programmed, the repetitive cycle commands will be canceled.

Example:

```
M3 S2000 (SPINDLE CW ROTATION)
(PERFORM BORING OPERATION AT X100 Y-250)
(RETURN TO THE R POINT AFTER THE OPERATION)
G90 G99 G86 X100. Y-250. Z-150. R10. F120.
X200.         (2ND DRILLING OPERATION)
Y0.           (3RD DRILLING OPERATION)
X100.         (4TH DRILLING OPERATION)
G98 Y250.     (5TH DRILLING OPERATION)
G80 G53 Z0.   (GO TO Z AXIS REFERENCE)
M5            (SPINDLE STOP)
```

4.35. G87: Boring Cycle, Backboring

This cycle performs a boring operation in the reverse direction. After a rapid move to the hole center, the spindle stops at a predefined angle. Then, the tool moves in the opposite direction for retraction, followed by a rapid move to the R point, and the operation continues. After a retraction move toward the tool direction to the hole center, the spindle rotates. This process continues until the Z coordinate is reached. Upon reaching the Z coordinate, the spindle repositions according to the predefined angle, and then the retraction movement is performed in the opposite direction of the tool. This way, a rapid movement towards the starting point is made. Once the starting point is reached, another retraction movement is made in the tool direction, and the spindle rotation is released.

Format:	**G87 X_ Y_ Z_ R_ Q_ P_ F_ K_**

X: Hole position X-axis coordinate

Y: Hole position Y-axis coordinate

Z: Hole bottom coordinate

R: The Z safe rapid descent coordinate

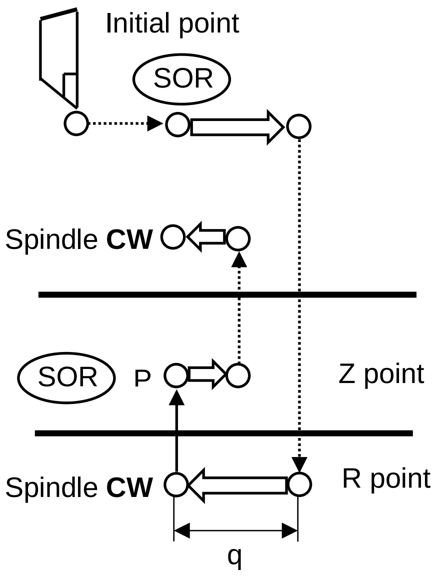
Q: Distance from the surface at the bottom of the hole

P: Dwell time at the bottom of the hole (unit: ms)

F: Cutting feedrate

K: Number of repetitions

G98	**G99**



Unavailable

The M-code must be issued to start the spindle rotation before executing the G87 command.

The M-code specified on the same line as the G87 command is executed once during the initial positioning. However, the drilling operation begins without waiting for the M-code to complete. If multiple boring operations are to be performed with the same command, the M-code specified with the G87 command is executed only once during the first operation.

To perform the drilling operation, one of the axes, such as X, Y, Z, R, or any other, must be specified in the command.

When programming the G87 code, the R and Q values must be specified in the first line of consecutive G87 commands. In subsequent boring lines, this is not mandatory. Since the Q value is used as the pecking distance in G73 and G83 cycles, care should be taken when using it with the G87 cycle. The previously defined Q value is retained in memory.

The G87 code cannot be executed while the tool radius compensation is active. Deactivate the tool radius compensation using G40 before issuing the G87 code. Tool length compensation commands can be used with the G87 code.

Canned cycle commands must not be programmed on the same line as group 1 G codes (G0, G1, G2, G3). If programmed, the repetitive cycle commands will be canceled.

For this command to work properly, spindle orientation parameter settings and ladder program implementation must be completed. The axis and direction of the retraction movement at the bottom of the hole can be adjusted with PRM314.

Example:

```
M3 S500 (SPINDLE CW ROTATION)
(PERFORM BORING OPERATION AT X100 Y-250)
(RETURN TO R POINT AFTER THE BORING OPERATION)
G90 G98 G87 X100. Y-250. Z10. R-50. Q5. P1000 F120.
X200.        (2ND DRILLING OPERATION)
Y0.          (3RD DRILLING OPERATION)
X100.        (4TH DRILLING OPERATION)
G98 Y250.    (5TH DRILLING OPERATION)
G80 G53 Z0.  (GO TO Z AXIS REFERENCE)
M5           (SPINDLE STOP)
```

4.36. G88: Boring Cycle

This cycle performs a boring operation. When the bottom of the hole is reached, the spindle is stopped after waiting for the P value, and manual retraction of the hole axis is expected. Once the retraction is completed, the spindle is turned on again and the program proceeds to the next line.

Format:	G88 X_ Y_ Z_ R_ P_ F_ K_

X: Hole position X-axis coordinate

Y: Hole position Y-axis coordinate

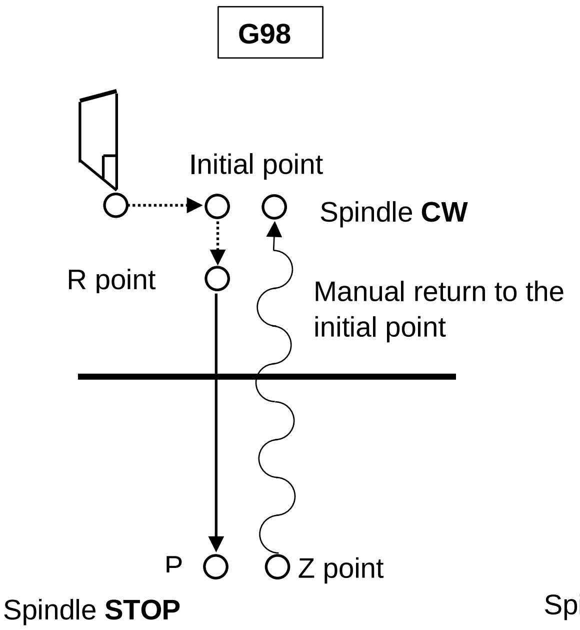
Z: Hole bottom coordinate

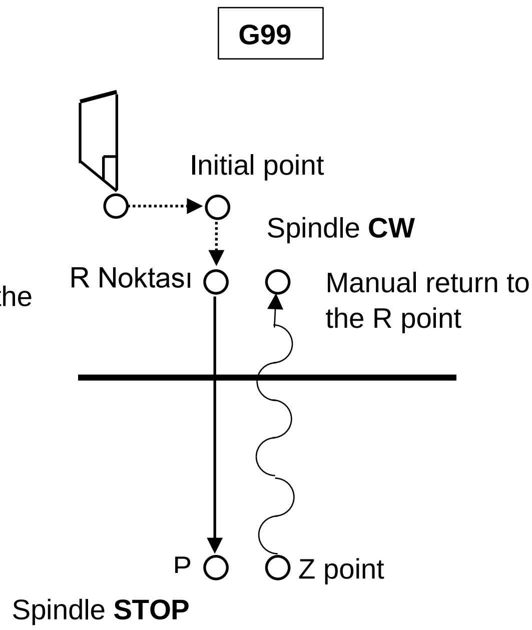
R: The Z safe rapid descent coordinate

P: Dwell time at the bottom of the hole (unit: ms)

F: Cutting feedrate

K: Number of repetitions





Before issuing the G88 command, the spindle rotation must be set using an M code.

The M code given in the same line as G88 is processed once during the first positioning. However, the drilling operation starts before the M code is completed. If multiple boring operations are to be performed with the same command, the M code given in the same line as G88 is executed only once during the first operation.

One of the axes X, Y, Z, R, or other axes must be specified as a command in order to perform the boring operation.

When programming the G88 code, the R value must be specified in the first line of successive G88 commands. It is not mandatory in the subsequent boring lines.

G88 code cannot be executed while tool radius compensation is enabled. Turn off tool radius compensation with G40 before using the G88 code. Tool length compensation commands can be used with G88.

Canned cycle commands must not be programmed on the same line as group 1 G codes (G0, G1, G2, G3). If programmed, the repetitive cycle commands will be canceled.

Example:

```
M3 S500 (SPINDLE CW ROTATION)
(PERFORM BORING OPERATION AT POINT X100 Y-250)
(RETURN TO POINT R AFTER BORING OPERATION)
G90 G99 G88 X100. Y-250. Z-150. R10. P1000 F120.
X200.        (2ND DRILLING OPERATION)
Y0.          (3RD DRILLING OPERATION)
X100.        (4TH DRILLING OPERATION)
G98 Y250.    (5TH DRILLING OPERATION)
G80 G53 Z0.  (GO TO Z AXIS REFERENCE)
M5           (SPINDLE STOP)
```

4.37. G89: Boring Cycle

This cycle performs a boring operation. Upon reaching the bottom of the hole, the cutting speed is used for retraction. It is almost identical to the G85 command, with the only difference being an additional dwell at the bottom of the hole.

Format:	**G89 X_ Y_ Z_ R_ P_ F_ K_**

X: Hole position X-axis coordinate

Y: Hole position Y-axis coordinate

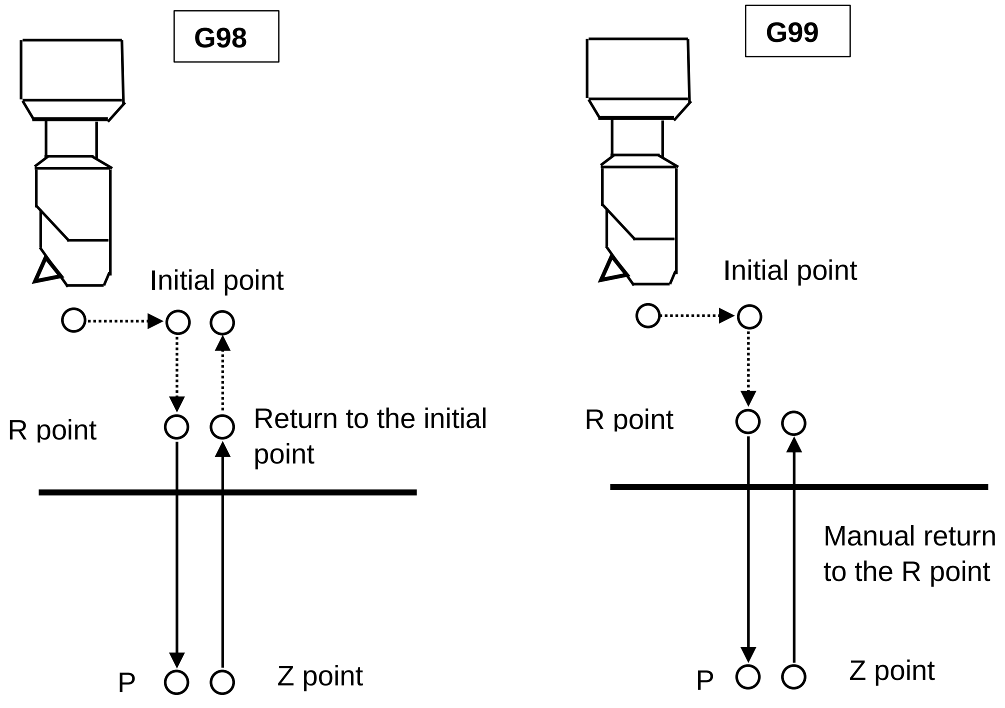
Z: Hole bottom coordinate

R: The Z safe rapid descent coordinate

P: Dwell time at the bottom of the hole (unit: ms)

F: Cutting feedrate

K: Number of repetitions



Before issuing the G89 command, spindle rotation must be activated using an M code. The M code provided on the same line as the G89 command is processed once during the initial positioning. However, the drilling operation proceeds without waiting for the completion of the M code. If multiple tapping operations are to be performed with the same command, the M code on the same line as G89 is executed only once during the first operation.

To perform the boring operation, one of the axes, such as X, Y, Z, R, or others, must be specified as a command.

When programming the G89 code, the R value must be specified in the first line of consecutive G89 commands. In subsequent boring lines, it is not mandatory.

The G89 code cannot be executed while tool radius compensation is active. Before using the G89 code, cancel the tool radius compensation with G40. Tool length compensation commands can be used with G89.

Canned cycle commands must not be programmed on the same line as group 1 G codes (G0, G1, G2, G3). If programmed, the repetitive cycle commands will be canceled.

Example:

```
M3 S100 (SPINDLE CW ROTATION)
(PERFORM BORING OPERATION AT POINT X100 Y-250)
(RETURN TO POINT R AFTER BORING OPERATION)
G90 G99 G89 X100. Y-250. Z-150. R10. P1000 F120.
X200.        (2ND DRILLING OPERATION)
Y0.          (3RD DRILLING OPERATION)
X100.        (4TH DRILLING OPERATION)
G98 Y250.    (5TH DRILLING OPERATION)
G80 G53 Z0.  (GO TO Z AXIS REFERENCE)
M5           (SPINDLE STOP)
```

4.38. G90/G91: Distance Command Type Selection (Absolute, Incremental)

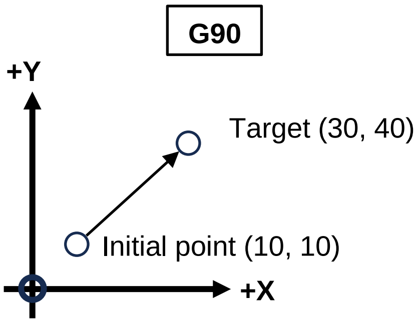
Specifies whether the target values for axis movements will be programmed as absolute or incremental.In absolute programming mode, the axes move to the coordinates specified by the command.In incremental mode, the axes move by the distance specified in the command relative to their current position.

Format:	**G90**

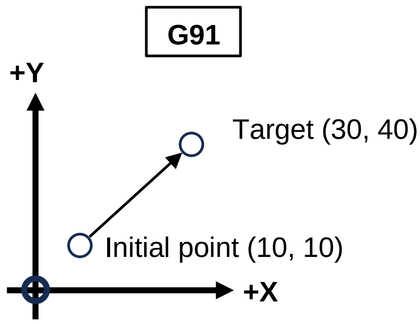
G90: The target coordinates are programmed as absolute

Format:	**G91**

G91: The target coordinates are programmed as incremental

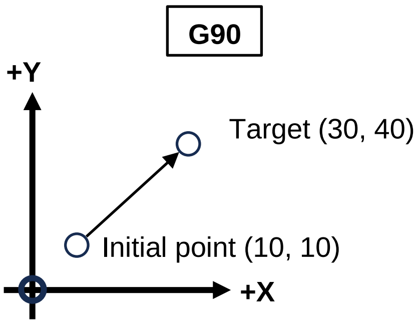


G90 X30. Y40.

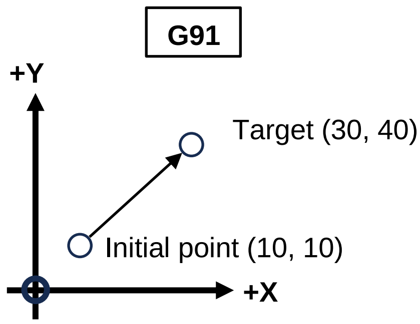


G91 X20. Y30.

In the lathe model, if the U/V/W axes are not defined, the U/V/W commands are used as incremental commands for the X/Y/Z axes. Both in G90 and G91 modes, the U/V/W commands serve as the incremental commands for the X/Y/Z axes, respectively.



G90 U20. V30.



G91 U20. V30.

4.39. G92: Reference Point Shifting

It is used to set the current position of the axes to the specified values. With this command, the shifted coordinate system shifts all part zeros (G54~G59.4). The shift values are persistent and must be manually cleared.

Format:	**G92 X_ Y_ Z_ A/B/C_ U/V/W_**

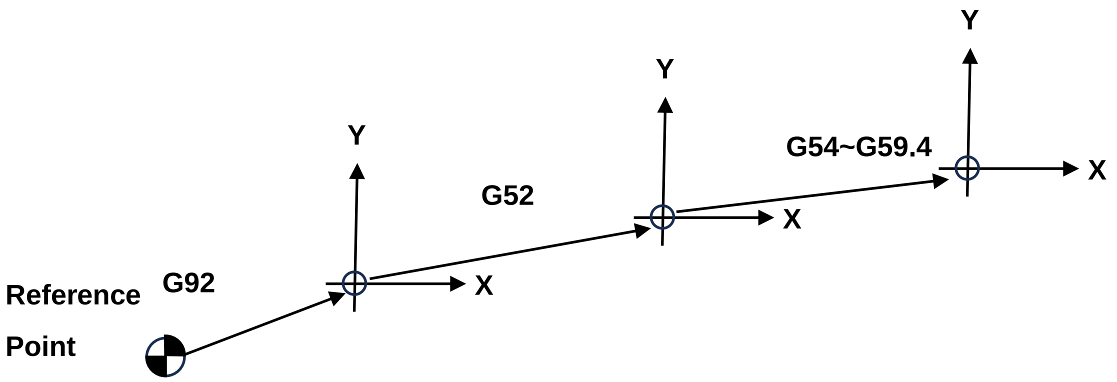
X: The absolute coordinate at which the X-axis is intended to be set

Y: The absolute coordinate at which the Y-axis is intended to be set

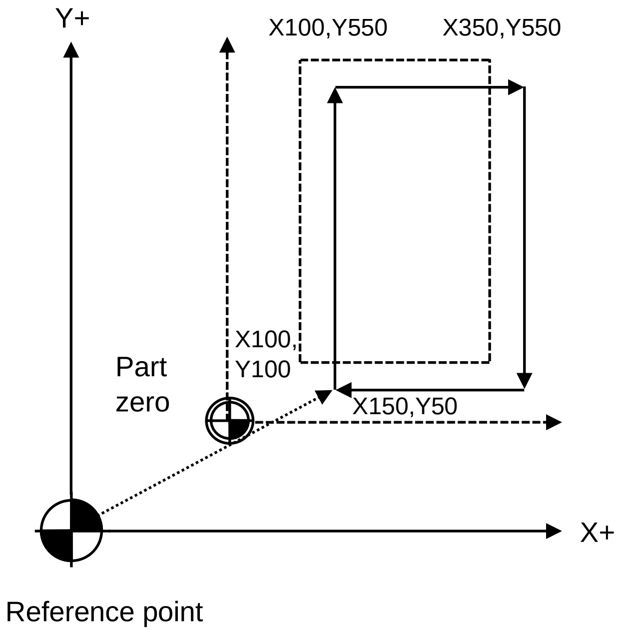
Z: The absolute coordinate at which the Z-axis is intended to be set

A/B/C: The absolute coordinate at which the A/B/C axes are intended to be set

U/V/W: The absolute coordinate at which the U/V/W axes are intended to be set



Example:




```
G00 G53 Z0.          (Z AXIS REFERENCE)
M3 S1000             (SPINDLE CW ROTATION)
G90 G00 X150. Y50.   (RAPID MOVEMENT TO X150 Y50 POINT)
G92 X100. Y100.      (SHIFT COORDINATE SYSTEM)
G90 G00 Z10.         (RAPID MOVE TO Z SAFE POSITION)
G01 Z-1. F500        (ENTER THE WORKPIECE AT F500)
Y550. F1000.         (PERFORM CUTTING AT F1000 UNTIL Y550.)
X350.                (PERFORM CUTTING AT F1000 UNTIL X350.)
Y100.                (PERFORM CUTTING AT F1000 UNTIL Y100.)
X100.                (PERFORM CUTTING AT F1000 UNTIL X100.)
G00 G53 Z0.          (GO TO Z AXIS REFERENCE)
M5                   (SPINDLE STOP)
```

4.40. G94/G95: Feed Mode (Units/Minute, Units/Rev)

Format:	**G94 F_**

Format:	**G95 F_**

G94: Feedrate per minute (Units: mm/min or in/min)

G95: Feedrate per revolution (Units: mm/rev or in/rev)

4.41. G96/G97:Constant Surface Speed Control On/Off

Format:	**G96 S___**

G96: Constant surface speed control On

S: Surface speed. Unit: m/min or ft/min

Format:	**G97 S___**

G97: Constant surface speed control Off

S: Spindle speed. Unit: rpm/min.

During a turning operation, when it is necessary to maintain a constant surface speed, the surface speed should be specified using G96. The spindle speed is automatically adjusted based on the X-axis absolute coordinate in such a way that the surface speed is kept constant. Therefore, for these codes to function correctly, the X-axis must be zeroed at the center of the workpiece. The relationship between the spindle speed, surface speed, and the X-axis absolute coordinate is as follows:

$$N = \frac{1000 * V}{\pi * D}$$

N: Spindle speed

V: Surface speed

D: Distance to the part center (X-axis absolute position)

To prevent the spindle speed from increasing excessively as it approaches lower diameters, the G92 S_ codes can be used. These codes allow for setting the maximum spindle speed.

When the **G96** code is used together with the **G95** code, the spindle speed must be specified by adding an **S** code next to the **G95** code.

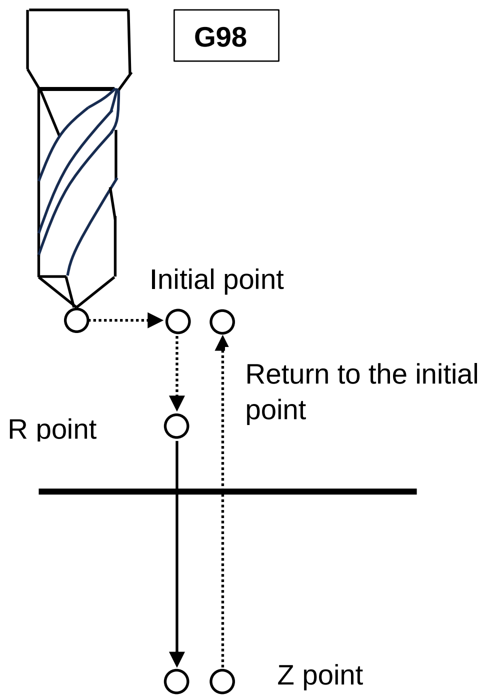
4.42. G98/G99: The position for retracting after cycles

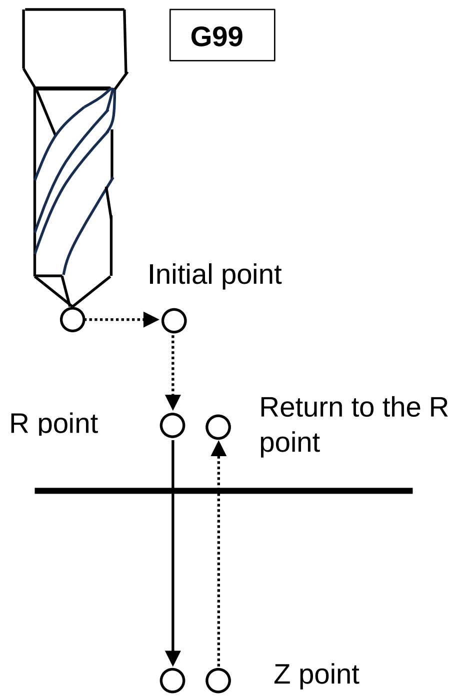
Format:	**G98**

G98: The hole axis is retracted to the starting position after cycles

Format:	**G99**

G99: The hole axis is retracted to the **R** position after cycles





5. M Code Explanations

5.1. M00: Program Stop

Pauses automatic operation. All data in memory is retained without being cleared. Automatic operation can be resumed from the halted point using automatic start.

Format:	**M00**

5.2. M01: Optional program stop

Pauses automatic operation only when the M01/OPS option is enabled. Otherwise, it skips the line and continues operation. When a pause occurs, all data in memory is retained without being cleared. Automatic operation can be resumed from the halted point using automatic start.

Format:	**M01**

5.3. M02/M30: End of program

Ends automatic operation and moves the cursor to the beginning of the selected program. Increments the part counter and stops the cycle time counter. It should be written in the last line of the program. Codes written after this command will not be processed.

Format:	**M02/M30**

5.4. M98: Subprogram call

Performs the subprogram call operation. The subprogram number is specified with P, and the repetition count is specified with L. M99 must be added at the last line of the subprogram to return to the main program.

Format:	M98 P_ L_

P: Subprogram number (0~9999)

L: Repeat count (1~9999)

If the subprogram is to be executed only once, the **L** command may be omitted.

5.5. M99: Return from subprogram

Used to return from a subprogram to the main program. If this command is written at the end of the main program, the cursor will be positioned at the beginning of the program after this command, resulting in continuous operation. The user must manually stop the automatic operation.

Format:	M99

6. Subprograms and Macro Commands

6.1. Subprograms

Subprograms used in the system pause the selected program at the current line and redirect to the first line of the relevant subprogram. It then returns to the line where the main program left off by encountering the M99 command inside the subprogram and continues from there. A group of operations that need to be repeated multiple times can be consolidated into a subprogram, such as tool change subprogram, table change subprogram, or tool offset subprogram.

The file names of subprograms should be in the **Oxxxx.cnc** format, where "xxxx" represents a numerical value. For example, the file O9001.cnc defines subprogram number 9001.

Correct examples of subprogram file names:

O0001.cnc

O1234.cnc

O9001.cnc

O9009.cnc

Incorrect examples of subprogram file names:

0001.cnc

4.cnc

SUB1.cnc

ABC.cnc

The last line of a subprogram should contain the **M99** command. A maximum of two nested subprogram calls can be made in the system. Macro commands can be used within subprograms. Other commands on the lines redirected to the subprogram will not be processed and will be directed to the subprogram.

6.1.1. Calling a Subprogram with M98

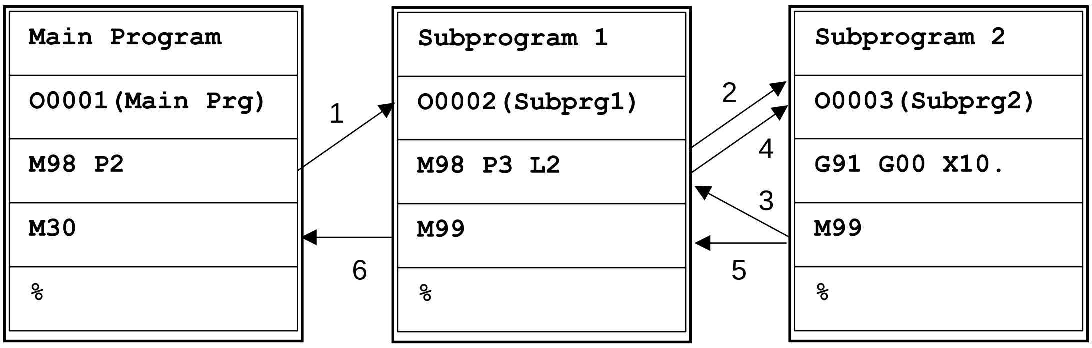
The subprogram can be called using the M98 command.

Format:	**M98 P_ L_**

P: Subprogram number (0~9999)

L: Repeat count (1~9999)

If the subprogram is to be executed only once, the L command may be omitted.



Example of nested subprogram call

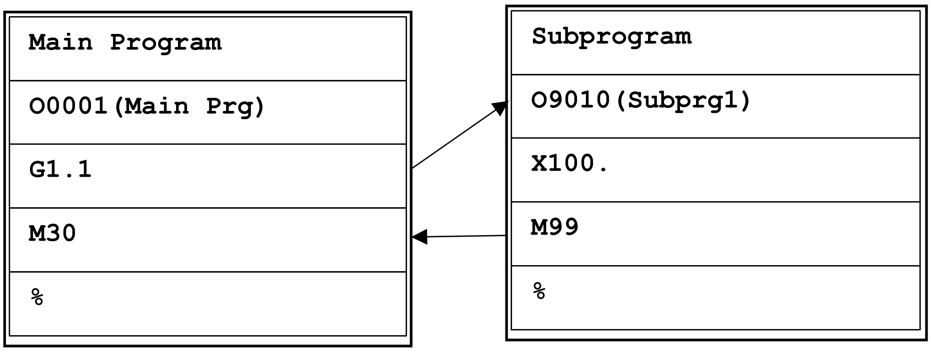
6.1.2. Calling Subprograms with User-Defined G Code

A subprogram can be called using a user-defined G code. Up to 10 custom G codes can be created. Each custom G code directs to subprograms starting from the O9010 subprogram in a sequential order. The G codes that will be used to call the subprograms must be entered into the PRM0-PRM9 parameters. These parameters should be specified in 0.0 format. For example, if the O9010.cnc subprogram is to be called with the G51 code, the value 510 should be entered into PRM0. Similarly, if the O9011.cnc subprogram is to be called with the G51.1 code, the value 511 should be entered into PRM1.

The values entered into these parameters mask the system-defined G codes. When G0 is directed to a subprogram, the G0 code loses all its functions within the system.

Example:

PRM0: 11

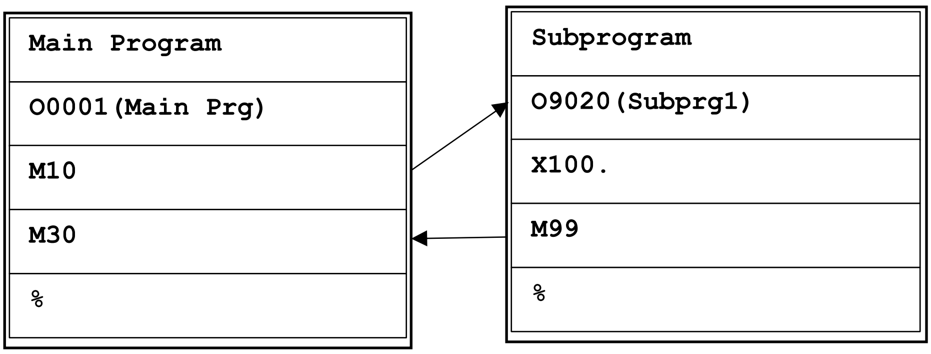


6.1.3. Calling Subprograms with User-Defined G Code

A subprogram call can be made using a user-defined M code. Up to 10 custom M codes can be created. Each custom M code will sequentially direct to subprograms starting from O9020. The M codes to be directed to the subprogram must be entered into the PRM10-PRM19 parameters. Values between 0 and 255 can be assigned. The values written in these parameters mask the system-defined M codes.

Example:

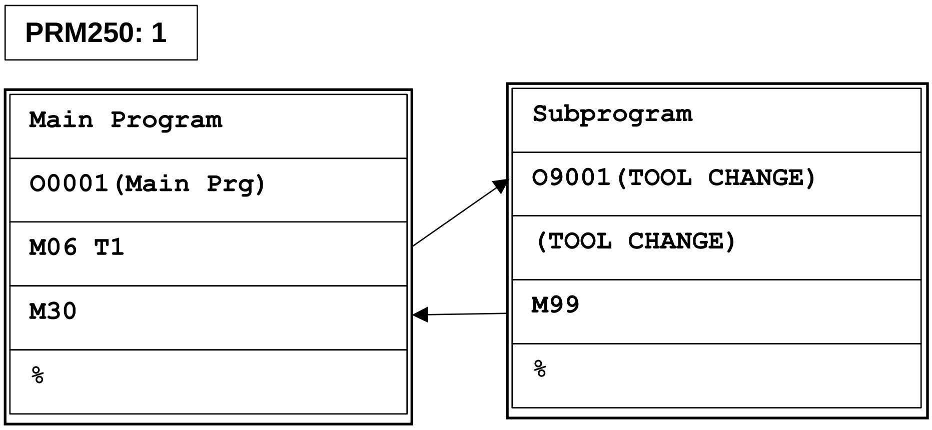
PRM10: 10



6.1.4. Calling the O9001 Subprogram with M06 Code

M06 code is used by many machine manufacturers as the tool change command. Similarly, O9001 is frequently used as the tool change subprogram number. When the parameter (PRM317) for automatically calling O9001 on M06 is activated (set to 1), the system will automatically direct to the O9001 subprogram when the M06 command is issued.

Example:



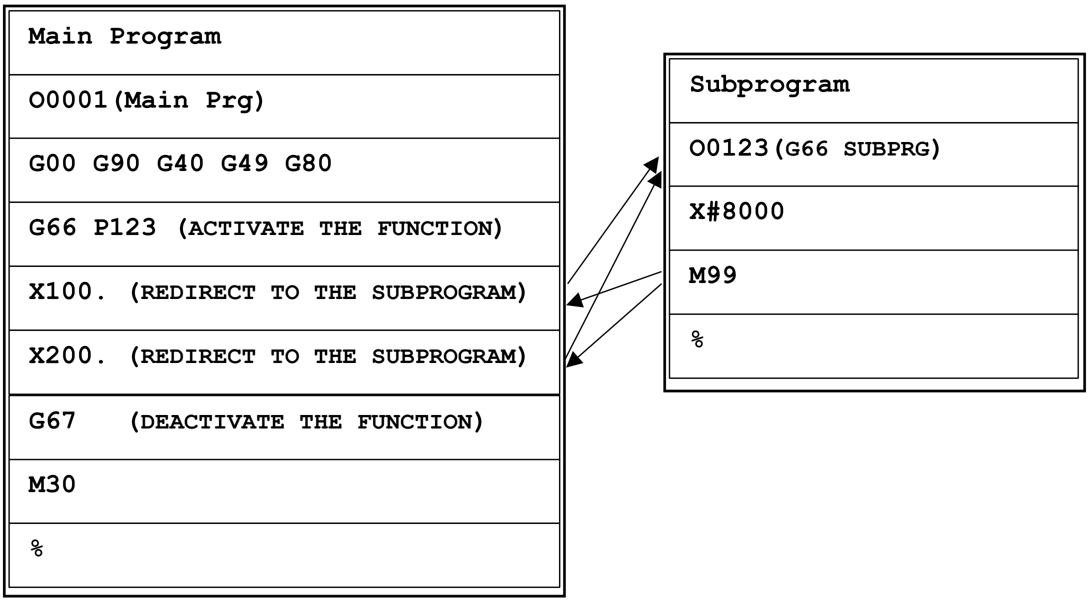
6.1.5. Calling a Repeating Subprogram with G66/G67

The G66 code allows for subprogram calls without requiring an additional command for each line. When G66 is used, the repeated subprogram call is activated. The P code written in the same line specifies the subprogram number to be called. All the commands between the G66 and G67 lines are skipped, and the system directly calls the subprogram specified by the P code.

Format:	**G66 P_**

P: Subprogram number (0~9999)

Example:



6.1.6. O9009: Resume Subprogram

When automatic operation is paused by the user, stopped due to an issue, or interrupted by a power failure, the system saves the processed line number and the current positions of the axes. The system can resume from the saved point. To enable this feature, PRM400 parameter must be activated (set to 1). Additionally, the spindle start and stop M codes must be entered into PRM403-PRM405 parameters. When the system detects a request to restart from the midpoint, it applies the G codes from the beginning of the selected program up to the line where the operation should resume. Then, the positions of the axes, along with the selected tool and spindle status, are loaded into variables #8050~#8067, and the O9009.cnc subprogram is called. The O9009 subprogram is dedicated to this operation and cannot be altered. All necessary preparations for starting from the midpoint must be made within the O9009 subprogram.

Address	Variable	Description	Format
16030~16031	#8015	Desired operation 1: Starting from the saved point 2: Stop->Run transition 3: Run->Stop transition 4: Sim->Run transition 5: Stop->Sim transition	0
16100~16101	#8050	X axis position for resuming operation	0.0000
16102~16103	#8051	Y axis position for resuming operation	0.0000
16104~16105	#8052	Z axis position for resuming operation	0.0000
16106~16107	#8053	4th axis position for resuming operation	0.0000
16108~16109	#8054	5th axis position for resuming operation	0.0000
16110~16111	#8055	6th axis position for resuming operation	0.0000
16112~16113	#8056	7th axis position for resuming operation	0.0000
16114~16115	#8057	8th axis position for resuming operation	0.0000
16120~16121	#8060	Spindle status for resuming operation (0: STOP/1: CW/2: CCW)	0
16122~16123	#8061	Spindle RPM for resuming operation	0
16124~16125	#8062	Tool number for resuming operation	0
16126~16127	#8063	Spindle 2 status for resuming operation (0: STOP/1: CW/2: CCW)	0
16128~16129	#8064	Spindle 2 RPM for resuming operation	0
16130~16131	#8065	Laser/plasma start command for resuming operation (0: None / 1: Present)	0
16132~16133	#8066	Laser/plasma status for resuming operation 0: Off 1: Piercing 2: Lead-In 3: Cutting 4: Lead-Out	0
16134~16135	#8067	Plasma AHC status for resuming operation 0: Off / 1: On	0

Sample resuming start subprogram for CNC router (O9009.cnc)

```
O9009(RESUME-START-STOP)
(--- RESUME ----------------------------)
N100 IF #8015 <> 1 THEN GOTO 200
N120 G53 G90 G00 X#2000 Y#2001 Z0.
N125 G43 H#100     (ACTIVATE TOOL LENGTH OFFSET)
N130 IF #8062 <= 0 THEN GOTO 800
N135 M6 T#8062              (GET TOOL)
N140 IF #8060 <> 1 THEN GOTO 150
N145 M03 S#8061              (SPINDLE CW START)
GOTO 170
N150 IF #8060 <> 2 THEN GOTO 160
M04 S#8061                   (SPINDLE CCW START)
GOTO 170
N160 IF #8060 <> 0 THEN GOTO 810
M05                          (SPINDLE STOP)
N170 G53 G90 G00 X#8050 Y#8051
N180 G53 G01 Z#8052
N190 M00
N199 GOTO 900
(--- STOP-> RUN TRANSITION-----------------)
N200 IF #8015 <> 2 THEN GOTO 300
G40 G69
N220 G53 G90 G00 X#2000 Y#2001 Z0.
N225 G43 H#100    (ACTIVATE TOOL LENGTH OFFSET)
N230 IF #8062 <= 0 THEN GOTO 800
M6 T#8062                    (GET TOOL)
N240 IF #8060 <> 1 THEN GOTO 250
M03 S#8061                   (SPINDLE CW START)
GOTO 270
N250 IF #8060 <> 2 THEN GOTO 260
M04 S#8061                   (SPINDLE CCW START)
GOTO 270
N260 IF #8060 <> 0 THEN GOTO 810
M05                          (SPINDLE STOP)
N270 G53 G90 G00 X#8050 Y#8051
N271 G53 G90 G01 Z#8052
N280 (M00)
N299 GOTO 900
(--- RUN-> STOP TRANSITION----------------)
N300 IF #8015 <> 3 THEN GOTO 400
N310 M05                    (SPINDLE STOP)
G40 G69
N399 GOTO 900
(--- SIM-> RUN TRANSITION -----------------)
N400 IF #8015 <> 4 THEN GOTO 500
N420 G53 G90 G00 Z0.
N425 G43 H#100 (ACTIVATE TOOL LENGTH OFFSET)
N430 IF #8062 <= 0 THEN GOTO 800
M6 T#8062                   (GET TOOL)
N440 IF #8060 <> 1 THEN GOTO 450
M03 S#8061                  (SPINDLE CW START)
GOTO 470
N450 IF #8060 <> 2 THEN GOTO 460
M04 S#8061               (SPINDLE CCW START)
GOTO 470
```

```
N460 IF #8060 <> 0 THEN GOTO 810
M05                         (SPINDLE STOP)
N470 G53 G90 G00 X#8050 Y#8051
N480 G53 G01 Z#8052
N499 GOTO 900
(--- STOP-> SIM TRANSITION -----------------)
N500 IF #8015 <> 5 THEN GOTO 900
N510 G53 G90 G00 Z0.
N520 G53 G90 G00 X#8050 Y#8051
N530 (M00)
N550 (M03)                (SPINDLE START)
N599 GOTO 900
(--- ALARMS---------------------------------)
N800 ALM 10 (WRONG TOOL COMMAND ALARM)
N805 GOTO 900
N810 ALM 11 (SPINDLE ROTATION COMMAND ALARM)
N815 GOTO 900
N900 G90 G43 H#100
N999 M99 (RETURN)
%
```

6.2. Commands Redirected to the Subprogram

When the subprogram call command is detected, the commands within this code block are not processed by the interpreter and are loaded into variables #8000~#8049 and #8100~#8149. This way, these commands can be controlled within the subprogram and used for different purposes.

In the code block below, when the system detects that a subprogram is called, it will not send a movement command to the X axis. Instead, it loads the value 1000000 into variable #8000 and the value 1 into variable #8100. As indicated in the table below, the formats of the commands are divided into integer and decimal numbers. Since axis commands are decimal, the command 100.0000 will be interpreted without the decimal point as (1000000) and loaded into variable #8000.

Main Program

```
O0001(MAIN PROG)
M98 P1000 G00 X100 (CALL THE SUBPROGRAM O1000.cnc.)
M30                (END OF THE PROGRAM)
%
```

Subprogram (O1000.cnc)

```
O1000(SUBPROG)
IF #8100 == 0 THEN GOTO 99 (IF THE X COMMAND IS NOT
GIVEN, RETURN)
X#8000             (MOVE THE X AXIS)
N99 M99            (RETURN TO THE MAIN PROGRAM)
%
```

Address	Variable	Description	Format
16000~16001	#8000	X Axis value in the line directed to subprogram	0.0000
16002~16003	#8001	Y Axis value in the line directed to subprogram	0.0000
16004~16005	#8002	Z Axis value in the line directed to subprogram	0.0000
16006~16007	#8003	4th Axis value in the line directed to subprogram	0.0000
16008~16009	#8004	5th Axis value in the line directed to subprogram	0.0000
16010~16011	#8005	6th Axis value in the line directed to subprogram	0.0000
16012~16013	#8006	7th Axis value in the line directed to subprogram	0.0000
16014~16015	#8007	8th Axis value in the line directed to subprogram	0.0000
16018~16019	#8009	D value in the line directed to subprogram	0
16020~16021	#8010	F value in the line directed to subprogram	0.0000
16022~16023	#8011	H value in the line directed to subprogram	0
16024~16025	#8012	I value in the line directed to subprogram	0.0000
16026~16027	#8013	J value in the line directed to subprogram	0.0000
16028~16029	#8014	K value in the line directed to subprogram	0.0000
16030~16031	#8015	L value in the line directed to subprogram	0
16032~16033	#8016	M value in the line directed to subprogram	0
16036~16037	#8018	P value in the line directed to subprogram	0
16038~16039	#8019	Q value in the line directed to subprogram	0.0000
16040~16041	#8020	R value in the line directed to subprogram	0.0000
16042~16043	#8021	S value in the line directed to subprogram	0
16044~16045	#8022	T value in the line directed to subprogram	0
16044~16045	#8023	B value in the line directed to subprogram	0
16060~16061	#8030	Group 0 G value in the line directed to subprogram	0.0
16062~16063	#8031	Group 1 G value in the line directed to subprogram	0.0
16064~16065	#8032	Group 2 G value in the line directed to subprogram	0.0
16066~16067	#8033	Group 3 G value in the line directed to subprogram	0.0
16068~16069	#8034	Group 4 G value in the line directed to subprogram	0.0
16070~16071	#8035	Group 5 G value in the line directed to subprogram	0.0
16072~16073	#8036	Group 6 G value in the line directed to subprogram	0.0
16074~16075	#8037	Group 7 G value in the line directed to subprogram	0.0
16076~16077	#8038	Group 8 G value in the line directed to subprogram	0.0
16078~16079	#8039	Group 9 G value in the line directed to subprogram	0.0
16080~16081	#8040	Group 10 G value in the line directed to subprogram	0.0
16082~16083	#8041	Group 11 G value in the line directed to subprogram	0.0
16084~16085	#8042	Group 12 G value in the line directed to subprogram	0.0
16086~16087	#8043	Group 13 G value in the line directed to subprogram	0.0
16088~16089	#8044	Group 14 G value in the line directed to subprogram	0.0
16090~16091	#8045	Group 15 G value in the line directed to subprogram	0.0
16092~16093	#8046	Group 16 G value in the line directed to subprogram	0.0
16094~16095	#8047	Group 17 G value in the line directed to subprogram	0.0
16096~16097	#8048	Group 18 G value in the line directed to subprogram	0.0
16098~16099	#8049	Group 19 G value in the line directed to subprogram	0.0

Addres	Variable	Description	Format
16200~16201	#8100	X Axis bit in the line directed to subprogram	0
16202~16203	#8101	Y Axis bit in the line directed to subprogram	0
16204~16205	#8102	Z Axis bit in the line directed to subprogram	0
16206~16207	#8103	4th Axis bit in the line directed to subprogram	0
16208~16209	#8104	5th Axis bit in the line directed to subprogram	0
16210~16211	#8105	6th Axis bit in the line directed to subprogram	0
16212~16213	#8106	7th Axis bit in the line directed to subprogram	0
16214~16215	#8107	8th Axis bit in the line directed to subprogram	0
16218~16219	#8109	D bit in the line directed to subprogram	0
16220~16221	#8110	F bit in the line directed to subprogram	0
16222~16223	#8111	H bit in the line directed to subprogram	0
16224~16225	#8112	I bit in the line directed to subprogram	0
16226~16227	#8113	J bit in the line directed to subprogram	0
16228~16229	#8114	K bit in the line directed to subprogram	0
16230~16231	#8115	L bit in the line directed to subprogram	0
16232~16233	#8116	M bit in the line directed to subprogram	0
16236~16237	#8118	P bit in the line directed to subprogram	0
16238~16239	#8119	Q bit in the line directed to subprogram	0
16240~16241	#8120	R bit in the line directed to subprogram	0
16242~16243	#8121	S bit in the line directed to subprogram	0
16244~16245	#8122	T bit in the line directed to subprogram	0
16244~16245	#8123	B bit in the line directed to subprogram	0
16260~16261	#8130	Group 0 G bit in the line directed to subprogram	0
16262~16263	#8131	Group 1 G bit in the line directed to subprogram	0
16264~16265	#8132	Group 2 G bit in the line directed to subprogram	0
16266~16267	#8133	Group 3 G bit in the line directed to subprogram	0
16268~16269	#8134	Group 4 G bit in the line directed to subprogram	0
16270~16271	#8135	Group 5 G bit in the line directed to subprogram	0
16272~16273	#8136	Group 6 G bit in the line directed to subprogram	0
16274~16275	#8137	Group 7 G bit in the line directed to subprogram	0
16276~16277	#8138	Group 8 G bit in the line directed to subprogram	0
16278~16279	#8139	Group 9 G bit in the line directed to subprogram	0
16280~16281	#8140	Group 10 G bit in the line directed to subprogram	0
16282~16283	#8141	Group 11 G bit in the line directed to subprogram	0
16284~16285	#8142	Group 12 G bit in the line directed to subprogram	0
16286~16287	#8143	Group 13 G bit in the line directed to subprogram	0
16288~16289	#8144	Group 14 G bit in the line directed to subprogram	0
16290~16291	#8145	Group 15 G bit in the line directed to subprogram	0
16292~16293	#8146	Group 16 G bit in the line directed to subprogram	0
16294~16295	#8147	Group 17 G bit in the line directed to subprogram	0
16296~16297	#8148	Group 18 G bit in the line directed to subprogram	0
16298~16299	#8149	Group 19 G bit in the line directed to subprogram	0

6.3. Macro Programming

Macro commands allow mathematical operations and conditional branching within the G-code file. Macro commands can be used with G65, and they can also be programmed using BASIC-like syntax. In the examples, usage in both forms is demonstrated as much as possible.

G Code	L Code	Operation	Definition
G65	L01	Assignment	#A = #B
G65	L02	Addition operation	#A = #B + #C
G65	L03	Subtraction operation	#A = #B - #C
G65	L04	Multiplication operation	#A = #B * #C
G65	L05	Division operation	#A = #B / #C
G65	L06	Block assignment	#A = BMOV[5, 3]
G65	L11	Logical OR operation	#A = #B | #C
G65	L12	Logical AND operation	#A = #B & #C
G65	L13	Logical XOR (Exclusive OR) operation	#A = #B ^ #C
G65	L14	Right shift	#A = SHR[#B, 3]
G65	L15	Left shift	#A = SHL[#B, 3]
G65	L21	Square root operation	#A = SQR[16]
G65	L22	Absolute value operation	#A = ABS[-16]
G65	L23	Modulus operation	#A = 18 % 4
G65	L24	Conversion from BCD code to Binary code	#A = BIN[18]
G65	L25	Conversion from Binary code to BCD code	#A = BCD[18]
G65	L27	Right triangle hypotenuse calculation	#A = SQRA[10, 15]
G65	L28	Right triangle side calculation	#A = SQRS[10, 15]
G65	L31	Sine calculation	#A = SIN[#B, 450000]
G65	L32	Cosine calculation	#A = COS[#B, 450000]
G65	L33	Tangent calculation	#A = TAN[#B, 450000]
G65	L34	Arctangent calculation	#A = ATAN[#C, #B]
G65	L35	Arcsine calculation	#A = ASIN[#B, #C]
G65	L36	Arccosine calculation	#A = ACOS[#B, #C]
G65	L80	Unconditional branching to a specific line	GOTO 300
G65	L81	Conditional branching to a specific line	IF #A == 100 THEN GOTO 300
G65	L82	Conditional branching to a specific line	IF #A <> 100 THEN GOTO 300
G65	L83	Conditional branching to a specific line	IF #A > 100 THEN GOTO 300
G65	L84	Conditional branching to a specific line	IF #A < 100 THEN GOTO 300
G65	L85	Conditional branching to a specific line	IF #A >= 100 THEN GOTO 300
G65	L86	Conditional branching to a specific line	IF #A <= 100 THEN GOTO 300
G65	L87	Unconditional branching to a specific line with cursor	INDX 10
G65	L99	Trigger an alarm	ALM 1

6.3.1. Assignment (a = b)

Assigns a fixed value or the value of another variable to a variable.

Example:

```
(ASSIGNS THE VALUE 100 TO VARIABLE #0)
G65 L1 P#0 Q100
(COPIES THE VALUE OF VARIABLE #1 INTO VARIABLE #0)
G65 L1 P#0 Q#1
```

Example:

```
(ASSIGNS THE VALUE 100 TO VARIABLE #0)
#0 = 100
(COPIES THE VALUE OF VARIABLE #1 INTO VARIABLE #0)
#0 = #1
```

6.3.2. Addition (a = b + c)

Adds two values and stores the result in the specified variable.

Example:

```
(ADDS 1 TO THE VALUE OF VARIABLE #1)
(AND STORES THE RESULT IN #0)
G65 L2 P#0 Q#1 R1
(ADDS THE VALUE OF VARIABLE #2 TO)
(THE VALUE OF VARIABLE #1 AND STORES THE RESULT IN #0)
G65 L2 P#0 Q#1 R#2
```

Example:

```
(ADDS 1 TO THE VALUE OF VARIABLE #1)
(AND STORES THE RESULT IN #0)
#0 = #1 + 1
(ADDS THE VALUE OF VARIABLE #2)
(TO THE VALUE OF VARIABLE #1 AND STORES THE RESULT IN #0)
#0 = #1 + #2
```

6.3.3. Subtraction (a = b – c)

It subtracts two values and stores the result in the specified variable.

Example:

```
(IT SUBTRACTS 1 FROM THE VALUE OF VARIABLE #1)
(AND STORES THE RESULT IN #0)
G65 L3 P#0 Q#1 R1
(IT SUBTRACTS THE VALUE OF VARIABLE #2 FROM)
(THE VALUE OF VARIABLE #1 AND STORES)
(THE RESULT IN #0)
G65 L3 P#0 Q#1 R#2
```

Example:

```
(IT SUBTRACTS 1 FROM THE VALUE OF VARIABLE #1)
(AND STORES THE RESULT IN #0)
#0 = #1 - 1
(IT SUBTRACTS THE VALUE OF VARIABLE #2)
(FROM THE VALUE OF VARIABLE #1 AND STORES THE RESULT IN #0)
#0 = #1 - #2
```

6.3.4. Multiplication (a = b * c)

It takes the product of two values and loads it into the specified variable.

Example:

```
(IT MULTIPLIES THE VALUE OF VARIABLE #1)
(BY 5 AND LOADS IT INTO #0)
G65 L4 P#0 Q#1 R5
(IT MULTIPLIES THE VALUE OF VARIABLE #1)
(BY THE VALUE OF VARIABLE #2 AND LOADS THE RESULT INTO #0)
G65 L4 P#0 Q#1 R#2
```

Example:

```
(IT MULTIPLIES THE VALUE OF VARIABLE #1)
(BY 5 AND LOADS IT INTO #0)
#0 = #1 * 5
(IT MULTIPLIES THE VALUE OF VARIABLE #1)
(BY THE VALUE OF VARIABLE #2 AND LOADS THE RESULT INTO #0)
#0 = #1 * #2
```

6.3.5. Division (a = b / c)

Divides the value of b by the value of c and loads the result into the specified variable.

Example:

```
(DIVIDES THE VALUE OF VARIABLE #1)
(BY 5 AND LOADS THE RESULT INTO #0)
G65 L5 P#0 Q#1 R5
(DIVIDES THE VALUE OF VARIABLE #1)
(BY THE VALUE OF VARIABLE #2 AND LOADS THE RESULT INTO #0)
G65 L5 P#0 Q#1 R#2
```

Example:

```
(DIVIDES THE VALUE OF VARIABLE #1)
(BY 5 AND LOADS THE RESULT INTO #0)
#0 = #1 / 5
(DIVIDES THE VALUE OF VARIABLE #1)
(BY THE VALUE OF VARIABLE #2 AND LOADS THE RESULT INTO #0)
#0 = #1 / #2
```

When the divisor value (c) is set to 0, **<ALM79> a division by zero error occurs in Macro Commands**.

6.3.6. Block Assignment (a[0] …a[c-1] = b)

Starting from the variable A, it loads the value of b for the number of times specified by c.

Example:

```
(STARTING FROM VARIABLE #0, IT LOADS THE VALUE 5 UP TO #2)
G65 L6 P#0 Q5 R3
(STARTING FROM VARIABLE #0, IT LOADS THE VALUE #5 UP TO #2)
G65 L6 P#0 Q#5 R3
```

Example:

```
(STARTING FROM VARIABLE #0, IT LOADS THE VALUE 5 UP TO #2)
#0 = BMOV[5, 3]
(STARTING FROM VARIABLE #0, IT LOADS THE VALUE #5 UP TO #2)
#0 = BMOV[#5, 3]
```

6.3.7. Logical OR (a = b | c)

b and c values are processed with a logical OR operation and the result is stored in variable a.

Example:

```
(THE VALUE OF VARIABLE #1 IS PROCESSED)
(WITH A LOGICAL OR OPERATION WITH #2)
(AND STORES THE RESULT IN VARIABLE #0)
G65 L11 P#0 Q#1 R#2
```

Example:

```
(THE VALUE OF VARIABLE #1 IS PROCESSED)
(WITH A LOGICAL OR OPERATION WITH #2)
(AND STORES THE RESULT IN VARIABLE #0)
#0 = #1 | #2
```

6.3.8. Logical AND (a = b & c)

It performs a logical AND operation between the values of b and c and loads the result into the a variable.

Example:

```
(IT PERFORMS A LOGICAL AND OPERATION)
(BETWEEN THE VALUE OF VARIABLE #1 AND #2)
(AND STORES THE RESULT IN VARIABLE #0)
G65 L12 P#0 Q#1 R#2
```

Example:

```
(IT PERFORMS A LOGICAL AND OPERATION)
(BETWEEN THE VALUE OF VARIABLE #1 AND #2)
(AND STORES THE RESULT IN VARIABLE #0)
#0 = #1 & #2
```

6.3.9. Logical Exclusive OR (a = b ^ c)

Performs a logical exclusive OR operation between the values of b and c, and stores the result in the variable a.

Example:

```
(IT PERFORMS A LOGICAL EXCLUSIVE OR)
(OPERATION BETWEEN THE VALUE OF VARIABLE #1 AND #2)
(AND LOADS THE RESULT INTO VARIABLE #0)
G65 L13 P#0 Q#1 R#2
```

Example:

```
(IT PERFORMS A LOGICAL EXCLUSIVE OR)
(OPERATION BETWEEN THE VALUE OF VARIABLE #1 AND #2)
(AND LOADS THE RESULT INTO VARIABLE #0)
#0 = #1 ^ #2
```

6.3.10. Right Bit Shift (a = b >> c)

Performs a right bit shift on the value of b by c positions and loads the result into the variable a.

Example:

```
(SHIFTS THE VALUE OF VARIABLE #1 BY 5 BITS)
(TO THE RIGHT AND LOADS THE RESULT INTO VARIABLE #0)
G65 L14 P#0 Q#1 R5
(SHIFTS THE VALUE OF VARIABLE #1 BY #2 BITS TO THE RIGHT)
(AND LOADS THE RESULT INTO VARIABLE #0)
G65 L14 P#0 Q#1 R#2
```

Example:

```
(SHIFTS THE VALUE OF VARIABLE #1 BY 5 BITS)
(TO THE RIGHT AND LOADS THE RESULT INTO VARIABLE #0)
#0 = SHR[#1, 5]
(SHIFTS THE VALUE OF VARIABLE #1 BY #2 BITS TO THE RIGHT)
(AND LOADS THE RESULT INTO VARIABLE #0)
#0 = SHR[#1, #2]
```

6.3.11. Left Bit Shift (a = b << c)

It shifts the value of b to the left by the amount specified by c and stores the result in variable a.

Example:

```
(IT SHIFTS THE VALUE OF VARIABLE #1 TO)
(THE LEFT BY 5 BITS AND STORES THE RESULT IN VARIABLE #0)
G65 L15 P#0 Q#1 R5
(IT SHIFTS THE VALUE OF VARIABLE #1 TO)
(THE LEFT BY THE NUMBER OF BITS SPECIFIED İN VARIABLE #2)
(AND IT LOADS THE RESULT INTO VARIABLE #0)
G65 L15 P#0 Q#1 R#2
```

Example:

```
(IT SHIFTS THE VALUE OF VARIABLE #1 TO THE LEFT)
(BY 5 BITS AND STORES THE RESULT IN VARIABLE #0)
#0 = SHL[#1, 5]
(IT SHIFTS THE VALUE OF VARIABLE #1 TO THE LEFT)
(BY THE NUMBER OF BITS SPECIFIED İN VARIABLE #2)
(AND IT LOADS THE RESULT INTO VARIABLE #0)
#0 = SHL[#1, #2]
```

6.3.12. Square Root (a = sqrt[b])

Takes the square root of the value of b and stores it in variable a.

Example:

```
(TAKES THE SQUARE ROOT OF 4 AND STORES IT IN #0)
G65 L21 P#0 Q4
(TAKES THE SQUARE ROOT OF VARIABLE #1 AND STORES IT IN #0)
G65 L21 P#0 Q#1
```

Example:

```
(TAKES THE SQUARE ROOT OF 4 AND STORES IT IN #0)
#0 = SQR[4]
(TAKES THE SQUARE ROOT OF VARIABLE #1 AND STORES IT IN #0)
#0 = SQR[#1]
```

6.3.13. Taking the Absolute Value (a = abs[b])

Takes the absolute value of b and assigns it to the variable a.

Example:

```
(THE ABSOLUTE VALUE OF -5 IS TAKEN AND LOADED INTO #0)
G65 L22 P#0 Q-5
(THE ABSOLUTE VALUE OF VARIABLE #1 IS TAKEN)
(AND LOADED INTO #0)
G65 L22 P#0 Q#1
```

Example:

```
(THE ABSOLUTE VALUE OF -5 IS TAKEN AND LOADED INTO #0)
#0 = ABS[-5]
(THE ABSOLUTE VALUE OF VARIABLE #1 IS TAKEN)
(AND LOADED INTO #0)
#0 = ABS[#1]
```

6.3.14. Getting the Remainder from the Division Operation (a = b % c)

It divides the value of b by the value of c and loads the remainder into the variable a.

Example:

```
(IT DIVIDES 5 BY 2 AND LOADS THE REMAINDER INTO VARIABLE #0)
G65 L23 P#0 Q5 R2
(IT DIVIDES VARIABLE #1 BY #2 AND LOADS)
(THE REMAINDER INTO #0)
G65 L23 P#0 Q#1 R#2
```

Example:

```
(IT DIVIDES 5 BY 2 AND LOADS THE REMAINDER INTO VARIABLE #0)
#0 = 5 % 2
(IT DIVIDES VARIABLE #1 BY #2 AND LOADS)
(THE REMAINDER INTO #0))
#0 = #1 % #2
```

When the divisor value (c) is set to 0, **<ALM79> a division by zero error occurs in Macro Commands**.

6.3.15. Converting BCD Value to Binary Number System (a = bin[b])

It converts the value of b, coded in BCD, to the binary number system and loads it into variable a.

Example:

```
(#1 = 0000 0000 0000 0000 0001 0010 0011 0100)
(        0    0    0    0    1    2    3    4)
(IT CONVERTS VARIABLE #1 TO THE BINARY NUMBER SYSTEM)
(AND LOADS IT INTO #0)
G65 L24 P#0 Q#1
(IT CONVERTS VARIABLE #1 TO THE BINARY NUMBER SYSTEM)
(AND LOADS IT INTO #0)
#0 = BIN[#1]
(#0 WILL BE 1234)
```

6.3.16. Encoding a Value as BCD (a = bcd[b])

It converts the value of b to BCD code and loads it into variable a.

Example:

```
(#1 = 1234)
(IT CONVERTS VARIABLE #1 TO THE BCD NUMBER SYSTEM)
(AND LOADS IT INTO #0)
G65 L25 P#0 Q#1
(IT CONVERTS VARIABLE #1 TO THE BCD NUMBER SYSTEM)
(AND LOADS IT INTO #0)
#0 = BCD[#1]
(#0 WILL BE 0000 0000 0000 0000 0001 0010 0011 0100)
```

6.3.17. Calculating the Hypotenuse of a Right Triangle (a = sqrt[b * b + c * c])

It adds the square of b and the square of c, then loads the square root of the resulting value into variable a.

Example:

```
(IT ADDS THE SQUARE OF 3 AND THE SQUARE OF 4)
(THEN TAKES THE SQUARE ROOT OF THE RESULTING VALUE)
(AND LOADS IT INTO VARİABLE #0)
G65 L27 P#0 Q3 R4 (#0 WILL BE 5)
(IT ADDS THE SQUARE OF #1 AND THE SQUARE OF #2)
(THEN TAKES THE SQUARE ROOT OF THE RESULTING VALUE)
(VALUE AND LOADS IT INTO VARIABLE #0)
G65 L27 P#0 Q#1 R#2
```

Example:

```
(IT ADDS THE SQUARE OF 3 AND THE SQUARE OF 4)
(THEN TAKES THE SQUARE ROOT OF THE RESULTING VALUE)
(AND LOADS IT INTO VARIABLE #0)
#0 = SQRA[3, 4] (#0 WILL BE 5)
(IT ADDS THE SQUARE OF #1 AND THE SQUARE OF #2)
(THEN TAKES THE SQUARE ROOT OF THE RESULTING VALUE)
(VALUE AND LOADS IT INTO VARIABLE #0)
#0 = SQRA[#1, #2]
```

6.3.18. Calculating the Side of a Right Triangle (a = sqrt[b * b – c * c])

It subtracts the square of c from the square of b, then takes the square root of the resulting value and loads it into variable a.

Example:

```
(IT SUBTRACTS THE SQUARE OF 3 FROM THE SQUARE OF 5)
(THEN TAKES THE SQUARE ROOT OF THE RESULTING VALUE)
(AND LOADS IT INTO VARIABLE #0)
G65 L28 P#0 Q5 R3 (#0 WILL BE 4)
(IT SUBTRACTS THE SQUARE OF #2 FROM THE SQUARE OF #1)
(THEN TAKES THE SQUARE ROOT OF THE RESULTING VALUE)
(AND LOADS IT INTO VARIABLE #0)
G65 L28 P#0 Q#1 R#2
```

Example:

```
(IT SUBTRACTS THE SQUARE OF 3 FROM THE SQUARE OF 5)
(THEN TAKES THE SQUARE ROOT OF THE RESULTING VALUE)
(AND LOADS IT INTO VARIABLE #0)
#0 = SQRS[5, 3] (#0 WILL BE 4)
(IT SUBTRACTS THE SQUARE OF #2 FROM THE SQUARE OF #1)
(THEN TAKES THE SQUARE ROOT OF THE RESULTING VALUE)
(AND LOADS IT INTO VARIABLE #0)
#0 = SQRS[#1, #2]
```

6.3.19. Sine (a = b * sin[c])

It calculates the sine of the angle given as c in degrees, multiplies the result by b, and loads it into variable a. The value of c must be specified in 0.0000 format: 1 degree should be given as 10000, and 30 degrees as 300000.

Example:

```
(IT CALCULATES THE SINE OF 30 DEGREES, MULTIPLIES)
(THE RESULT BY 10000, AND LOADS IT INTO VARIABLE #0)
G65 L31 P#0 Q10000 R300000 (#0 WILL BE 5000)
(IT CALCULATES THE SİNE OF THE ANGLE SPECİFİED BY #2)
(MULTIPLIES THE RESULT BY #1, AND LOADS IT INTO VARIABLE #0)
G65 L31 P#0 Q#1 R#2
```

Example:

```
(IT CALCULATES THE SINE OF 30 DEGREES, MULTIPLIES)
(THE RESULT BY 10000, AND LOADS IT INTO VARIABLE #0)
#0 = SIN[10000, 300000] (#0 WILL BE 5000)
(IT CALCULATES THE SİNE OF THE ANGLE SPECİFİED BY #2)
(MULTIPLIES THE RESULT BY #1, AND LOADS IT INTO VARIABLE #0)
#0 = SIN[#1, #2]
```

6.3.20. Cosine (a = b * cos[c])

It calculates the cosine of the angle given as c in degrees, multiplies the result by b, and loads it into variable a. The value of c must be specified in 0.0000 format: 1 degree should be given as 10000, and 30 degrees as 300000.

Example:

```
(IT CALCULATES THE COSINE OF 60 DEGREES, MULTIPLIES)
(THE RESULT BY 10000, AND LOADS IT INTO VARIABLE #0)
G65 L32 P#0 Q10000 R600000 (#0 WILL BE 5000)
(IT CALCULATES THE COSINE OF THE ANGLE SPECIFIED BY #2)
(MULTIPLIES THE RESULT BY #1, AND LOADS IT INTO VARIABLE #0)
G65 L32 P#0 Q#1 R#2
```

Example:

```
(IT CALCULATES THE COSINE OF 60 DEGREES, MULTIPLIES)
(THE RESULT BY 10000, AND LOADS IT INTO VARIABLE #0)
#0 = COS[10000, 600000] (#0 WILL BE 5000)
(IT CALCULATES THE COSINE OF THE ANGLE SPECIFIED BY #2)
(MULTIPLIES THE RESULT BY #1, AND LOADS IT INTO VARIABLE #0)
#0 = COS[#1, #2]
```

6.3.21. Tangent (a = b * tan[c])

It calculates the tangent of the angle given as c in degrees, multiplies the result by b, and loads it into variable a. The value of c must be specified in 0.0000 format: 1 degree should be given as 10000, and 30 degrees as 300000.

Example:

```
(IT CALCULATES THE TANGENT OF 45 DEGREES, MULTIPLIES)
(THE RESULT BY 10000, AND LOADS IT INTO VARIABLE #0)
G65 L33 P#0 Q10000 R450000 (#0 WILL BE 10000)
(IT CALCULATES THE TANGENT OF THE ANGLE SPECIFIED BY #2)
(MULTIPLIES THE RESULT BY #1, AND LOADS IT INTO VARIABLE #0)
G65 L33 P#0 Q#1 R#2
```

Example:

```
(IT CALCULATES THE TANGENT OF 45 DEGREES, MULTIPLIES)
(THE RESULT BY 10000, AND LOADS IT INTO VARIABLE #0)
#0 = TAN[10000, 450000] (#0 WILL BE 10000)
(IT CALCULATES THE TANGENT OF THE ANGLE SPECIFIED BY #2)
(MULTIPLIES THE RESULT BY #1, AND LOADS IT INTO VARIABLE #0)
#0 = TAN[#1, #2]
```

6.3.22. Arctangent (a = atan[b / c])

It divides the value of b by the value of c, then takes the arctangent of the result and loads it into variable a.

Example:

```
(IT DIVIDES #1 BY #2, THEN CALCULATES THE ARCTANGENT)
(OF THE RESULT AND LOADS IT INTO VARIABLE #0)
G65 L34 P#0 Q#1 R#2
```

Example:

```
(IT DIVIDES #1 BY #2, THEN CALCULATES THE ARCTANGENT)
(OF THE RESULT AND LOADS IT INTO VARIABLE #0)
#0 = ATAN[#1, #2]
```

When the divisor value (c) is set to 0, **<ALM79> a division by zero error occurs in Macro Commands**.

6.3.23. Arcsine (a = b * asin[c])

It multiplies the value of b by the arcsine of the value of c and loads the result into variable a. The values of b and c must be specified in 0.0000 format.

Example:

```
(IT CALCULATES THE ARCSINE OF 0.5, MULTIPLIES)
(THE RESULT BY 10000, AND LOADS IT INTO VARIABLE #0)
G65 L35 P#0 Q10000 R5000 (#0 WILL BE 300000)
(IT CALCULATES THE ARCSINE OF THE VALUE SPECIFIED BY #2)
(MULTIPLIES #1 BY THIS VALUE, AND LOADS THE RESULT)
(INTO VARIABLE #0)
G65 L35 P#0 Q#1 R#2
```

Example:

```
(IT CALCULATES THE ARCSINE OF 0.5, MULTIPLIES)
(THE RESULT BY 10000, AND LOADS IT INTO VARIABLE #0)
#0 = ASIN[10000, 5000] (#0 WILL BE 300000)
(IT CALCULATES THE ARCSINE OF THE VALUE SPECIFIED BY #2)
(MULTIPLIES #1 BY THIS VALUE, AND LOADS THE RESULT)
(INTO VARIABLE #0)
#0 = ASIN[#1, #2]
```

6.3.24. Arccosine (a = b * acos[c])

It multiplies the value of b by the arccosine of the value of c and loads the result into variable a. The values of b and c must be specified in 0.0000 format.

Example:

```
(IT CALCULATES THE ARCCOSINE OF 0.5, MULTIPLIES)
(THE RESULT BY 10000, AND LOADS IT INTO VARIABLE #0)
G65 L36 P#0 Q10000 R5000 (#0 WILL BE 600000)
(IT CALCULATES THE ARCCOSİNE OF THE VALUE SPECIFIED BY #2)
(MULTIPLIES #1 BY THIS VALUE, AND LOADS THE RESULT)
(INTO VARIABLE #0)
G65 L36 P#0 Q#1 R#2
```

Example:

```
(IT CALCULATES THE ARCCOSINE OF 0.5, MULTIPLIES)
(THE RESULT BY 10000, AND LOADS IT INTO VARIABLE #0)
#0 = ACOS[10000, 5000] (#0 WILL BE 600000)
(IT CALCULATES THE ARCCOSİNE OF THE VALUE SPECIFIED BY #2)
(MULTIPLIES #1 BY THIS VALUE, AND LOADS THE RESULT)
(INTO VARIABLE #0)
#0 = ACOS[#1, #2]
```

6.3.25. Unconditional Branching to a Specific Line (goto a)

It searches for the line number (N) specified by a within the program, moves the cursor to that line, and continues execution from there.

Example:

O0001 (GOTO EXAMPLE)
G00 G54 G90 G40 G49 G80
G65 L80 P10 (BRANCH TO THE LINE MARKED WITH N10)
X100. (THIS LINE IS SKIPPED WITHOUT BEING PROCESSED)
N10 X0. (THE PROGRAM CONTINUES FROM HERE)
GOTO 20 (BRANCH TO THE LINE MARKED WITH N20)
X-100.(LINE IS BRANCHED OVER WITHOUT BEING PROCESSED)
N20 X50. (THE PROGRAM CONTINUES FROM HERE)
M30 (END OF THE PROGRAM)
%

If the target line N is not found, **<ALM99> Program End** error occurs.

6.3.26. Branch to the Specified Line Number if Two Values are Equal (if B == B then goto a)

If the value of b is equal to the value of c, it searches for the line number (N) specified by a within the program, moves the cursor to that line, and continues execution from there.

Example:

O0001 (BRANCH IF EQUAL EXAMPLE)
G00 G54 G90 G40 G49 G80
G65 L81 P10 Q1 R2(BRANCH TO THE LINE MARKED WITH N10)
X100. (SINCE 1 != 2, THIS LINE IS PROCESSED)
N10 X0. (THE PROGRAM CONTINUES FROM HERE)
IF 1 == 1 THEN GOTO 20 (SINCE 1 == 1)
X-100.(LINE IS BRANCHED OVER WITHOUT BEING PROCESSED)
N20 X50. (THE PROGRAM CONTINUES FROM HERE)
M30 (END OF THE PROGRAM)
%

If the target line N is not found, **<ALM99> Program End** error occurs.

In the given example, constant numbers are used for simplicity. The values of b and c can be expressed as either constants or variables.

6.3.27. Branch to the Specified Line if Two Values are Not Equal (if b <>b then goto a)

If the value of b is not equal to the value of c, it searches for the line number (N) specified by a within the program, moves the cursor to that line, and continues execution from there.

Example:

```
O0001 (BRANCH IF NOT EQUAL EXAMPLE)
G00 G54 G90 G40 G49 G80
G65 L82 P10 Q1 R1(BRANCH TO THE LINE MARKED WITH N10)
X100.           (SINCE 1 == 1, THIS LINE IS PROCESSED)
N10 X0.                (THE PROGRAM CONTINUES FROM HERE)
IF 1 <> 2 THEN GOTO 20                     (SINCE 1 != 2)
X-100.(LINE IS BRANCHED OVER WITHOUT BEING PROCESSED)
N20 X50.               (THE PROGRAM CONTINUES FROM HERE)
M30                                 (END OF THE PROGRAM)
%
```

If the target line N is not found, **<ALM88> Program Not Found** error occurs.

In the given example, constant numbers are used for simplicity. The values of b and c can be expressed as either constants or variables.

6.3.28. Branch to the Specified Line if b is Greater Than c (if b > c then goto a)

If the value of b is greater than the value of c, it searches for the line number (N) specified by a within the program, moves the cursor to that line, and continues execution from there.

Example:

```
O0001 (BRANCH IF GREATER THAN EXAMPLE)
G00 G54 G90 G40 G49 G80
G65 L83 P10 Q1 R2(BRANCH TO THE LINE MARKED WITH N10)
X100.          (SINCE 1 < 2, THIS LINE IS PROCESSED)
N10 X0.              (THE PROGRAM CONTINUES FROM HERE)
IF 2 > 1 THEN GOTO 20                    (SINCE 2 > 1)
X-100.(LINE IS BRANCHED OVER WITHOUT BEING PROCESSED)
N20 X50.             (THE PROGRAM CONTINUES FROM HERE)
M30                               (END OF THE PROGRAM)
%
```

If the target line N is not found, **<ALM88> Program Not Found** error occurs.

In the given example, constant numbers are used for simplicity. The values of b and c can be expressed as either constants or variables.

6.3.29. Branch to the Specified Line if b is Less Than c (if b < c then goto a)

If the value of b is less than the value of c, it searches for the line number (N) specified by a within the program, moves the cursor to that line, and continues execution from there.

Example:

```
O0001 (BRANCH IF LESS THAN EXAMPLE)
G00 G54 G90 G40 G49 G80
G65 L84 P10 Q2 R1(BRANCH TO THE LINE MARKED WITH N10)
X100.             (SINCE 2 > 1, THIS LINE IS PROCESSED)
N10 X0.              (THE PROGRAM CONTINUES FROM HERE)
IF 1 < 2 THEN GOTO 20                    (SINCE 1 < 2)
X-100.(LINE IS BRANCHED OVER WITHOUT BEING PROCESSED)
N20 X50.             (THE PROGRAM CONTINUES FROM HERE)
M30                              (END OF THE PROGRAM)
%
```

If the target line N is not found, the error **<ALM88> Program Not Found** occurs.

In the provided example, fixed numbers are used for simplicity. However, both b and c values can be expressed as constants or variables.

6.3.30. Branch to the Specified Line if b is Greater Than or Equal to c (if b >= c then goto a)

If b is greater than or equal to c, it searches for the specified line number (N) within the program and moves the cursor to this line to continue processing from there.

Example:

```
O0001 (GREATER THAN OR EQUAL EXAMPLE)
G00 G54 G90 G40 G49 G80
G65 L85 P10 Q1 R2(BRANCH TO THE LINE MARKED WITH N10)
X100.            (SINCE 1 < 2, THIS LINE IS PROCESSED)
N10 X0.                (THE PROGRAM CONTINUES FROM HERE)
IF 1 >= 1 THEN GOTO 20                   (SINCE 1 == 1)
X-100.(LINE IS BRANCHED OVER WITHOUT BEING PROCESSED)
N20 X50.               (THE PROGRAM CONTINUES FROM HERE)
M30                                 (END OF THE PROGRAM)
%
```

If the target line N is not found, the **<ALM88> Program Not Found** error occurs.

In the given example, fixed numbers have been used for simplicity. The values of b and c can be expressed as either fixed or variable.

6.3.31. Branch to the Specified Line if b is Less Than or Equal to c (if b <= c then goto a)

If b is less than or equal to c, it searches for the line number (N) specified by a within the program and moves the cursor to this line, continuing execution from there.

Example:

```
O0001 (LESS THAN OR EQUAL TO EXAMPLE)
G00 G54 G90 G40 G49 G80
G65 L86 P10 Q2 R1(BRANCH TO THE LINE MARKED WITH N10)
X100.             (SINCE 2 > 1, THIS LINE IS PROCESSED)
N10 X0.                 (THE PROGRAM CONTINUES FROM HERE)
IF 1 <= 2 THEN GOTO 20                       (SINCE 1 < 2)
X-100.(LINE IS BRANCHED OVER WITHOUT BEING PROCESSED)
N20 X50.                (THE PROGRAM CONTINUES FROM HERE)
M30                                   (END OF THE PROGRAM)
%
```

If the target line N cannot be found, the error **<ALM88> Program Not Found** occurs.

In the provided example, fixed numbers are used for simplicity. The values of b and c can be expressed as either fixed or variable.

6.3.32. Shift the Cursor by the Specified Number (indx a) (indx a)

The program cursor is shifted up or down by the number specified by 'a'. The line to be branched is specified relative to the line after this command. If a line number lower than the program's starting point is given, it will branch to the first line. If a value higher than the total number of lines is provided, the program will branch to the last line

Example:

O0001 (LESS THAN OR EQUAL TO EXAMPLE)	
G00 G54 G90 G40 G49 G80	
G65 L87 P1	(BRANCH 1 LINE DOWN)
X100.	(THİS LINE IS SKIPPED)
X10.	(THE PROGRAM CONTINUES FROM HERE)
GOTO 20	(BRANCH TO LINE N20)
N10 X20.	(LINE BRANCHED UPWARD)
GOTO 30	(END THE PROGRAM)
N20 INDX -3	(BRANCH 3 LINES UP)
N30 M30	(END OF THE PROGRAM)
%	

6.3.33. Alarm (alm a)

This command is used to trigger an alarm within the program. When the system enters the alarm state, execution halts at this line. A total of 16 different alarms can be created. The value of a must be between 0 and 15. If a value outside this range is specified, a macro alarm with number 15 is generated. The descriptions of these alarms can be customized by the machine manufacturer.

Example:

```
(CREATE MACRO ALARM 0)
G65 L99 P0
```

When this line is reached, **<ALM288> Macro Alarm 0** error is triggered.

Example:

```
(CREATE MACRO ALARM 0)
ALM 5
```

When this line is reached, **<ALM293> Macro Alarm 5** error is triggered.

6.4. Subprogram and Macro Programming Example

Subprogram for Fixed Magazine Tool Change and Tool Length Offset Setting (O9001.cnc)

```
O9001(TOOL CHANGE)
(---TOOL NUMBER CHECK -------------------------)
N049 IF #8122 == 0 THEN GOTO 800(NO TOOL,PROBING)
N050 #10 = #8022                 (REQUESTED TOOL)
N051 IF #10 == #100 THEN GOTO 800(T_NEW == T_OLD)
N052 IF #100 < 0 THEN GOTO 900        (T_OLD < 0)
N053 IF #100 > #109 THEN GOTO 900 (T_OLD > T_MAX)
N054 IF #10 < 0 THEN GOTO 900         (T_NEW < 0)
N055 IF #10 > #109 THEN GOTO 900  (T_NEW > T_MAX)
#101 = 0
(---PREPARATION -------------------------------)
N070 G90 G53 G00 Z0.                      (Z REF)
N071 M5                           (SPINDLE STOP)
N072 M50                           (CONVEYOR UP)
     M52                              (BRUSH UP)
     G04 P1000
     G49
N073 IF #100 == 0 THEN GOTO 200      (T_OLD = 0)
(---RELEASE TOOL-----------------------------)
N100 #11 = #100 - 1        (T1-MAX->T0-T_MAX-1)
N101 #12 = #11 * 5                 (ADR START)
N102 #13 = #12 + 4010             (ADR OFFSET)
N103 #15 = #10013                     (TOOL X)
N104 #13 = #13 + 1                     (ADR++)
N105 #16 = #10013                     (TOOL Y)
N106 #13 = #13 + 1                     (ADR++)
N107 #17 = #10013                     (TOOL Z)
N108 #13 = #13 + 1                     (ADR++)
N109 #18 = #10013                    (TOOL Y2)
N110 #13 = #13 + 1                     (ADR++)
N111 #19 = #10013                     (TOOL F)
N112 #13 = #13 + 1                     (ADR++)
N113 #20 = #15 + #107      (TOOL X + X RETRACT)
N114 #21 = #16 + #108      (TOOL Y + Y RETRACT)
(---TOOL RELEASE MOVEMENT--------------------)
N120 G90 G00 G53 X#20 Y#21    (TOOL XY RETRACT)
N121 G90 G00 G53 Z#18                 (TOOL Z)
N122 G90 G01 G53 Z#17 F#106          (TOOL Z2)
N122 G90 G01 G53 X#15 Y#16 F#19      (TOOL XY)
N123 G04 P500
N124 M22                        (TOOL UNCLAMP)
     M23                         (TURN ON AIR)
     #101 = 1
N125 #100 = 0          (EMPTY TOOL REGISTRATION)
     T0
     #95 = 0
N126 G04 P250
N127 IF #10 <> 0 THEN GOTO 200       (T_NEW = 0)
N128 G53 G90 Z0.                         (Z REF)
N129 M21                          (TOOL CLAMP)
     M24                        (TURN OFF AIR)
```

```
T0
GOTO 990
(---TOOL TAKE--------------------------------)
N200 #11 = #10 - 1          (T1_MAX->T0-T_MAX-1)
N201 #12 = #11 * 5                   (ADR_START)
N202 #13 = #12 + 4010               (ADR OFFSET)
N203 #15 = #10013                       (TOOL X)
N204 #13 = #13 + 1                       (ADR++)
N205 #16 = #10013                       (TOOL Y)
N206 #13 = #13 + 1                       (ADR++)
N207 #17 = #10013                       (TOOL Z)
N208 #13 = #13 + 1                       (ADR++)
N209 #18 = #10013                      (TOOL Y2)
N210 #13 = #13 + 1                       (ADR++)
N211 #19 = #10013                       (TOOL F)
N212 #13 = #13 + 1                       (ADR++)
N213 #20 = #15 + #107       (TOOL X + X RETRACT)
N214 #21 = #16 + #108       (TOOL Y + Y RETRACT)
(---TOOL TAKE MOVEMENT-----------------------)
N215 IF #101 == 1 THEN GOTO 220
     #101 = 0
N216 G90 G53 G00 Z0.                     (Z REF)
N217 G90 G00 G53 X#15 Y#16             (TOOL XY)
N220 G90 G00 G53 Z#18                   (TOOL Z)
     M22                          (TOOL UNCLAMP)
     #101 = 0
N221 G90 G00 G53 X#15 Y#16             (TOOL XY)
N222 G90 G01 G53 Z#17 F#106             (TOOL Z)
N223 M21                            (TOOL CLAMP)
     M24                          (TURN OFF AIR)
N224 G04 P250
N122 G90 G01 G53 X#20 Y#21 F#19(TOOL XY RETRACT)
N225 G91 G00 G28 Z0.                    (Z HOME)
N226 #100 = #10            (TOOL REGISTRATION)
     T#100
N227 GOTO 800
(---PROBING----------------------------------)
N800 IF #8102 == 0 THEN GOTO 990        (Z CMD?)
     IF #100 <= 0 THEN GOTO 901        (NO TOOL)
N801 G49 G91 G00 G28 Z0.                (Z HOME)
     M50                           (CONVEYOR UP)
     M52                              (BRUSH UP)
     G49
N802 G90 G00 G53 X#170 Y#171          (PROBE XY)
N803 G00 G00 G53 Z#172
N804 G90 G01 G59.4 G31 Z#173 F#174
N805 G91 G00 Z#176                   (Z RETRACT)
N806 G90 G01 G59.4 G31 Z#173 F#175
G54
G04 P250
N807 #22 = #2002 - #177
N808 G90 G10 L1 P#100 R#22
G43 H#100
N809 G91 G00 G28 Z0. (Z HOME)
N810 G53 G90 G00 Z0.0
N811 GOTO 990
(---EXIT-----------------------------------)
N900 ALM 0
```

```
N901 ALM 1
N990 G90 G54 G43 H#100
#101 = 0
#95 = #100
N999 M99 (RETURN)
%
```

7. Alarm List And Troubleshooting

Alarm No	Description	Suggestions
ALM<020>	Unknown system error	An unexpected situation has occurred in the system's core software. Please contact HSC Control.

Alarm No	Description	Suggestions
ALM<022>	Nvalid negative value	A G-code block cannot have negative values assigned to the commands D, F, H, L, N, O, S, or T. Replace the value assigned to these commands with a positive one. In a variable pitch threading cycle, if the pitch value within the motion is calculated as a negative value due to the given pitch change amount and length, modify either the pitch change amount or the motion length.

Alarm No	Description	Suggestions
ALM<023>	Invalid ”D” value	The D value specified alongside the G41/G42 commands exceeds the supported maximum tool offset. Enter a value between 0 and 100 for the D command.

Alarm No	Description	Suggestions
ALM<024>	Invalid ”G” value	The value written next to the G command must be between 0 and 99.9.

Alarm No	Description	Suggestions
ALM<025>	Invalid ”H” value	The H value specified next to the G43/G44 commands exceeds the supported maximum tool offset. Enter a value between 0 and 100 for the H command.

Alarm No	Description	Suggestions
ALM<027>	Invalid ”M” value	The value specified next to the M commands must be between 0 and 9999.

Alarm No	Description	Suggestions
ALM<028>	Invalid ”P” value	The P value specified next to the G10 command is out of range. Refer to the Pulser3 Programming Manual. In laser cutting software, the technology block can take a value between 0 and 9. The G30 command is missing the P command, or the specified P value is not between 2 and 4. Add a P command with a value between P2 and P4 to the G30 command.

Alarm No	Description	Suggestions
ALM<029>	Invalid ”T” value	In the lathe software, the specified T value is greater than the supported maximum number of tools. Enter a value between 1 and 50.

Alarm No	Description	Suggestions
ALM<030>	Invalid variable value	The variable number specified next to a command must be between 0 and 19999. The variable number to be read by the PC or HMI must be between 0 and 19999.

Alarm No	Description	Suggestions
ALM<032>	Invalid ”L” value	The L value specified next to the M98 command must be between 1 and 9999999. The L value specified next to the G10 command is out of range. Refer to the Pulser3 Programming Manual.

Alarm No	Description	Suggestions
ALM<033>	There is a syntax error in the G-code line	Each G-code line must contain a maximum of 63 characters. Reduce the number of characters in the problematic line. A value might be missing next to a command. Ensure that all commands are followed by adjacent values. A value might be written without a preceding command. Each opened parenthesis must be closed. Verify the correct order and number of parentheses. The value written next to a command should consist of digits and a dot. The number of digits should not exceed 10. Next to a variable sign (#), only the minus sign (-) and digits should be written on the left side.

Alarm No	Description	Suggestions
ALM<034>	There is a missing command in the G-code line	A rotary axis target command (A/B/C) has not been specified next to the G33.1 or G34.1 command. Add a rotary axis target command next to these commands. In the lathe software, the S command has either not been written or its value has been set to 0 next to the G96 command. Write an S command greater than 0 next to the G96 command.

Alarm No	Description	Suggestions
ALM<041>	The 'P' command is missing	Add a valid 'P' command next to the G98 command. Add a valid 'P' command next to the G10 command. Add a valid 'P' command next to the G65 command. In the laser cutting software, add a valid 'P' command next to the G72.1 command. If the G72, G74, G76, G82, G84, G87, G88, or G89 commands are being processed for the first time, add a valid 'P' command next to them.

Alarm No	Description	Suggestions
ALM<042>	The 'Q' command is missing	Add a valid 'Q' command next to the G65 command. In the laser cutting software, add a valid 'Q' command next to the G72.1 or G72.2 command. If the G72, G73, G76, G83, or G87 commands are being processed for the first time, add a valid 'Q' command next to them.

Alarm No	Description	Suggestions
ALM<043>	The 'R' command is missing	Add a valid 'R' command next to the G10 command. Add a valid 'R' command next to the G65 command . Add a valid 'R' command next to the G68 command. In the laser cutting software, add a valid 'R' command next to the G72.1 or G72.2 command. If the repetitive cycle commands are being processed for the first time, add a valid 'R' command next to them.

Alarm No	Description	Suggestions
ALM<044>	The 'X' command is missing	If the repetitive cycle commands are being processed for the first time in the YZ plane, add a valid 'X' command next to them.

Alarm No	Description	Suggestions
ALM<045>	The 'Y' command is missing	If the repetitive cycle commands are being processed for the first time in the ZX plane, add a valid 'Y' command next to them.

Alarm No	Description	Suggestions
ALM<046>	“The 'Z' command is missing	If the repetitive cycle commands are being processed for the first time in the XY plane, add a valid 'Z' command next to them.

Alarm No	Description	Suggestions
ALM<047>	'R' or 'IJK' commands are missing	An 'R' or 'IJK' command must be added next to a circular motion command (G02/G03).

Alarm No	Description	Suggestions
ALM<048>	'R' and 'IJK' commands are provided together	'R' and 'IJK' commands must not be written together next to a circular motion command (G02/G03)

Alarm No	Description	Suggestions
ALM<049>	Use the 'R' command in polar mode	When the polar coordinate system is activated, only the 'R' command can be written next to a circular motion command (G02/G03). 'IJK' commands cannot be used.

Alarm No	Description	Suggestions
ALM<051>	The 'F' command is missing	If the threading commands are being processed for the first time, add a valid 'F' command next to them.

Alarm No	Description	Suggestions
ALM<056>	The start and end coordinates of the arc are the same	When an 'R' command is added to a circular motion command (G02/G03), the start and end coordinates of the arc must not be the same.

Alarm No	Description	Suggestions
ALM<057>	The radius of the arc is too small	The given radius of the arc is too small to be processed. Enter a larger radius value.

Alarm No	Description	Suggestions
ALM<058>	The radius of the arc is zero	The given radius of the arc is 0. Enter a radius value greater than zero.

Alarm No	Description	Suggestions
ALM<059>	The start and end radius of the arc are different	The difference between the start and end radius of the given arc is outside the tolerance.

Alarm No	Description	Suggestions
ALM<061>	The 'R' value is smaller than the 'Z' value	In canned cycle commands, the value of the 'R' command must be greater than the target value of the hole axis (G17: Z / G18: Y / G19: X).

Alarm No	Description	Suggestions
ALM<065>	Turn off tool radius compensation to change the plane	When the tool radius compensation command is activated, the plane cannot be changed. Turn off radius compensation with the G40 command before the plane commands (G17/G18/G19).

Alarm No	Description	Suggestions
ALM<072>	Turn off tool radius compensation	Turn off tool radius compensation before the G94/G95 commands. Turn off tool radius compensation before the G15/G16 commands. "Turn off tool radius compensation before the canned cycle commands. Turn off tool radius compensation before the G28 and G30 commands. Turn off tool radius compensation before the G31 command. Turn off tool radius compensation before the G92 command.

Alarm No	Description	Suggestions
ALM<075>	Invalid macro command	There is no macro command corresponding to the L value specified next to the G65 command.

Alarm No	Description	Suggestions
ALM<076>	The macro command is missing	Add a valid 'L' command next to the G65 command.

Alarm No	Description	Suggestions
ALM<077>	G66 is already active	Nested G66 commands have been written. Terminate the active macro command with G67 before issuing a new G66 command.

Alarm No	Description	Suggestions
ALM<078>	The 'P' command must be specified with a variable	Specify the value of the 'P' command next to the G65 command with a variable.

Alarm No	Description	Suggestions
ALM<079>	Division by zero error in macro commands	A division by zero operation is requested in macro commands. Division by zero is not allowed. Check that the divisor is a non-zero value before the macro command.

Alarm No	Description	Suggestions
ALM<080>	The 'Q' value in the macro command is zero	In the ATN, SINH, COSH macro commands, the specified divisor is zero. Division by zero is not allowed. Check that the divisor is a non-zero value before the macro command.

Alarm No	Description	Suggestions
ALM<085>	A maximum of two nested subprogram calls are allowed	

Alarm No	Description	Suggestions
ALM<086>	Invalid subprogram number	The 'P' value specified next to the G66 command must be between 0 and 9999. The 'P' value specified next to the M98 command must be between 0 and 9999.

Alarm No	Description	Suggestions
ALM<087>	The program is already open	The subprogram to be called has already been selected as the main program or opened as a subprogram and has not been closed yet. To call the subprogram again, the M99 command must be used to return from the subprogram to the main program.

Alarm No	Description	Suggestions
ALM<088>	Program not found	The search operation within the G-code file was unsuccessful. The searched word was not found. The search operation within the library was unsuccessful. The requested G-code file was not found. A request to jump to an N command line was made, but the specified N number was not found.

Alarm No	Description	Suggestions
ALM<089>	The program already exists	The G-code file to be created already exists in the library. Create it with a different name.

Alarm No	Description	Suggestions
ALM<099>	End of program	The end of the G-code file was reached, but none of the termination commands (M02/M30/M99) were found. Add one of these commands (M02/M30/M99) at the end of the file. M02/M30/M99 were found at the end of the G-code file, but no newline character was added after these commands. Add a blank line at the end of the file. When simulating in reverse, the program start was reached. You can press the reset button to continue. A request to jump to an N command line was made, but the specified N number was not found.

Alarm No	Description	Suggestions
ALM<100>	Restart the system	The unit of measurement has been changed with G20 or G21. Turn off the system's power and turn it on again. The new unit of measurement will be valid when the system restarts.

Alarm No	Description	Suggestions
ALM<109>	Machine panel communication timeout	Machine panel communication could not be started or has timed out. Check the Ethernet cable used in the machine panel line. Ensure that the cable is at least CAT6. Verify that all grounding lines of the machine comply with the standards. Ensure that the machine is properly grounded according to the standards. Check that there is no issue with the power line of the machine panel. If the machine panel is not in use, enter the appropriate setting value for SPRM19.

Alarm No	Description	Suggestions
ALM<110>	Real time communicatıon initializatıon error	Communication with the servo drives could not be started. Ensure that the cables are properly connected. Verify that the Ethernet cable used for ECAT and RTEX models is at least CAT6. Ensure that the SPRM0-SPRM16 parameters are properly configured. Check that all servo drives are powered on. Verify that all grounding lines of the machine comply with the standards. Ensure that the machine is grounded according to the standards. Ensure that the Pulser3 model used is compatible with the same protocol as the servo drives.

Alarm No	Description	Suggestions
ALM<113>	External I/O module communication timeout	Communication with the external I/O module could not be started or has timed out. Check the Ethernet cable used in the external I/O module line. Ensure that the cable is at least CAT6. Verify that all grounding lines of the machine comply with the standards. Ensure that the machine is grounded according to the standards. Check that there is no issue with the power line of the machine panel. Verify that the SPRM32-SPRM41 parameters are properly configured according to the number of external I/O modules used. Check that the selection buttons on the external I/O modules are properly configured. Ensure that there is no issue with the power line of the external I/O modules.

Alarm No	Description	Suggestions
ALM<114>	Parameter has been changed	A parameter has been changed by the user, and this alarm was triggered for safety purposes. If any measurement(PRM24-PRM39) or feedback calibration parameters have been modified, turn off the system and turn it on again. For other parameters, press the Reset button.

Alarm No	Description	Suggestions
ALM<120>	Communication with the servo drive with serial number 1 could not be established	Communication with the servo drive specified by the slot number cannot be established. Ensure that the cables are properly connected. Verify that the Ethernet cable used in ECAT and RTEX models is at least CAT6. Make sure that the SPRM0-SPRM16 parameters are properly configured. Check that all servo drives are powered on. Verify that all grounding lines of the machine comply with the standards. Ensure that the machine is grounded according to the standards. Ensure that the Pulser3 model used is compatible with the same protocol as the servo drives.
ALM<121>	Communication with the servo drive with serial number 2 could not be established	
ALM<122>	Communication with the servo drive with serial number 3 could not be established	
ALM<123>	Communication with the servo drive with serial number 4 could not be established	
ALM<124>	Communication with the servo drive with serial number 5 could not be established	
ALM<125>	Communication with the servo drive with serial number 6 could not be established	
ALM<126>	Communication with the servo drive with serial number 7 could not be established	
ALM<127>	Communication with the servo drive with serial number 8 could not be established	

Alarm No	Description	Suggestions
ALM<128>	Data reading error	Data reading from internal memory has failed. Please contact HSC Kontrol.

Alarm No	Description	Suggestions
ALM<129>	Data writing error	Data writing to internal memory has failed. Please contact HSC Kontrol.

Alarm No	Description	Suggestions
ALM<130>	Parameters could not be read	Parameters saved to internal memory could not be read properly. Please contact HSC Kontrol.

Alarm No	Description	Suggestions
ALM<131>	Parameters could not be written	Parameters saved to internal memory could not be written properly. Please contact HSC Kontrol.

Alarm No	Description	Suggestions
ALM<132>	Invalid parameter value	One or more values entered for the parameters are outside the allowed range. Please adjust the entered parameter values according to the allowed range

Alarm No	Description	Suggestions
ALM<133>	System parameters could not be read	The system parameters saved in the internal memory could not be read properly. Please contact HSC Kontrol.

Alarm No	Description	Suggestions
ALM<134>	The system parameters could not be written	The system parameters saved in the internal memory could not be written correctly. Please contact HSC Kontrol.

Alarm No	Description	Suggestions
ALM<135>	Invalid system parameter value	One or more values entered for the system parameters are outside the allowed range. Please adjust the entered system parameter values to be within the permitted range.

Alarm No	Description	Suggestions
ALM<140>	The program could not be created	The G-code file could not be created. Ensure that the SD card is inserted in the Pulser3. Check that the SD card is formatted in FAT32. Replace the SD card if necessary.

Alarm No	Description	Suggestions
ALM<141>	The program could not be deleted	The G-code file could not be deleted. Ensure that the SD card on the Pulser3 is formatted with FAT32. If necessary, replace the SD card.

Alarm No	Description	Suggestions
ALM<142>	The program could not be read	The G-code file could not be read. Ensure that the SD card is inserted into the Pulser3. Check that the SD card is formatted with FAT32. Verify that the subprogram to be called exists on the SD card. Ensure the subprogram is named OXXXX.cnc (XXXX: 0-9999). Replace the SD card.

Alarm No	Description	Suggestions
ALM<143>	The program could not be written	The G-code file could not be written. Check that the SD card on the Pulser3 is formatted as FAT32. Replace the SD card.

Alarm No	Description	Suggestions
ALM<150>	Tool offset values could not be read	The tool offset values stored in the internal memory could not be read properly. Please contact HSC Kontrol.

Alarm No	Description	Suggestions
ALM<151>	The tool offset values could not be written	The tool offset values saved in the internal memory could not be written correctly. Please contact HSC Kontrol.

Alarm No	Description	Suggestions
ALM<152>	The part zero offset values could not be read	The part offset values stored in the internal memory could not be read correctly. Please contact HSC Kontrol.

Alarm No	Description	Suggestions
ALM<153>	The part offset values could not be written	The part offset values stored in the internal memory could not be written properly. Please contact HSC Kontrol.

Alarm No	Description	Suggestions
ALM<154>	Holding user variables could not be read	The holding user variables saved in the internal memory could not be read correctly. Please contact HSC Kontrol..

Alarm No	Description	Suggestions
ALM<155>	Holding user variables could not be written	The holding user variables stored in internal memory could not be written properly. Please contact HSC Kontrol.

Alarm No	Description	Suggestions
ALM<156>	Stored general data could not be read	General values such as the last selected tool, the number of parts produced, and the selected measurement unit stored in internal memory could not be read properly. Please contact HSC Kontrol.

Alarm No	Description	Suggestions
ALM<157>	Stored general data could not be written	The stored general data, such as the last selected tool, the number of produced parts, and the selected unit of measurement, could not be written properly. Please contact HSC Kontrol.

Alarm No	Description	Suggestions
ALM<158>	Holding data of the internal PLC could not be read	The internal PLC permanent memory area stored in the internal memory could not be read properly. Please contact HSC Kontrol.

Alarm No	Description	Suggestions
ALM<159>	The internal PLC holding data could not be written	The internal PLC permanent memory area stored in the internal memory could not be written correctly. Please contact HSC Kontrol.

Alarm No	Description	Suggestions
ALM<170>	Network Initialization Error	TCP connection cannot be established or proper read/write operation cannot be performed. Please contact HSC Kontrol.
ALM<171>	Network Data Read Error	
ALM<172>	Network Data Write Error	
ALM<173>	Unknown Error in Network Read/Write	

Alarm No	Description	Suggestions
ALM<190>	X Axis Servo Alarm	The servo driver of the specified axis is in an alarm state. Check the error code on the servo driver and refer to the documentation of the used servo model.
ALM<191>	Y Axis Servo Alarm	
ALM<192>	Z Axis Servo Alarm	
ALM<193>	4th Axis Servo Alarm	
ALM<194>	5th Axis Servo Alarm	
ALM<195>	6th Axis Servo Alarm	
ALM<196>	7th Axis Servo Alarm	
ALM<197>	Eksen 8 Servo Alarm	

Alarm No	Description	Suggestions
ALM<200>	X axis is at the (+) direction limit switch	The specified axis has reached the positive (+) direction limit switch. Move the axis in the reverse direction. Check the limit switch and stopper. Ensure the switch connection is secure.
ALM<201>	Y axis is at the (+) direction limit switch	
ALM<202>	Z axis is at the (+) direction limit switch	
ALM<203>	4th axis is at the (+) direction limit switch	
ALM<204>	5th axis is at the (+) direction limit switch	
ALM<205>	6th axis is at the (+) direction limit switch	
ALM<206>	7th axis is at the (+) direction limit switch	
ALM<207>	8th axis is at the (+) direction limit switch	

Alarm No	Description	Suggestions
ALM<210>	X axis is at the (-) direction limit switch	The specified axis has reached the negative (-) direction limit switch. Move the axis in the reverse direction. Check the limit switch and stopper. Ensure the switch connection is secure.
ALM<211>	Y axis is at the (-) direction limit switch	
ALM<212>	Z axis is at the (-) direction limit switch	
ALM<213>	4th axis is at the (-) direction limit switch	
ALM<214>	5th axis is at the (-) direction limit switch	
ALM<215>	6th axis is at the (-) direction limit switch	
ALM<216>	7th axis is at the (-) direction limit switch	
ALM<217>	8th axis is at the (-) direction limit switch	

Alarm No	Description	Suggestions
ALM<220>	The position difference of X axis is too high	The difference between the commanded position and the feedback position of the specified axis is greater than the value specified in PRM120-PRM125. The servo system is unable to follow the command. Check the load on the servo. A command is being sent above the maximum pulse output frequency of 100kHz for the PLSE model. Reduce the axis operating speed or adjust the axis measurement calibration values.
ALM<221>	The position difference of Y axis is too high	
ALM<222>	The position difference of Z axis is too high	
ALM<223>	The position difference of 4th axis is too high	
ALM<224>	The position difference of 5th axis is too high	
ALM<225>	The position difference of 6th axis is too high	
ALM<226>	The position difference of 7th axis is too high	
ALM<227>	The position difference of 8th axis is too high	

Alarm No	Description	Suggestions
ALM<230>	X Axis has reached the positive (+) software limit	The specified axis has reached the positive (+) software limit. Move the axis in the opposite direction.
ALM<231>	Y Axis has reached the positive (+) software limit	
ALM<232>	Z Axis has reached the positive (+) software limit	
ALM<233>	4th Axis has reached the positive (+) software limit	
ALM<234>	5th Axis has reached the positive (+) software limit	
ALM<235>	6th Axis has reached the positive (+) software limit	
ALM<236>	7th Axis has reached the positive (+) software limit	
ALM<237>	8th Axis has reached the positive (+) software limit	

Alarm No	Description	Suggestions
ALM<240>	X Axis has reached the negative (-) software limit	The specified axis has reached the negative (-) software limit. Move the axis in the opposite direction.
ALM<241>	Y Axis has reached the negative (-) software limit	
ALM<242>	Z Axis has reached the negative (-) software limit	
ALM<243>	4th Axis has reached the negative (-) software limit	
ALM<244>	5th Axis has reached the negative (-) software limit	
ALM<245>	6th Axis has reached the negative (-) software limit	
ALM<246>	7th Axis has reached the negative (-) software limit	
ALM<247>	8th Axis has reached the negative (-) software limit	

Alarm No	Description	Suggestions
ALM<252>	Tool life counter has reached the target value	Reset the tool life counter.

Alarm No	Description	Suggestions
ALM<253>	Tip touch error	Check the consumables.

Alarm No	Description	Suggestions
ALM<256>	PLC Alarm 0	These alarms are defined by the machine manufacturer. Please contact the
ALM<257>	PLC Alarm 1	
ALM<258>	PLC Alarm 2	
ALM<259>	PLC Alarm 3	
ALM<260>	PLC Alarm 4	
ALM<261>	PLC Alarm 5	
ALM<262>	PLC Alarm 6	
ALM<263>	PLC Alarm 7	
ALM<264>	PLC Alarm 8	
ALM<265>	PLC Alarm 9	
ALM<266>	PLC Alarm 10	
ALM<267>	PLC Alarm 11	
ALM<268>	PLC Alarm 12	
ALM<269>	PLC Alarm 13	
ALM<270>	PLC Alarm 14	
ALM<271>	PLC Alarm 15	
ALM<272>	PLC Alarm 16	
ALM<273>	PLC Alarm 17	
ALM<274>	PLC Alarm 18	
ALM<275>	PLC Alarm 19	
ALM<276>	PLC Alarm 20	
ALM<277>	PLC Alarm 21	
ALM<278>	PLC Alarm 22	
ALM<279>	PLC Alarm 23	
ALM<280>	PLC Alarm 24	
ALM<281>	PLC Alarm 25	
ALM<282>	PLC Alarm 26	
ALM<283>	PLC Alarm 27	
ALM<284>	PLC Alarm 28	
ALM<285>	PLC Alarm 29	
ALM<286>	PLC Alarm 30	
ALM<287>	PLC Alarm 31	

Alarm No	Description	Suggestions
ALM<288>	Makro Alarm 0	These alarms are defined by the machine manufacturer. Contact the machine manufacturer.
ALM<289>	Makro Alarm 1	
ALM<290>	Makro Alarm 2	
ALM<291>	Makro Alarm 3	
ALM<292>	Makro Alarm 4	
ALM<293>	Makro Alarm 5	
ALM<294>	Makro Alarm 6	
ALM<295>	Makro Alarm 7	
ALM<296>	Makro Alarm 8	
ALM<297>	Makro Alarm 9	
ALM<298>	Makro Alarm 10	
ALM<299>	Makro Alarm 11	
ALM<300>	Makro Alarm 12	
ALM<301>	Makro Alarm 13	
ALM<302>	Makro Alarm 14	
ALM<303>	Makro Alarm 15	