NCT® 3xxM

Control for Milling Machines and Machining Centers

Programmer's Manual

From SW version n.16.0

Manufactured by **NCT Automation kft**.
H-1148 Budapest Fogarasi út 7
☎ Phone: (+36 1) 467 6300
☎ Fax:(+36 1) 467 6309
E-mail: nct@nct.hu
Home page: www.nct.hu

ATTENTION!

☞***Note: Please read the changes carefully before using new software!***

New features and differences of software version 16 compared to version 15

10.2.3 Compensating the Angular Position of the Workpiece: on page 95

The misalignment compensation can now be specified along all three rotary axes at the same time, and is not limited to just one axis.

25.20 Compensating the Angular Position of the Workpiece on 5-axis Machines: on page 441

Values of misalignment compensation are also taken into account in case of rotary axes working continuously.

15.1 Compensation Memory. Referring to Tool Compensation (H and D): on page 148

In the milling machine channel, the structure of and managing the tool compensation memory have changed if lathe compensations are also used. See also the following subchapters:

15.4 Call for Lathe Compensations (G43.7): on page 161
23.1.16 Value of the Actual Length Compensation: on page 297
23.1.18 Values of the Tool Compensation Memory: on page 298

20.8 Managing Non-perpendicular Axes: on page 265

25.13 Managing Other Transformations after G68.2: on page 403

Contents

March 12, 2024

1 Introduction

1.1 Part Program

The Part Program is a set of instructions that can be interpreted by the control system in order to control the operation of the machine.
The Part Program consists of blocks which, in turn, comprise words.

Word: Address and Data
Each word is made up of two parts - an address and a data. The address has one or more characters, the data is a numerical value (an integer or decimal value). Some addresses may be given a sign or operator I. In the program, addresses can be given using small letter or capital letter too, the control accepts both variants.

Address Chain:

Address	Meaning	Value limits
O	program number	0001 - 9999
/	optional block	1 - 9
N	block number	1 - 99999999
G	preparatory function	*
X, Y, Z, U, V ,W	length coordinates	I, –, *
A, B, C	angular coordinates, length coordinates, auxiliary functions	I, –, *
R	circle radius, auxiliary data	I, –, *
I, J, K	circle center coordinate, auxiliary coordinate	–, *
E	auxiliary coordinate	–, *
F	feed rate	*
S	spindle speed	*
M	miscellaneous function	1 - 99999999
T	tool number	1 - 99999999
H, D	number of length and radius compensation	1 - 999
L	repetition number	1 - 99999999
P	auxiliary data, dwell time	*
Q	auxiliary data	*
,C	distance of chamfer	–, *
,R	radius of fillet	–, *
,A	angle of straight line	–, *

At an address marked with a * in the Value Limits column, the data may have a decimal value as well.

At an address marked with **I** and –, an incremental operator or a sign can be assigned, respectively.
The positive sign + is not indicated and not stored.

Block

A block is made up of words.
The blocks are separated by characters ⌙ (**L**ine **F**eed) in the memory. The use of a block number is not mandatory in the blocks.

Main Program and Subprogram

The part programs may be divided into two main groups:
main programs, and
subprograms.
The procedure of machining a part is described in the main program. If, in the course of machining repeated patterns should be machined at different places, it will not be necessary to write those program-sections over and over again in the main program, instead, a subprogram will have to be organized, which can be called from any place (even from another subprogram). The user can return from the subprogram to the calling program.

Program Format in the Memory

The ***part programs*** stored in the memory ***are ASCII code text files***.

1.2 Channel

Generally, at one time on one machine one tool is working motion of which is controlled by the part program through the control.
We speak about multichannel operation, when the same control simultaneously directs the path of two or more tools independent of the other's motion, executing two or more part programs.
For each channel it can be assigned a part program to execute.
Multichannel operation is required too, when at a time only one tool is working, but on the machine there is a workpiece feeder for which motions of the workpiece should be programmed while the tool is cutting.
Synchronization points can be programmed for those points of the programs, where ***tool path motions should wait for each other***.
Each channel has its own workpiece zero point table, tool compensation table and macro variables. The part of tool compensation table and the macro variables assignable on one parameter can be made common for each channel.
In standard version, the NCT3xx controls are single-channel ones, however, optionally, they are able to function as multi-channel ones. As a maximum, 8 channels can be built in a control.
In the control, for each channel, the mode of channel operation can be specified in parameter. In case of the NCT200 controls there could be ***two ones: lathe channel or milling machine channel***. Within a control, the channels can be mixed. For example:
Channel 1: lathe
Channel 2: lathe
Channel 3: milling machine
Hereafter, this manual describes programming the milling machine channel.
If lathe channel is also built in the given control, description of programming for it can be found in the NCT3xxT Control for Lathes Programmer's Manual.

1.3 Fundamental Terms

Interpolation

The control system can move the tool along straight lines and arcs in the course of machining. These activities will be hereafter referred to as 'interpolation'.
Tool movement along a straight line:

program:

```
G01 X__ Y__
```

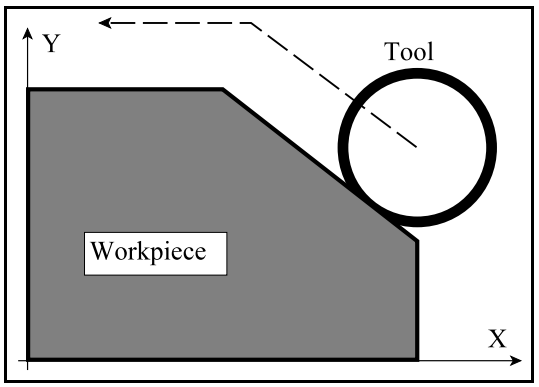


Fig. 1.3-1

Tool movement along an arc:

program:

```
G03 X__ Y__ R__
```

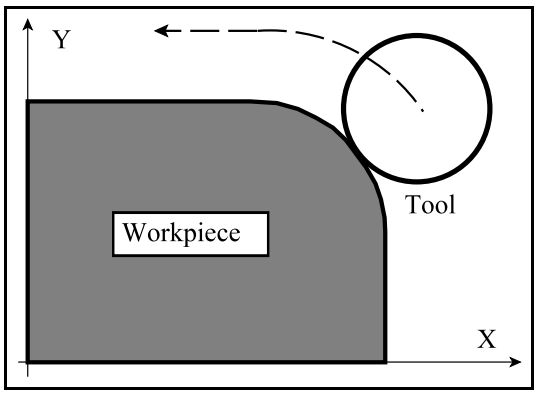


Fig. 1.3-2

Although, in general, the table with the workpiece and not the tool moves, this description will always refer to the motion of the tool against the workpiece.

Preparatory Functions (codes G)

The type of activity to be performed by a given block is described with the use of preparatory functions (also referred to as codes G). E.g., code G01 introduces a linear interpolation.

Feed

The term 'feed' refers to the speed of the tool relative to the workpiece during the process of cutting. The desired feed can be specified in the program at address F and with a numerical value. For example F150 means 150 mm / minute.

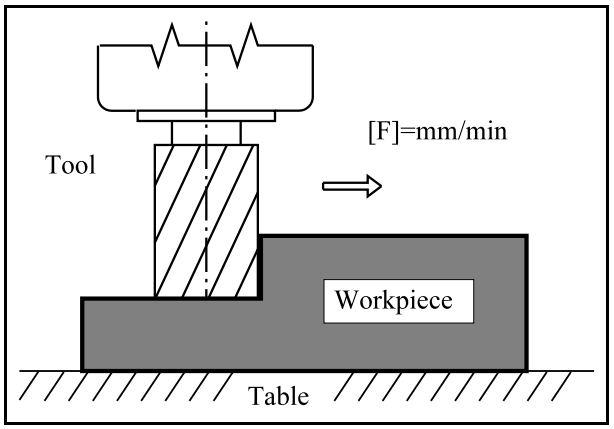


Fig. 1.3-3

Reference Point

The reference point is a fixed point on the machine tool. After power-on of the machine, the slides have to be moved to the reference point. Afterwards the control system will be able to interpret data of absolute coordinates as well.

Coordinate System

The dimensions indicated in the part drawing are measured from a given point of the part. That point is the origin of the workpiece coordinate system. Those dimensional data must be written at the coordinate address in the part program. For example, X30 Y20 Z1 means a coordinate point of 30, 20 and 1mm in the coordinate system of the workpiece in the direction X, Y and Z, respectively.

In order that the control can interpret programmed coordinate data, the distance between the machine zero point and the workpiece zero point must be given. This has to be done by measuring the workpiece zero point.

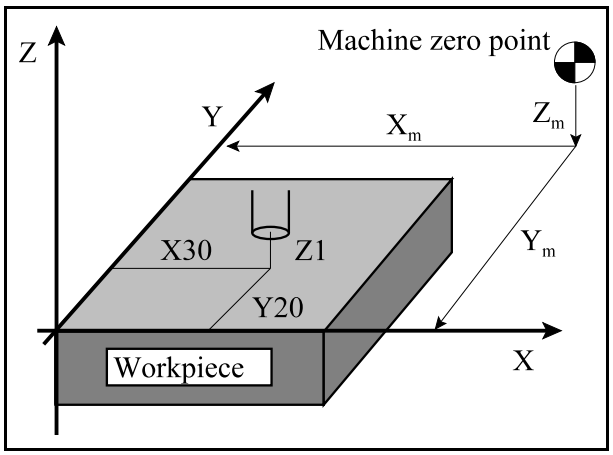


Fig. 1.3-4

Absolute Coordinate Specification

When absolute coordinates are specified, the tool travels a distance measured from the origin of the coordinate system, i.e. to a point whose position has been specified by the coordinates.

The code of absolute data specification is G90.

The instruction line

```
G90 X20 Y30 Z0
```

will move the tool to a point of the above position, irrespective of its position before the instruction has been issued.

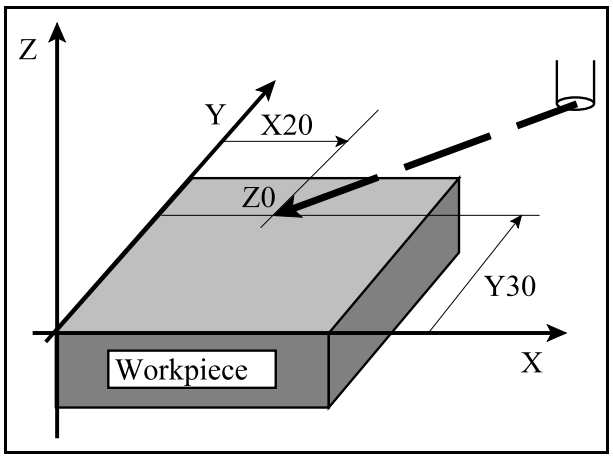


Fig. 1.3-5

Incremental Coordinate Specification

In the case of an incremental data specification, the control system will interpret the coordinate data in such a way that the tool will travel a distance measured from its instantaneous position.

The code of incremental data specification is G91. Code G91 refers to all the coordinate values.

The instruction line

```
G91 X-40 Y-20 Z-5
```

will move the tool over the above distance from its preceding position.

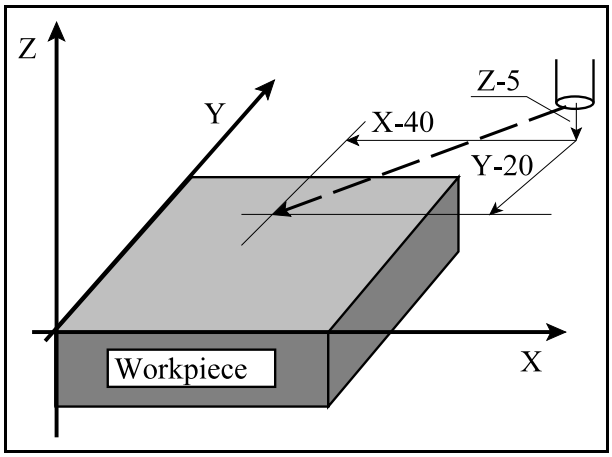


Fig. 1.3-6

Modal Functions

In the programming language, some instructions stays in effect or their value is valid until a instruction of contrary meaning is issued or different value is given to the adequate function. For example: In the program detail

```
N15 G90 G1 X20 Y30 F180
N16 X30
N17 Y100
```

the state of G90 (absolute data specification) and G1 (linear interpolation) and the value of F specified in the block N15 will be modal ones in the blocks N16 and N17. Thus, it is not necessary to specify these codes block by block.

Non-modal (One-Shot) Functions

The effect of some functions or the values of data are valid in a given block. These functions are non-modal or one-shot ones.

Spindle Speed Instruction

The spindle speed can be specified at address S. It is also named S function. Instruction S1500 tells the spindle to rotate at a speed of 1500 rpm.

Tool Function

In the course of machining, different tools have to be used for various cutting operations. Numbers are used to differentiate tools from each other. Tools can be referred by code T. The instruction

```
T25
```

in the program means that the tool No.25 is to be called.

Miscellaneous Functions

A number of switching operations, e.g. starting the spindle, turning on the coolant have to be carried out in the course of machining. These operations can be executed by M (miscellaneous) functions. For example: In the series of instructions

```
M3 M8
```

M3 means rotation of the spindle clockwise, M8 means turning on the coolant.

Tool Length Compensation

In the course of machining, tools of different length are used for the various operations. On the other hand, in production of a bigger series the same operation also has to be performed with tools of different lengths (e.g. when the tool breaks). In order to make the motions described in the part program independent of the length of the tool, i.e. of its offset, the various tool lengths must be set in the control system. If the program is intended to move the tip of the tool to the specified point, the value of the particular length data will have to be called using an attendant code . This is feasible at the address H. For example, the instruction H1 refers to the length data No.1. Henceforth the control will move the tip of the tool to the specified point. This procedure is named setting the length compensation.

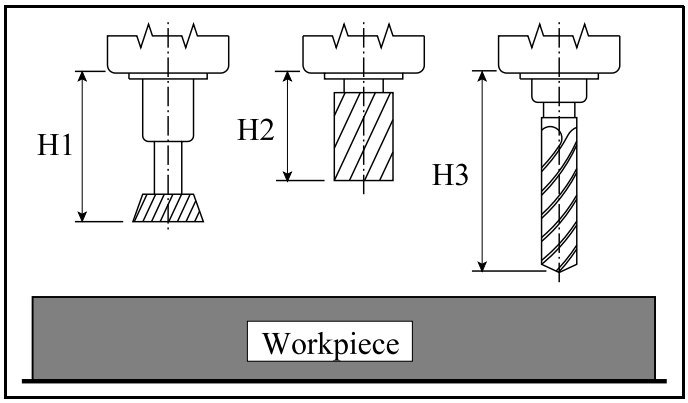


Fig. 1.3-7

Tool Radius Compensation

In the case of milling a contour , the workpiece has to be machined using tools of different radii. In order to write in the program not the path of the tool center taking the radius of tools into account, but the real contour data of the workpiece, introduction of tool radius compensation is necessary. The values of the tool radius compensations must be set in the control. In the program, the tool radius compensation can be referred at the address D.

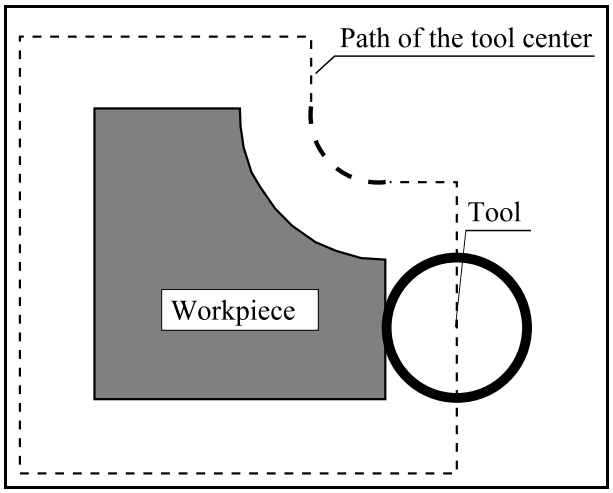


Fig. 1.3-8

2 Controlled Axes

Number of axes in basic configuration	3 axes
Number of expansion axes	Maximum 13 additional axes in the same channel
Maximum number of axes	32 axes altogether in several channels

2.1 Naming and Numbering the Axes

The ***names of the controlled axes*** can be defined in the parameter memory. It can be assigned here, which physical axis has to move to which address.

In the basic configuration, the names of axes in the milling control are: X, Y and Z. These axes will be set in the parameter N0103 Axis to Plane as ***main axes***.
The names of the expansion axes depends on type of the given axis.
Possible names of additional axes performing linear motions are: U, V and W. If these axes are parallel with one of the main directions, the names of the ***expansion axes parallel*** with the axes X, Y and Z will be U, V and W, respectively.

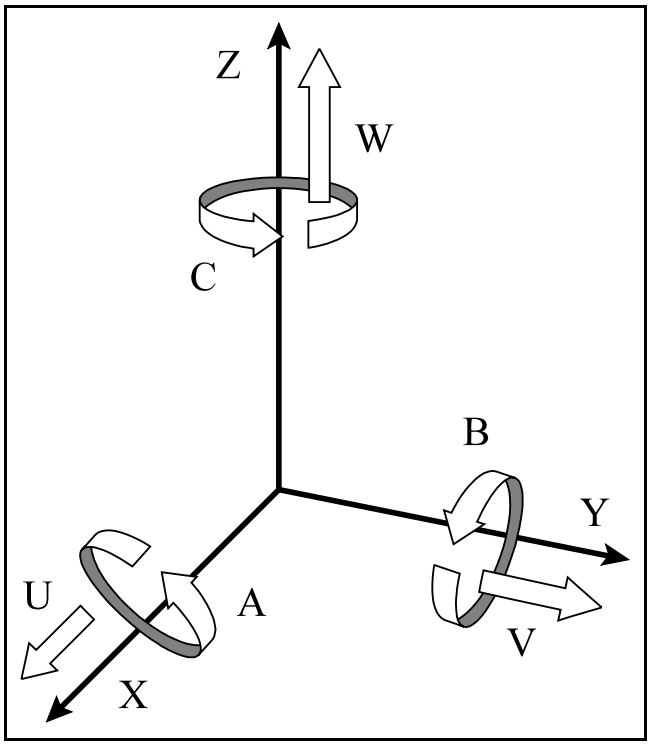


Fig. 2.1-1

Wether a linear expansion axis is parallel with a basic axis or not, it can be set in the parameter N0103 Axis to Plane.
The names of the axes performing rotational motions are: A, B and C. The names of the rotary axes parallel with the directions X, Y and Z will be A, B and C, respectively. Whether an axis is a rotary one or not, it can be set by ROT=1 in the parameter N0106 Axis Properties.

The axes are registered by the control **on the basis of their numbers**. The names of the axes must be assigned to the numbers of the axes. Usually, the axis X is set as the axis 1, the axis Y is set as the axis 2, the axis Z is set as the axis 3.
In certain program instructions, for example in case of referring to the macro variable querying the axis position, not the name of the axis, but the number of the axis must be given.
For numbering please ask the machine tool builder.

2.2 Extended Axis Names

In the case of complex multi-axis machines, the abovementioned 9 letters are not enough to name all the axes. For this reason, extended axis names have been introduced ***to name an axis using*** not one letter but ***up to three characters***.
The first character must be X, Y, Z, U, V, W, A, B, or C necessarily, and it can be specified in the parameter N0100 Axis Name1. This parameter will have to be filled in even if one-character axis names are used only.
The second and the third character names can be specified in the parameter N0101 Axis Name2 and in the parameter N0102 Axis Name3, respectively. They could be the letters of the English alphabet: A, B, C, D, ... Y, Z, or numbers: 0, 1, 2, ..., 9. If the second or third axis name is not used, the value of the parameter will be 0.
Thus, the name of an axis could be XDE, but Z1 and Z2 could also be used.
If the name of the axis ends in a letter, its inherent value could be written next to it. The meaning of

```
XDE127.81
```

is: the axis XDE has to go to the position 127.8.
If the name of the axis ends in a number, equality symbol = will have to be written after the axis. The meaning of

```
Z1=87.257
```

is: the axis Z1 has to go to the position 87.257.
Certainly, even if using extended axis names, it will have to be specified in the parameter N0103 Axis to Plane which ones are main or basic axes, and which ones are parallel axes.
Hereafter in this manual, one-character axis names will be used, in general.

2.3 Assigning Axes to Channels

Using parameters, the builder of the machine tool assigns each axis to different channels. These parameter settings mean the state after the turning on.
The ***axes*** are always assigned to the channels ***in accordance with their number***. So, ***axis number*** in the control is ***global***, while ***axis names*** are ***local***, they are assigned to channels. Certainly, axes with the same name are not allowed within one channel, on the other hand they could be in different channels. The control is capable of managing maximum 16 axes being within one channel.
An example:

Channel 1:	Channel 2:
Axis 1: X	Axis 5: X
Axis 2: Y	Axis 6: Y
Axis 3: Z	Axis 7: Z
Axis 4: C	Axis 8: C

In the course of the machining, it could be necessary to use one or more axes in another channel. In this case, it is allowed to interchange two axes between two channels, or to transfer one axis to another channel. The interchange is executed in accordance with the axis number. After change, the axis name may remain, or may change too.
The interchange of axes is realized by the builder of the machine tool through the PLC program, for example using M function. Description of them is always contained in the manual of the given machine tool.

2.4 Unit System of Axes and Accuracy of Position Display

Coordinate data can be given with an ***accuracy of 15 digit*** as maximum. The decimal point must be put in the only case, when positioning to a point of non-integer coordinate has to be performed. Coordinate data could also have sign. The sign + is omitted.
In the program, data of length coordinates can be given either in ***mm*** or in ***inch***. This is the ***input unit system***. The input unit system can be selected from the program, using code G (G21/G20).
The path-measuring device mounted on the machine tool can measure position either in mm or in inch. The path-measuring device determines the ***output unit system*** which must be given at the bit IND of the parameter N0104 Unit of Measure of the control. On a given machine tool, it is not possible to combine output unit systems among the axes.
If the input and output unit systems are different ones, conversion will be performed by the control automatically.
The rotational axes are always provided with ***degrees*** as units of measure. ***The rotary axes can be designated by ROT=1 in the parameter N0106 Axis Properties. Correct setting of this parameter is important because the control does not execute inch/mm conversion for the axes designated in this way.***
The ***position display accuracy*** (i.e. number of decimal places) can be set in the parameter N0105 Increment System.
The inner position representation of the system is independent of the value of the Increment System parameter. The accuracy is as follows:

In case of linear axes and measurement in mm: 10^{-6} mm;
In case of linear axes and measurement in inch: 10^{-7} inch;
In case of circular axes: 10^{-6} degrees.

System name	Axis	Display accuracy in metric and inch systems	
		G21 metric	**G20 inch**
ISA	linear	0.01 mm	0.001 in
	rotary	0.01 deg	0.01 deg
ISB	linear	0.001 mm	0.0001 in
	rotary	0.001 deg	0.001 deg
ISC	linear	0.0001 mm	0.00001 in
	rotary	0.0001 deg	0.0001 deg
ISD	linear	0.00001 mm	0.000001 in
	rotary	0.00001 deg	0.00001 deg
ISE	linear	0.000001 mm	0.0000001 in
	rotary	0.000001 deg	0.000001 deg

Any axis can be designated for data input and position display in ***diameter*** by setting DIA=1 in the parameter N0106 Axis Properties.

3 Preparatory Functions (Codes G)

The nature of an instruction in a given block is determined by the address G and the number following it.
There are so-called ***one-shot codes G*** effect of which is valid in the given block, and there are so-called ***modal functions G*** effect of which is valid until this effect is switched off or changed by another code G.
The one-shot codes G relate to the group numbered with 0 (zero). Those ones from the modal codes G which influence each other have ***group number*** different from 0 (zero).
More than one code G may be written into one block provided that only one of the functions related to the same group has to be per group.
The ***leading zero has not to be written in the code*** but the control accepts it. For example: Either G01 or G1 may be written in the program.
The table below contains codes G interpreted by the control, their group numbers and functions.

Code G	Group	Function	Page
G0*	1	Positioning	26
G1*		Linear Interpolation	28
G2		Circular and spiral (planar, helical and conical) interpolation, clockwise (CW)	30, 35, 37, 40
G3		Circular and spiral (planar, helical and conical) interpolation, counter-clockwise (CCW)	30, 35, 37, 40
G4	0	Dwell	86
G5.1		Smooth interpolation	51, 55, 435
G9		Exact stop in the given block	72
G7.1		Cylindrical interpolation	48
G10		Programmed data input	98, 155, 374, 127
G11		Programmed data input cancel	127
G10.9		Interchange radius/diameter programming from program	61
G12.1	21	Polar coordinate interpolation on	44
G13.1*		Polar coordinate interpolation off	44
G15*	17	Data specification in polar coordinates off	62
G16		Data specification in polar coordinates on	62

Code G	Group	Function	Page
G17*	02	Selection of the plane X_pY_p	102
G18*		Selection of the plane Z_pX_p	102
G19*		Selection of the plane Y_pZ_p	102
G20*	6	Data input in inch	60
G21*		Data input in mm	60
G22*	04	Programable stroke check function on	272
G23*		Programable stroke check function off	272
G28	0	Programmed return to the reference point	88
G30		Return to the second, third and fourth reference point	89
G31		Skip function	266
G33	01	Thread cutting	42
G37	00	Automatic tool length measurement	269
G38		Holding the radius compensation vector	189
G39		Corner rounding with radius compensation	190
G40*	07	Cancelling calculation of tool radius compensation	86
G41		Calculation of tool radius compensation from the left	162
G42		Calculation of tool radius compensation from the right	162
G43*	8	Tool length compensation on +	156
G44*		Tool length compensation on −	156
G43.1		Tool length compensation in the tool axis direction on	365
G43.4		Type 1 control of the tool center point on	404
		P1: Type 1 tool posture control	423
G43.5		Type 2 control of the tool center point on	404
		P1:Type 2 tool posture control	423
G43.7		Calling the lathe compensations	161
G43.8		Type 1 control of the cutting point and the tool center point on	429
		P1: Type 1 control of the cutting point and the tool posture on	429
G43.9		Type 2 control of the cutting point and the tool center point on	429
		P1: Type 2 control of the cutting point and the tool posture on	429

Code G	Group	Function	Page
G49*		Tool length compensation, TCP off, tool center point, tool posture and cutting point control off	156, 365, 404 429
G50*	11	Scaling off	202
G51		Scaling on	202
G50.1*	22	Mirroring off	205
G51.1		Mirroring on	205
G52	0	Coordinate offset	100
G53		Positioning in the machine coordinate system	92
G53.1		Setting rotary axes in the tool direction	393
G53.6		Setting rotary axes in the tool direction by TCP (G53.6)	419
G54*	14	Selection of the workpiece coordinate system 1	94
G55		Selection of the workpiece coordinate system 2	94
G56		Selection of the workpiece coordinate system 3	94
G57		Selection of the workpiece coordinate system 4	94
G58		Selection of the workpiece coordinate system 5	94
G59		Selection of the workpiece coordinate system 6	94
G54.1		Selection of additional workpiece coordinate system	95
G54.2		Dynamic zero point of rotary tables	371
G61	15	Exact stop mode	73
G62		Automatic corner override mode	73
G63		Override inhibition	73
G64*		Continuous cutting mode	73
G65	0	Simple macro call	328
G66	12	Modal macro call after each motion instruction	329
G66.1		Modal macro call from each block	331
G67*		Cancelling the modal macro call	330, 331
G68	16	In-plane rotation of an object about a given point	200
G69*		In-plane rotation off	200
G68.1	18	Three-dimensional coordinate transformation	368

Code G	Group	Function	Page
G68.2		Defining a plane in space by giving Eulerian angles	383
		P1: Defining a plane in space by rotation around axes	385
		P2: Defining a plane in space by its 3 points	387
		P3: Defining a plane in space by giving 2 vectors	389
		P4: Defining a plane in space by giving projection angles	391
G69.1		Three-dimensional coordinate transformation off	368 383
G70.7	0	Finishing cycle	248
G71.7		Stock removal cycle in turning	247
G72.7		Stock removal cycle in facing	247
G73.7		Pattern repeating cycle	248
G74.7		Face grooving cycle	248
G75.7		Grooving cycle	248
G76.7		Thread cutting cycle	248
G77.7	1	Turning cycle	247
G78.7		Simple thread cutting cycle	247
G79.7		Facing cycle	247
G73	09	High-speed peck drilling cycle	228
G74		Left-handed tapping cycle using spring tap	229
G76		Boring Cycle with Automatic Tool Shift	230
G80*		Cycle state cancel	231
G81		Drilling cycle, rapid-traverse retraction	231
G82		Drilling cycle with dwell, rapid-traverse retraction	232
G83		Peck drilling cycle	233
G84		Right-handed tapping cycle using spring tap	234
G84.2		Right-handed rigid tapping cycle Right-handed peck rigid tapping cycle	235 237
G84.3		Left-handed rigid tapping cycle Left-handed peck rigid tapping cycle	235 237
G85		Boring cycle, retraction with feed	240
G86		Boring Cycle with Retraction with Spindle in Standstill	241
G87		Manual Control/Back Boring Cycle	242

Code G	Group	Function	Page
G88		Boring cycle, manual operation at the bottom point	244
G89		Boring cycle, dwell at the bottom point, retraction with feed	245
G80.8	34	Electronic gear box off	250
G81.8		Electronic gear box on	250
G90*	03	Absolute dimensioning	59
G91*		Incremental dimensioning	59
G92	00	Creating a new work coordinate system	99
G94*	05	Feed per minute	72
G95*		Feed per revolution	72
G96	13	Calculation of constant cutting speed on	108
G97*		Calculation of constant cutting speed off	108
G98*	10	Return from the drilling cycle to the initial point	204
G99		Return from the drilling cycle to the (approach) point R	204

Basic State after Turning On

In a group codes G marked with * represent state the control takes on ***after turning on***.
Where there are several codes G marked with * in a group, on the basis of the parameters N1300 DefaultG1 and N1301 DefaultG2 can it be selected which ones have to be valid ***after turning on.***
These codes G are as follows:

G00, G01;
G17, G18, G19;
G20, G21;
G22, G23;
G43, G44, G49;
G90, G91;
G94, G95.

Basic State after Pushing Reset Key or after Program End

When ***reset*** key is pushed or a ***program ends*** (M2, M30), if the bit CLR of the parameter N1301 DefaultG2
= 0: the control, without any condition, will take on state marked with * in the G code table, or it will reset the code G values into after-turning-on basic state set in the parameters N1300 DefaultG1 and N1301 DefaultG2;
= 1: the control, on the basis of values given in the parameter CLR G Table1, 2, 3, 4, 5, gets into basic state according to the G code groups, or it leaves the modal values unchanged.
If in the parameter CLR G Table1, 2, 3, 4, 5 the G code group's bit Cnn (where nn is the group number of the code G) is
= 0: the control will set the adequate code G group into basic state;
= 1: the control will leave the adequate code G group in emerged and inherited state.

4 Interpolation

4.1 Positioning (G0)

The positioning instruction G0 moves the tool along all the axes programmed in the block to the specified point.
Motion is performed using rapid traverse. The value of rapid traverse is specified by the builder of the machine tool in parameter for each axis, and it cannot be set from the program.
In the case of absolute dimensioning, the tool moves to the point of given coordinates in the coordinate system of the actual workpiece.
In the case of incremental dimensioning , the tool moves the given distance from its actual position.

The format of the block is:

G0 v

where v are the coordinates given in the block. The designation v refers here (and hereafter) to all the controlled axes used in the given channel. Simultaneous positioning along all the axes of the channel is possible.
Instead of G0, G00 can be given too.
An example:

```
G0 X20 Y30 Z0
```

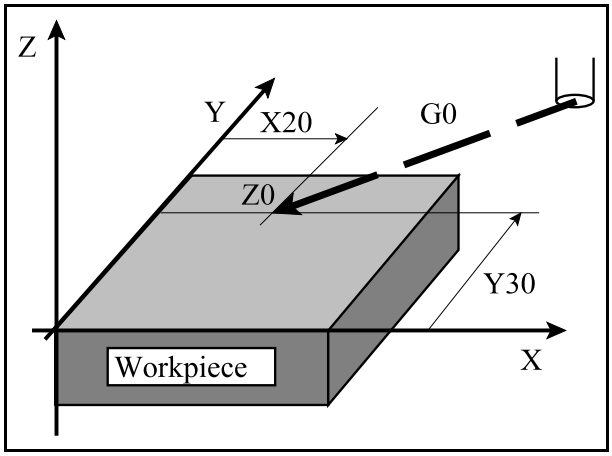


Fig. 4.1-1

Another code G and function can also be given in the block

An example:

```
G0 G90 X20 Y30 Z0 S2000 M3
```

The modal code G0 remains effective until it is re-written by another interpolation instruction.
An example:

```
G0 X20 Y30 Z0
X10 Z1 (positioning block, because G0 is being modal)
```

After turning on, code G0 will in effect if bit position of the parameter N1300 DefaultG1 is G01=1.

4.1.1 Positioning by Linear Interpolation

In the case of moving several axes simultaneously, the control will move the tool along a straight line connecting the starting point and the end point while positioning, if bit position of the parameter N0421 Acc Contr is ROL=0. The control calculates the resultant velocity vector (**v**) in such a way that positioning be executed in a minimum time, and the value of velocity along none of the axes does not exceed the rapid traverse value set for the given axis. After completing motion, the control will check the 'in position' signal if bit position of the parameter N1337

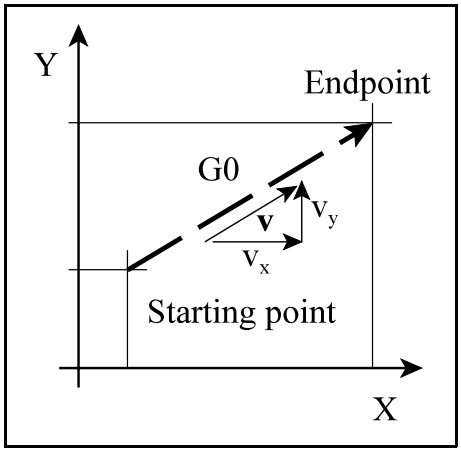


Fig. 4.1.1-1

Execution Config is PCH=1.
The control waits for the 'in-position' signal for the time set in the parameter N1340 Inpos Timeout ; if the signal does not received even from this time on, the control will give warning *2501 Position error*.
The maximum acceptable deviation from the position can be specified in the parameter N0516 Inpos, for each axis.
In-position check should be set when it is reasonable, otherwise the execution time could increase. For example, executing the program detail

```
G0 Z10
X30
Z1
```

the difference between the two time periods will be as it is illustrated in the figure.

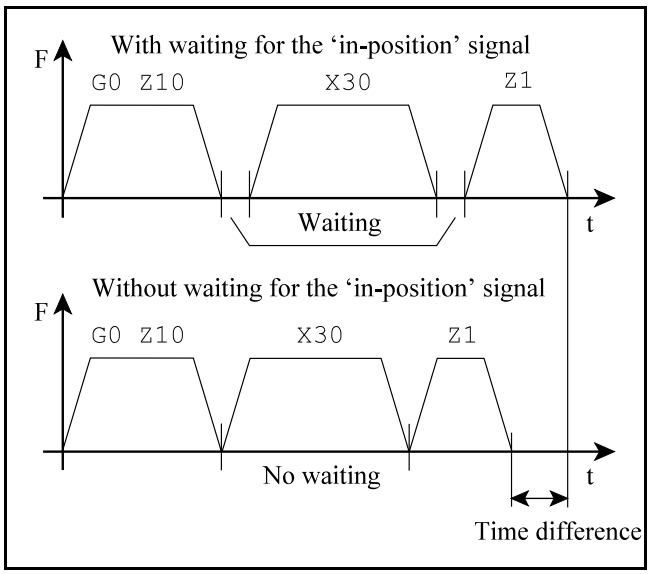


Fig. 4.1.1-2

The control ***always carries out in-position check*** at the end of the block
in state G61 (exact stop mode), or
in the positioning block in which code G9 (exact stop) has been written, even at parameter position PCH=0 (no in-position check).

4.1.2 Positioning by Overlapping Rapid Traverse Motions

Consecutive positionings carried out on different axes can be accelerated further by the use of overlapping the motion of the positioning blocks. This means, that while in a positioning block one of the axes decelerates approaching the end point position, an other axis being in the next positioning block begins accelerate already.
Overlapping will be switched on by the bit position ROL=1 of the parameter N0421 Acc Contr.
The percentage of speed, after reaching which in the deceleration period of the previous block the next block starts, can be set in the parameter N0422 Rapid Reduct. Ratio, in percentage.
Using the former example, executing the program detail

```
G0 Z10
```

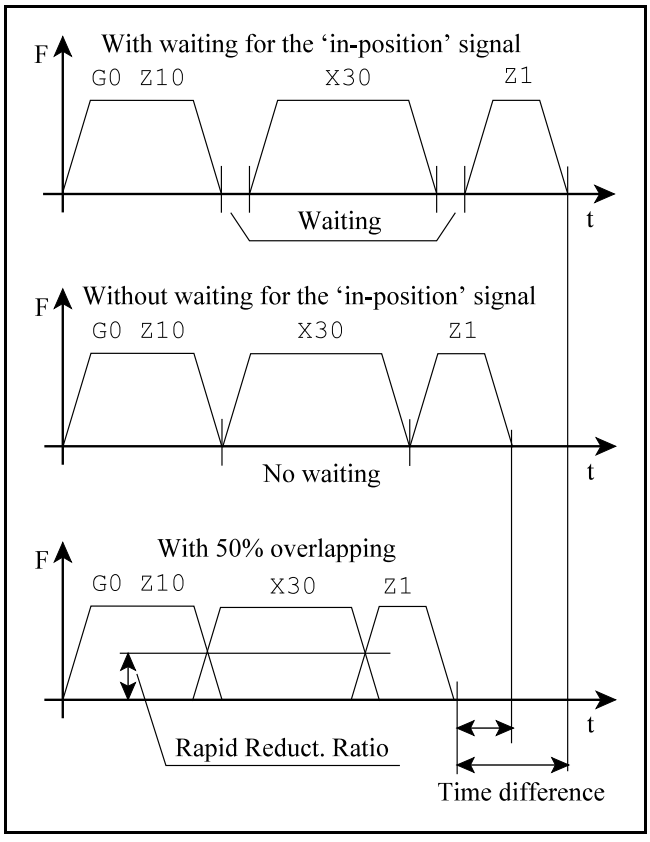


Fig. 4.1.2-1

```
X30
Z1
```

the difference between execution times will be as it is illustrated in the figure.

In the case of positioning with overlap, tool path is not cornered, but it is a rounded one. Because of this, care should be taken in retracting the tool from the workpiece; maybe degree of retracting must be higher than usual in the program.
In the case of positioning ***several axes*** programmed in one block, the ***tool motion path is straight approximately only***, and the different axes arrive to the position with time difference.

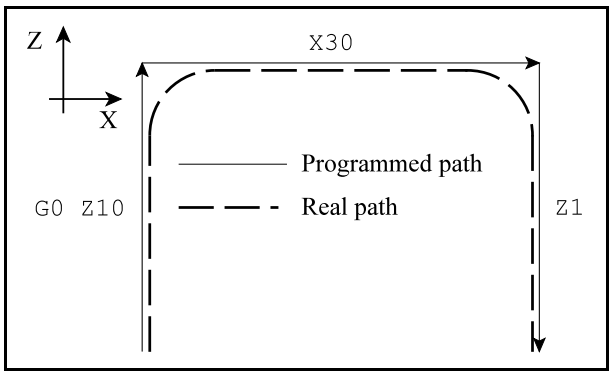


Fig. 4.1.2-2

The control ***suspends overlapping*** between positioning blocks and ***always carries out in-position check*** at the end of the block

in the state G61 (exact stop mode), or in the positioning block in which code G9 (exact stop) has been written, even at parameter position PCH=0 (there is no in-position check)

4.2 Linear Interpolation (G1)

The linear interpolation instruction G1 moves the tool to the specified point along straight path, along all the axes programmed in the block.
Motion is carried out at feed F programmed in block, or being modal.
In the case of absolute data specification, the tool moves in the actual workpiece coordinate system to the point of specified position.
In the case of incremental data specification, the tool steps the specified distance from its actual position.

The format of the block is:

G1 v F

where v are the coordinates given in the block, and F is the value of the feed. Motion along all the axes of the channel at the same time is possible.
Instead of G1, G01 can be given too.

An example:

```
G1 X80 Y115 F500
```

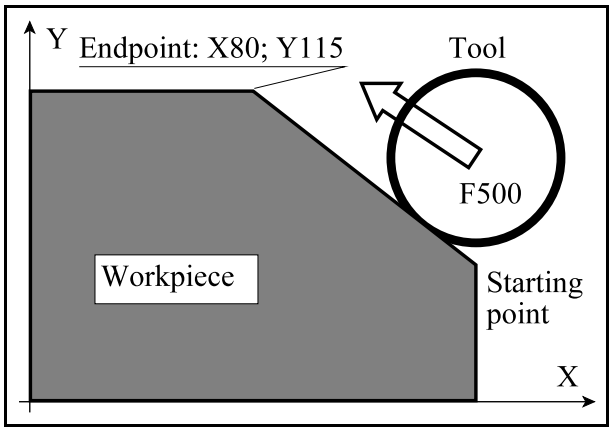


Fig. 4.2-1

In the block, other code G and function can also be given.
An example:

```
G1 G91 X50 Y30 Z10 S2000 M3
```

The modal code G1 remains effective until it is re-written by another interpolation instruction. An example:

```
G1 X80 Y115 F500
X0 (linear interpolation by F500, because G1 and F are being
modal)
```

After turning on, code G1 will be in effect if bit position of the parameter N1300 DefaultG1 is G01=1.

The feed programmed at the address F is valid always along the path programmed. Its axial components are as follows:

Feed along the axis X is: $F_x = \frac{\Delta x}{L} F$

Feed along the axis Y is: $F_y = \frac{\Delta y}{L} F$

.....

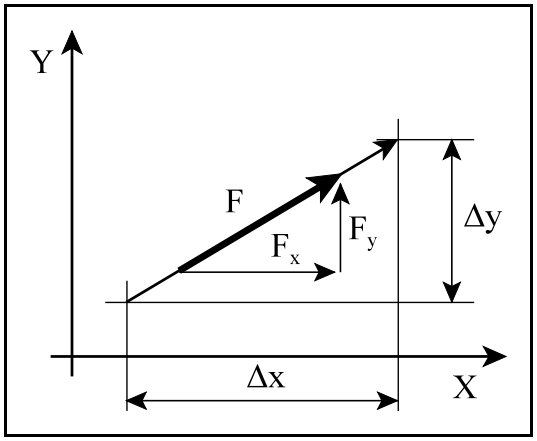


Fig. 4.2-2

The formula continues for all the axes programmed in the block.

where: Δx, Δz, ...: displacements calculated along the respective axes,

L: length of the programmed displacement:

$$L = \sqrt{\Delta x^2 + \Delta y^2 + \ldots}$$

The feed along a rotational axis is interpreted in unit of degrees per minute (°/min). In the block

```
G1 C270 F120
```

F120 means: 120 deg/min.

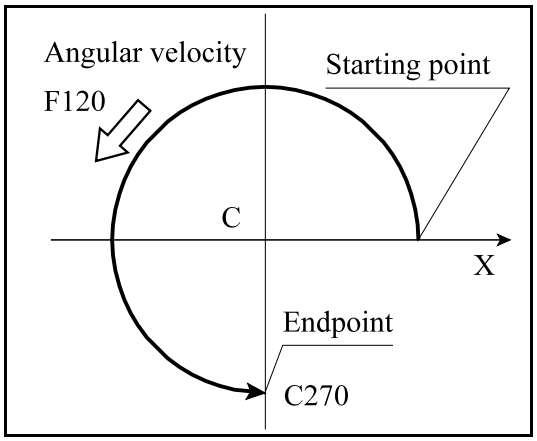


Fig. 4.2-3

If motions of a linear axis and a rotary axis are combined through linear interpolation, the feed components will be distributed according to the above formulas.

For example, in the block:

```
G91 G01 Z100 C45 F120
```

the feed components in the Z and B directions are as follows:

Feed along the axis Z is: $F_z = \frac{100}{\sqrt{100^2 + 45^2}} 120 = 109.4$ mm/min

Feed along the axis C is: $F_c = \frac{45}{\sqrt{100^2 + 45^2}} = 49.2$ °/min

4.3 Circular Interpolation (G2, G3)

The G2 or G3 instruction moves the tool in the selectes plane to the point specified in the block, along circular arc. Motion is carried out at feed F programmed in block, or being modal.
The format of the block is:

$$G17\begin{Bmatrix}G2\\G3\end{Bmatrix}X_p\_Y_p\_\begin{Bmatrix}R\_\\I\_J\_\end{Bmatrix}F\_$$

$$G18\begin{Bmatrix}G2\\G3\end{Bmatrix}X_p\_Z_p\_\begin{Bmatrix}R\_\\I\_K\_\end{Bmatrix}F\_$$

$$G19\begin{Bmatrix}G2\\G3\end{Bmatrix}Y_p\_Z_p\_\begin{Bmatrix}R\_\\J\_K\_\end{Bmatrix}F\_$$

Circular interpolation is accomplished in the plane selected by the instructions G17, G18, G19; in the case of ***G2*** in ***clockwise*** direction, and in the case of G3 in ***counter-clockwise*** direction:

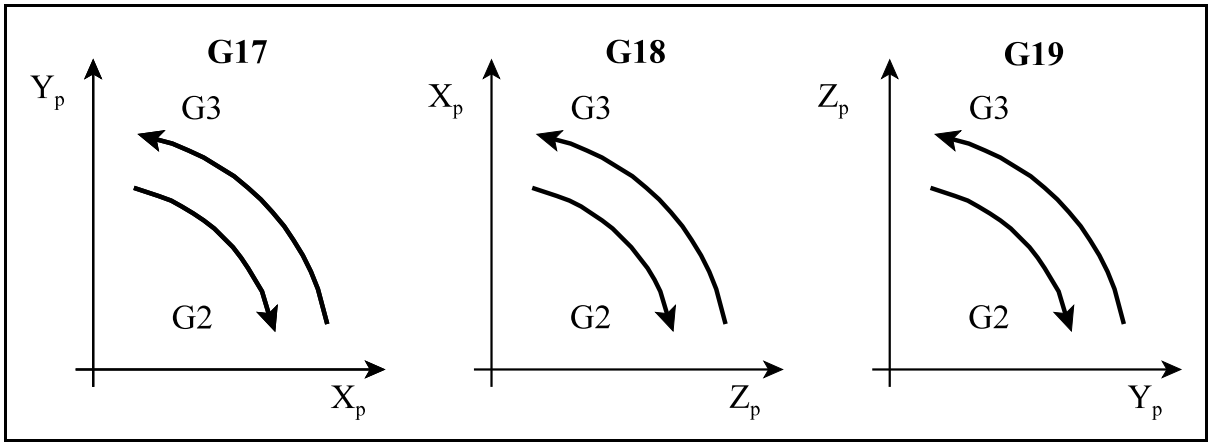


Fig. 4.3-1

Instead of codes G2 and G3, codes G02 and G03 can also be written in the program.
The codes G2 and G3 are modal ones, they remain effective until they are re-written by another interpolation instruction.

Here and hereafter, the meanings of X_p, Y_p, and Z_p are as follows:

X_p: axis X or an axis parallel with it;
Y_p: axis Y or an axis parallel with it;
Z_p: axis Z or an axis parallel with it.

The values of ***X_p, Y_p, and Z_p*** are ***the coordinates of the circle endpoint*** in the given coordinate system, specified as absolute data, or incremental data measured from the starting point.

Further data of the circle can be specified in two different ways:

Case 1: Specifying the circle by its radius at the address R

In this case, the control automatically calculates the coordinates of the circle center from the coordinates of the starting point (this is the point where the control is in the moment of reading-in the block of the circle), from the coordinates of the end point (values defined at the addresses X_p, Y_p and Z_p) and from the programmed circle radius R. In the case of a given circulation direction (G2 or G3), two different circle of radius R can be drawn passing the starting and end points.

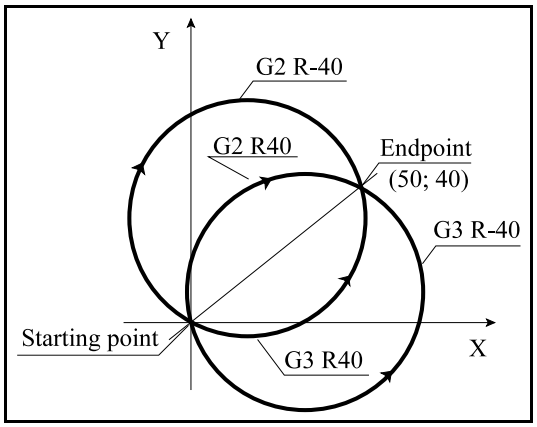


Fig. 4.3-2

If the circle radius ***R*** is given
as a ***positive*** number, the control will interpolate an arc ***smaller than 180°***,
as a ***negative*** number, the control will interpolate an arc ***larger than 180°***. For example:

Arc section 1: `G2 X50 Y40 R40`
Arc section 2: `G2 X50 Y40 R-40`
Arc section 3: `G3 X50 Y40 R40`
Arc section 4: `G3 X50 Y40 R-40`

Case 2: Specifying the circle by its center at the addresses I, J, K

The control interprets the values specified at the addresses I, J and K ***incrementally*** in such a way that the vector defined by the values I, J and K points ***from the starting point to the center point*** of the circle.

The values ***I, J and K*** must always be specified in ***radius***, even if the axes adherent to them are set to be programmed in diameter.

For example:

in the case of G17: `G3 X10 Y70 I-50 J-20`
in the case of G18: `G3 X70 Z10 I-20 K-50`
in the case of G19: `G3 Y10 Z70 J-50 K-20`

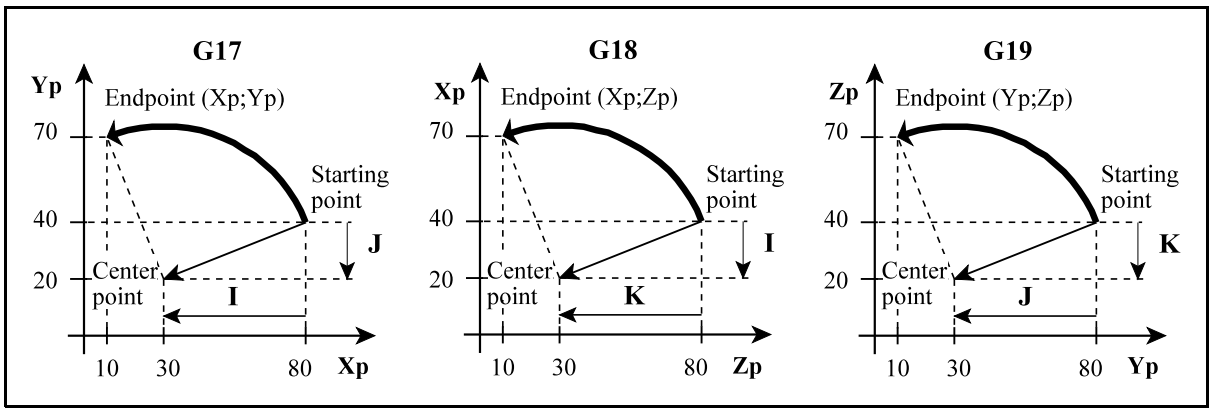


Fig. 4.3-3

At the address F, the feed along the path can be programmed which points always in the direction of the circle tangent and is constant along the whole path.

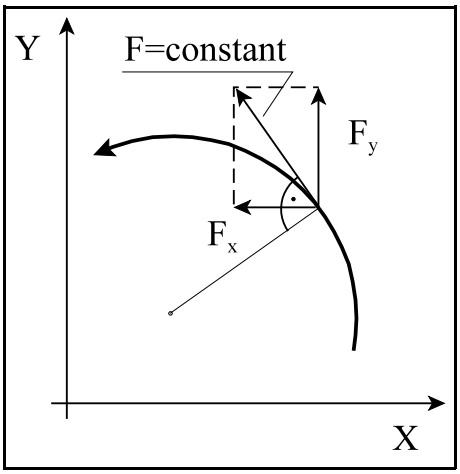


Fig. 4.3-4

An example:
Programming the path shown in the figure.

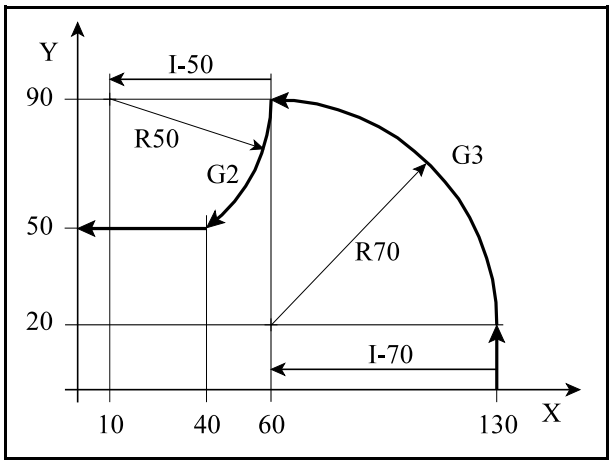


Fig. 4.3-5

Programming a circle using absolute coordinates and specifying R:

```
G90 G17
G0 X130 Y0 M3 S1000
G1 Y20 F500
G3 X60 Y90 R70
G2 X40 Y50 R50
G1 X0
...
```

Programming a circle using absolute coordinates and specifying the circle center I, K

```
G90 G17
G0 X130 Y0 M3 S1000
G1 Y20 F500
G3 X60 Y90 I-70
G2 X40 Y50 I-50
G1 X0
...
```

Programming a circle using incremental coordinates and specifying R:

```
G90 G17 G0 X130 Y0 M3 S1000
G91
G1 Y20 F500
G3 X-70 Y70 R70
G2 X-20 Y-40 R50
G1 X-40
...
```

Programming a circle using incremental coordinates and specifying the circle center I, K:

```
G90 G17 G0 X130 Y0 M3 S1000
G91
G1 Y20 F500
G3 X-70 Y70 I-70
G2 X-20 Y-40 I-50
```

```
G1 X-40
...
```

Specification of ***I0, J0*** and ***K0*** may be omitted. For example:

```
G0 X100 Y0 F500
G17 G03 X0 Y100 I-100
```

In the case of programming a quarter circle with radius of 100 mm and with center in the origin, I0 has not to be written, since the distance of the circle center in the direction X from the point X0 Z100 is 0.

If all three of X_p, Y_p and Z_p are omitted, then

– if coordinates of circle center are specified at the addresses I, J and K, the control will interpolate a full circle with the arc of 360°. For example, in the case of

```
G0 X200 Y0 F500
G17 G03 I-100
```

the control interpolates a full circle with radius 100 mm and with center point of X100 Y0;

– if radius R is specified, for example

```
G0 X200 Y0 F500
G17 G03 R100
```

the control will not move and will not indicate error.

If the circle block contains ***neither radius R nor I, J and K***, the control will send the message '2015 Circle definition error'.

If reference to the addresses ***I, J and K being outside of the selected plane*** is made, the control will send the message '2015 Circle definition error'.

If the ***difference between the starting point radius and the end point radius of the circle*** specified in the G2, G3 block is greater than the value given in the parameter *N1339 Radius Diff*, the control will send the signal *2012 Circle radius difference*.
If the difference between the radii less than the value given in the abovementioned parameter, the control will move the tool along a planar spiral path where the radius changes linearly as a function of the central angle.
In the case of interpolation of an arc with variable radius, not the tangential velocity, but the angular one will be constant.
The value of the parameter N1339 Radius Diff must be greater than 0, for example 0.001, otherwise the control will send unnecessary error indications.

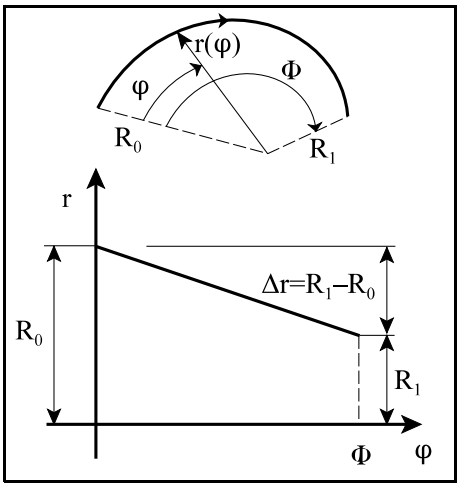


Fig. 4.3-6

Radius difference error or interpolation of a circle of variable radius may be occurred in the following cases:

If the circle center positioning specified at the addresses I, J and K is not correct. For example:

```
G17 G90 G0 X50 Y0
G3 X-20 I-50
```

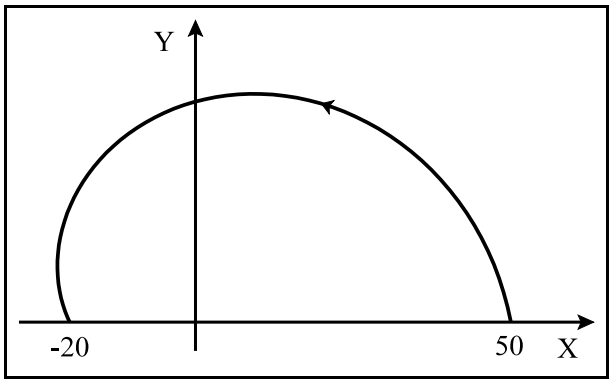


Fig. 4.3-7

If the given circle radius is smaller than the half of the straight line segment between the starting and end points, the control will regard the given circle radius as the starting-point-radius, and it will interpolate such a circle of variable radius, the center of which is on the line linking the starting and end points at a distance R from the starting point:

```
G17 G0 G90 X0 Y0
G2 X40 Y30 R10
```

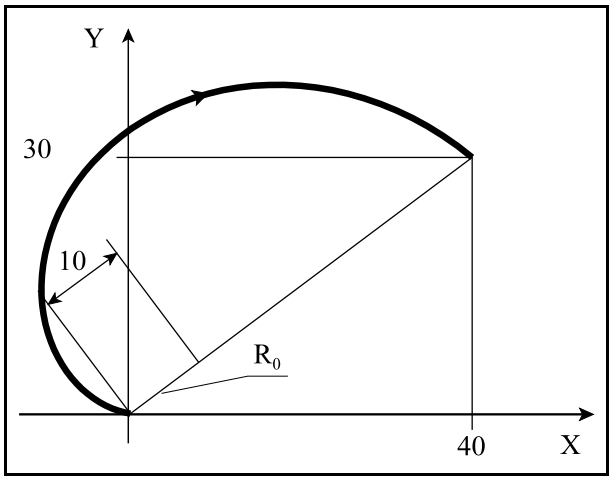


Fig. 4.3-8

The circle center position, i.e, I, J and K can also be given by absolute value calculated from the workpiece zero point. For this, the set position #2 CCA=1 of the parameter N1337 Execution Config is necessary. This case should be avoided.

4.3.1 Planar Spiral Interpolation (G2, G3)

By the instruction G2 or G3, planar spiral interpolation can be programmed in the way of giving the number of revolutions of the spiral too, at the address L. ***The center and the end point of the circle*** have to be specified in such a way ***that the starting-point-radius and the endpoint-radius be different***. Motion will be executed at a feed F programmed in the block or being modal.
The format of the block is:

$$G17\begin{Bmatrix}G2\\G3\end{Bmatrix}X_p\_Y_p\_I\_J\_L\_F\_$$

$$G18\begin{Bmatrix}G2\\G3\end{Bmatrix}X_p\_Z_p\_I\_K\_L\_F\_$$

$$G19\begin{Bmatrix}G2\\G3\end{Bmatrix}Y_p\_Z_p\_J\_K\_L\_F\_$$

The values X_p, Y_p, Z_p are the ***coordinates of the spiral end point*** in the given coordinate system; they are specified as absolute data, or incremental data measured from the starting point.
At the addresses ***I, J and K***, ***the coordinates of the center point of the spiral*** are specified as ***incremental*** data measured from the starting point in such a way that the vector defined by the values I, J and K points ***from the spiral starting point to the spiral center***.
The values ***I, J and K*** will have to always be given ***in radius***, even if the axes adherent to them are set to be programmed in diameter.
The number of revolutions of the spiral is given ***at the address L.*** Every start will mean a new revolution even if the next one is not a full revolution. The address ***L*** is a ***positive integer number***.

In the course of spiral interpolation, the control varies the starting-point-radius (R_0) with the swivel angle (φ) linearly in such a way that the endpoint-radius correspond to the programmed data, when the radius executed revolutions specified at the address L and reached the end point position
It follows from the foregoing that in the course of spiral interpolation what is indicated for the control by filling in the address L, the starting-point-radius differs from the endpoint-radius. If the address L is filled in, the control will never check the maximum value of the radius difference specified in the parameter N1339 Radius Diff. The ***feed F*** specified in the spiral interpolation remains

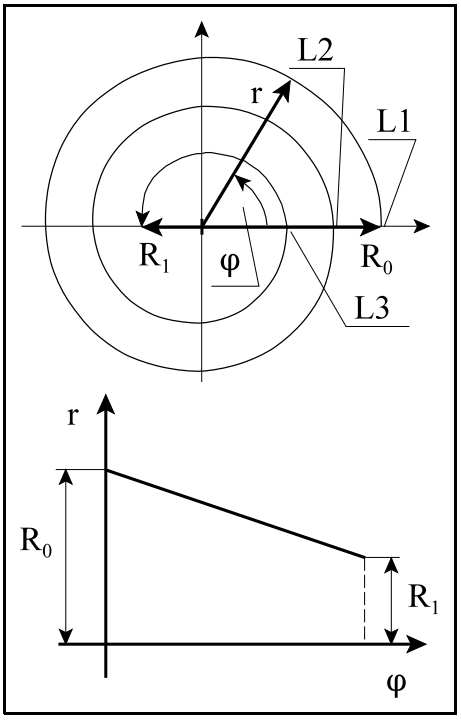


Fig. 4.3.1-1

constant along the full length of the spiral.

An example:
Programming the spiral shown in the figure.
Programming the spiral shown in the figure.
The starting point of the spiral is X180 Y0, its center measured from the starting point is I-90, J0, the radius variation per revolution is 24, and the number of revolutions is 2.5. So, the endpoint radius of the spiral is:

R=90-2*24-24/2=30

The number of revolutions entered is 3.
The program is as follows:

```
G17 G90 G94 M3 S1000
G0 X90 Y0 F1000
G3 X-30 I-90 L3
```

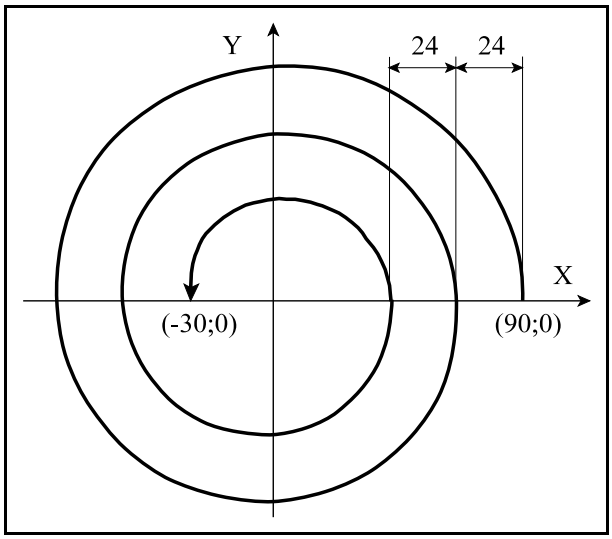


Fig. 4.3.1-2

If we want the control not to indicate error in the loop 2012 Circle radius difference as it is illustrated by the example given in the subsection 4.3 Circular interpolation (page 30), the program will have to be modified in the following way:

```
G17 G90 G0 X50 Y0
G3 X-20 I-50 L1
```

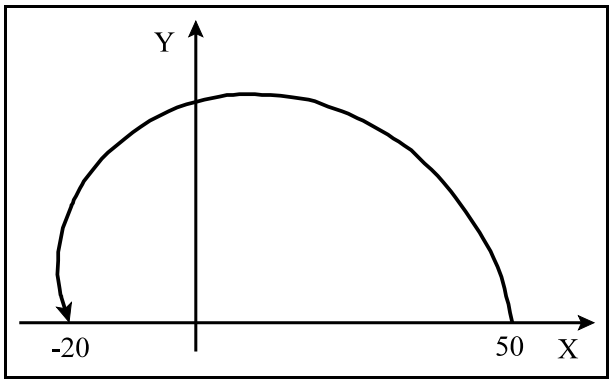


Fig. 4.3.1-3

4.3.2 Helical Spiral Interpolation (G2, G3)

By the instruction G2 or G3, helical spiral interpolation can be programmed in the way of ***programming motion along the axis perpendicular to the plane of the circle***. ***The number of revolution of the spiral*** can be given ***at the address L***. Motion will be executed at a feed F programmed in the block or being modal.
The format of the block is:

$$G17\begin{Bmatrix}G2\\G3\end{Bmatrix}X_p\_Y_p\_\begin{Bmatrix}R\\I\_J\_\end{Bmatrix}Z_p\_q\ldots L\_F\_$$

$$G18\begin{Bmatrix}G2\\G3\end{Bmatrix}X_p\_Z_p\_\begin{Bmatrix}R\_\\I\_K\_\end{Bmatrix}Y_p\_q\ldots L\_F\_$$

$$G19\begin{Bmatrix}G2\\G3\end{Bmatrix}Y_p\_Z_p\_\begin{Bmatrix}R\_\\J\_K\_\end{Bmatrix}X_p\_q\ldots L\_F\_$$

Specification of the circle is carried out according to the rules determined for circle interpolation. Displacement along the axis perpendicular to the plane of the circle arc is proportional to displacement along the circle arc.
In addition to the axis perpendicular to the circle, displacement can be specified for axes of arbitrary quantity, which are marked with q in the formula above and programmable in the channel.
The number of revolutions of the helical spiral is given ***at the address L.*** Every start will mean a new revolution even if the next one is not a full revolution. The address ***L*** is a ***positive integer number***.
In the course of helical spiral interpolation, the control varies the pitch with the swivel angle (φ) linearly in such a way that the endpoint-position on the axes which are outside of the plane of the circle correspond to the programmed data, when revolutions specified at the address L have been executed and the end point position have been reached.

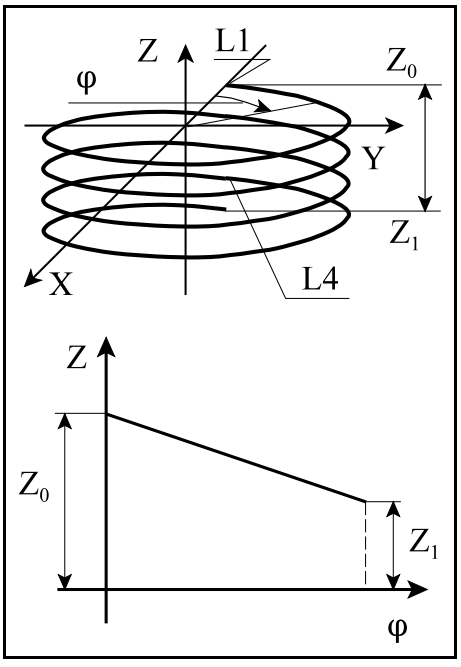


Fig. 4.3.2-1

An example:
The helical spiral interpolation shown in the figure can be specified in the following way:

```
G17 G90 G0 X100 Y0 Z0
G03 X0 Y100 Z20 R100 F150
```

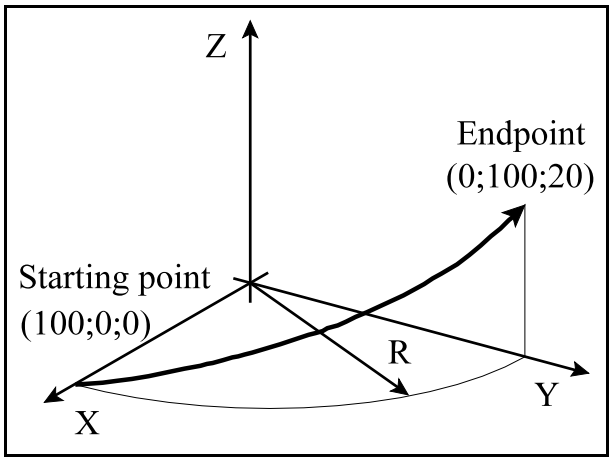


Fig. 4.3.2-2

In this figure, milling of a circle arc into the mantle of an oblique cylinder is illustrated. The axis V is parallel with the axis Y and is moved together with the axis Z by the control:

```
G17 G90 G0 X100 Y0 Z0 V0
G03 X0 Y-100 Z50 V20 I-100
```

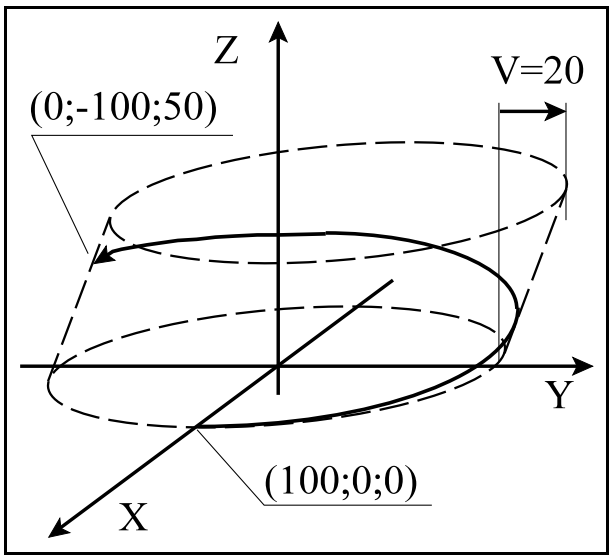


Fig. 4.3.2-3

This picture is for programming a helical spiral with 4 revolutions.
The starting point of the spiral is X-50 Y0 and Z0, its center measured from the starting point is I-50, J0, the pitch per revolution is 5, and the number of full revolutions is 4.
The program is:

```
G54 G17 G90 ...
G0 X-50 Y0
Z0
G2 I50 Z-20 L4 F100
...
```

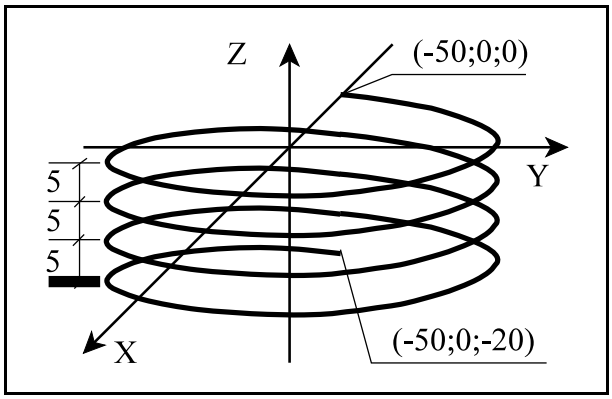


Fig. 4.3.2-4

Basically, the feed given at the address ***F*** is valid ***along circle path*** when the bit position of the parameter N1337 Execution Config #3 is ***HEF=0***. Then, the following formulas give feed for the axis:

$$F_{arc} = F$$

$$F_q = \frac{L_q}{L_{arc}} F$$

In the case of bit position ***HEF=1***, feed will be calculated by the control along spiral path. Then, the following formulas give feed for the axis:

$$F_{arc} = \frac{L_{arc}}{\sqrt{L_{arc}^2 + L_q^2}} F$$

$$F_q = \frac{L_q}{\sqrt{L_{arc}^2 + L_q^2}} F$$

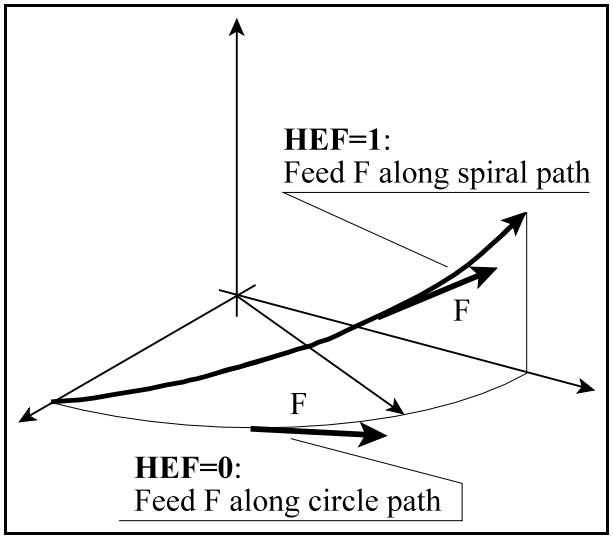


Fig. 4.3.2-5

where L_q: the displacement along the axis q;
L_{arc}: the length of the circle arc;
F: the programmed feed;
F_q: the feed along the axis q;
F_{arc}: the feed along the circle arc.

The specified ***tool radius compensation*** is always valid in the plane of the ***circle*** along the circle path.
In that case, if ***the radius of the circle*** given in the selected plane ***varies***, interpolation will be executed along the mantle of the given cone.

4.3.3 Conical Spiral Interpolation (G2, G3)

By the instruction G2 or G3, conical spiral interpolation can be programmed in the way of ***programming motion along the axis perpendicular to the plane of the circle***. ***The center point and the end point of the circle*** have to be specified in such a way that ***the starting-point-radius*** and ***the endpoint-radius*** of the circle have to be ***different***. ***The number of revolution of the conical spiral*** can be given ***at the address L***. Motion will be executed at a feed F programmed in the block or being modal.
The format of the block is:

$$G17\begin{Bmatrix}G2\\G3\end{Bmatrix}X_p\_Y_p\_I\_J\_Z_p\_q\ldots L\_F\_$$

$$G17\begin{Bmatrix}G2\\G3\end{Bmatrix}X_p\_Y_p\_I\_J\_Z_p\_q\ldots L\_F\_$$

$$G18\begin{Bmatrix}G2\\G3\end{Bmatrix}X_p\_Z_p\_I\_K\_Y_p\_q\ldots L\_F\_$$

$$G19\begin{Bmatrix}G2\\G3\end{Bmatrix}Y_p\_Z_p\_J\_K\_X_p\_q\ldots L\_F\_$$

The values X_p, Y_p, Z_p are the ***coordinates of the spiral end point*** in the given coordinate system; they are specified as absolute data, or incremental data measured from the starting point.
In addition to the axis perpendicular to the circle, displacement can be specified for axes of arbitrary quantity, which are marked with q in the formula above and programmable in the channel.
At the addresses ***I, J and K***, ***the coordinates of the center point of the conical spiral*** are specified in the selected plane as ***incremental*** data measured from the starting point in such a way that the vector defined by the values I, J and K points ***from the spiral starting point to the spiral center***.
The values ***I, J and K*** will have to always be given ***in radius***, even if the axes adherent to them are set to be programmed in diameter.

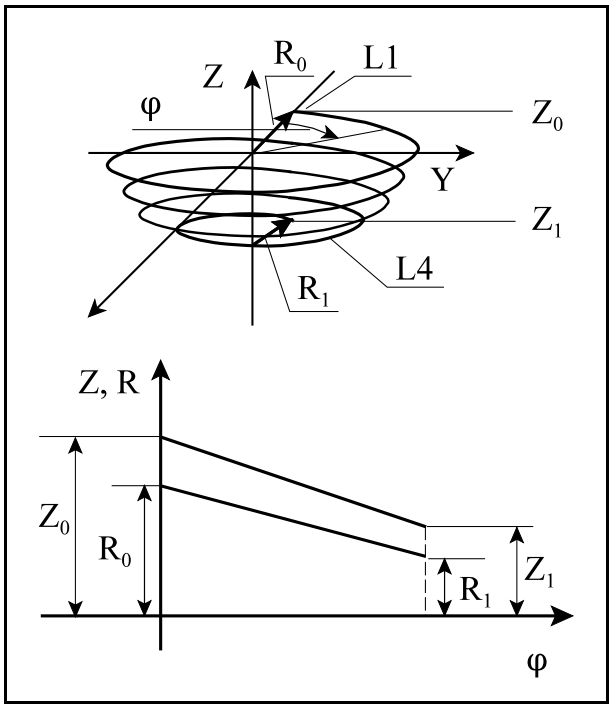


Fig. 4.3.3-1

The number of revolutions of the spiral is given ***at the address L.*** Every start will mean a new revolution even if the next one is not a full revolution. The address ***L*** is a ***positive integer number***.
In the course of conical spiral interpolation, the control varies the pitch with the swivel angle (φ) linearly in such a way that the endpoint-position on the axes which are outside of the plane of the circle corresponds to the programmed data, when revolutions specified at the address L have been executed and the end point position have been reached. It will also vary linearly the circle radius with the swivel angle (φ).

An example:
This picture is for programming a 4-revolution conical spiral whose starting-point-radius is 50, endpoint-radius is 20, and pitch per revolution is 5.
The program is:

```
G17 G90 ...
G0 X-50 Y0
Z0
G2 X-20 I50 Z-20 L4 F100
...
```

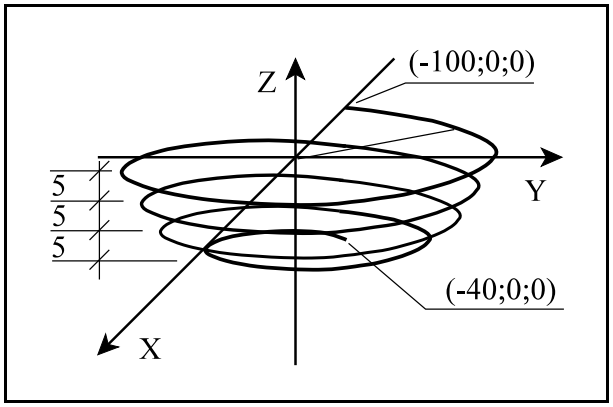


Fig. 4.3.3-2

4.4 Equal Lead Thread Cutting (G33)

By the instruction G33, cutting of straight or tapered thread of equal lead can be programmed. The format of the block is:

G33 v F Q

or

G33 v E Q

For vector ***v***, coordinate data of ***maximum two axes*** can be written. If data of two coordinates are assigned for the vector v, the control will cut tapered thread. Lead will be taken into account by the control on the axis along which the displacement is longer, i.e.
if ***α<45°***, i.e. Z>X, the programmed lead will be taken into account ***along the axis Z***,
if ***α>45°*** , i.e. X>Z, the programmed lead will be taken into account ***along the axis X***.
The lead can be defined in the following two ways:

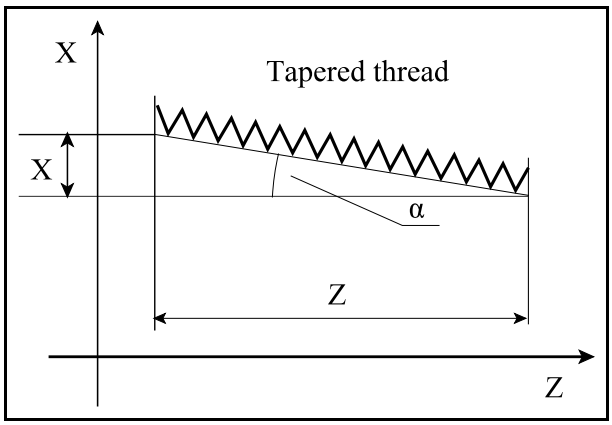


Fig. 4.4-1

Specification of the lead at address F
If the lead is specified at ***address F***, the data will be interpreted in ***mm/rev*** or ***inch/rev.*** Accordingly, F2.5 will has to be programmed if a thread of 2.5 mm lead is to be cut.

Specification of the lead at address E
If the lead is specified at address E, the control will cut inch thread. Interpretation of address E is number of threads per inch. If E8 is programmed, the control will cut a thread, characterized by the lead of 1/8"=25.4/8=3.175 mm.

The ***value of angle*** in degree by which ***the spindle has to rotate from the zero pulse of the spindle encoder*** before starting the thread cutting can be given ***at the address Q***. ***Multiple-start thread*** can be cut by an adequate programming of the value of ***Q***, i.e. the control can be programmed here for the particular angular displacements of the spindle, at which cutting the various starts of the thread are to be started. For example, if a double-start thread is to be cut, the first start will has to be commenced from Q0 (no special programming is needed), while the second start from the Q180.

G33 is a modal function. If several thread-cutting blocks are programmed in succession, it is possible to cut thread on an arbitrary surface bounded by straight line segments.
The control is synchronized to the zero pulse of the spindle encoder in the first block, no synchronization will be performed in the subsequent blocks what results in a continuous lead in each line segment. Therefore, the programmed angular displacement Q of the spindle will also be taken into account in the first block only.

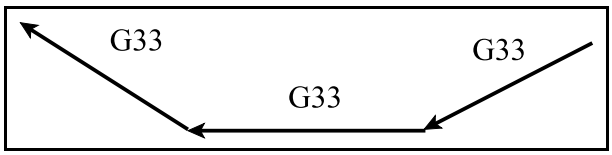


Fig. 4.4-2

An example of programming a thread-cutting:

```
N50 G90 G0 X0 Y0 S100 M4
N55 Z2
N60 G33 Z-100 F2
N65 M19
N70 G0 X5
N75 Z2 M0
N80 X0 M4
N85 G4 P2
N90 G33 Z-100 F2
...
```

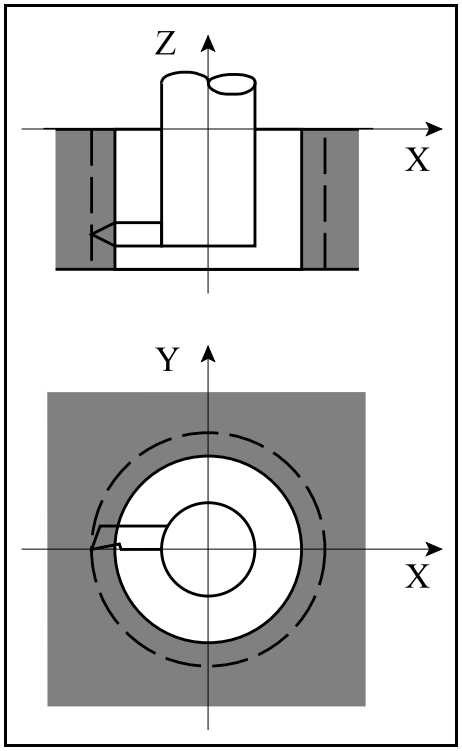


Fig. 4.4-3

Explanation:
N50, N55: Moving the tool over the center of the hole, starting the spindle in counter-clockwise rotation,
N60: First thread-cutting cycle, lead is 2 mm,
N65: Oriented spindle stop (the spindle stops in a fixed position),
N70: Tool retraction along the axis X,
N75: Tool retraction to the top of the hole, programmed stop, the operator adjust the tool to the next thread-cutting cycle,
N80: Return to the center of the hole, re-start of the spindle,
N85: Waiting for the adequate speed to be assumed by the spindle,
N90: Second thread-cutting cycle.

☞ *Notes*:
– If in the thread-cutting block
both addresses F and E are also filled in,
F0 or E0 are specified
the control will indicate the error message ‘2021 Thread programming error’.
– In the course of execution of the instruction G33, the control considers the values of feed and spindle override 100% automatically, and the key Stop is ineffective.
– Because of following error of the servo system, run-in and run-out distances outside the material have to be provided for the tool at the forepart and the end of the thread in order to obtain constant lead all along the full length.
– Before issuing instruction on thread cutting, the state of calculation of constant cutting speed (G96) must be switched off.

4.5 Polar Coordinate Interpolation (G12.1, G13.1)

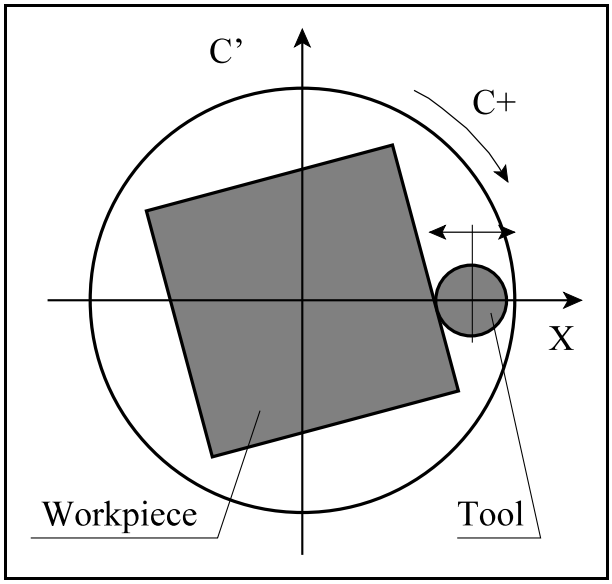


Fig. 4.5-1

The polar coordinate interpolation is a control operation mode, in case of which the control goes along the path of the workpiece contour described in the orthogonal (Descartes) coordinate system moving a ***linear axis*** and a ***rotating axis***, i.e. the path given by orthogonal coordinates is converted by the control into a path represented by polar coordinate data, moment by moment during the motion.

The instruction polar coordinate interpolation on

G12.1

switches the polar coordinate mode on. In the subsequent part of the program, the path of the milling tool ***in orthogonal coordinate system*** can be described by programming a linear axis and a ***rotating axis*** (a ***virtual linear*** axis). *The instruction must be issued in a separate block and no other instruction can be written beside.*

The instruction polar coordinate interpolation off

G13.1

switches the polar coordinate mode off. *The instruction must be issued in a separate block and no other instruction can be written beside*. After switching-on or reset, the control always enters into the state G13.1.

Selection of the Linear and the Rotary Axes

Before switching-on the polar coordinate interpolation, a linear axis and a rotary (virtual) axis have to be selected, these ones will be the axes participating in polar coordinate interpolation. Selection of axes is carried out using plane selection instructions G17, G18 and G19.

G17 X_P_ the address of a rotary (virtual) axis_
G18 Z_P_ the address of a rotary (virtual) axis_
G19 Y_P_ the address of a rotary (virtual) axis_

The first axis of the selected plane will always be ***the linear axis***. Parallel axis can be selected too.

The ***rotary axis*** specified in the plane selection instruction will be the rotary axis of the polar coordinate interpolation. Then, the instruction G12.1 will calculate with data given at the addresses of linear and rotary axes specified in the abovementioned way.

For example, the instruction

G17 X_ C_

designates the axis X as the linear one, and the axis C for the rotary one.

A On the other hand, the instruction

G19 Y_ C_

designates the axis Y for the linear one, and the axis C for the rotary one.

Position of the Workpiece Zero Point in the Course of Polar Coordinate Interpolation

Before switching-on the polar coordinate interpolation, it is necessary to select such workpiece coordinate system, in which ***the center of rotation of the rotary axis coincides with the origin of the linear axis*** of the polar interpolation.
For example, if C is the rotary axis and X is the linear axis, the origin of the coordinate system will has to be chosen on the axis X in such a way that the X=0 position of the tool coincides with the axis of rotation of the rotary axis (C).

If the center of rotation of the rotary axis does not coincide with the first axis of the selected plane (with the linear axis), in other words, if the center of rotation of the rotary axis does not coincide with the axis of rotation of the tool in the direction perpendicular to the linear axis, the machined workpiece will become deformed. The closer the tool moves to the origin the bigger the extent of distortion will be.
Since there is generally no axis in this direction, therefore this deviation cannot be compensated by zero point offset.
This deviation can be compensated in the parameter N0217 Polar Intp. Comp. Amount. The value of compensation must be written ***always at the address of the rotary axis*** in mm or in inch.

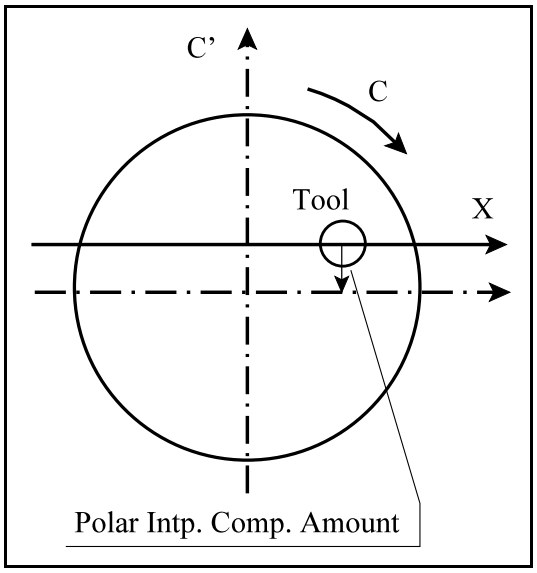


Fig. 4.5-2

Position of the Axes at the Moment of Switching the Polar Coordinate Interpolation on

Before switching the polar coordinate interpolation (the instruction G12.1), it is necessary to ensure ***being of the rotary axis in the point of 0 position***. Position of the linear axis can be either ***negative*** or ***positive***, but ***it cannot be 0***.

Programming the Length Data in the Course of Polar Coordinate Interpolation

In the switched-on state of the polar coordinate interpolation, length coordinate data may be programmed on both axes related to the selected plane: the rotary axis in the selected plane functions as the second (virtual) axis. For example, if the axes X and C have been selected by means of instruction G17 X_ C_ , the address C can be programmed like the axis Y in the case of plane selection G17 X_ Y_.
In polar coordinate interpolation, the function G16 of programming in polar coordinate can also be applied. In this case, obviously, the polar radius is given at the first axis address of the selected plane, and the polar angle is given on the rotary axis.

Basically, it does not influence programming the virtual axis whether the first axis is programmed in diameter or not, the coordinate data on the virtual axis must always be given in radius. For example, if polar coordinate interpolation is executed in the plane XC, the value written at the address C must be given in radius independently of giving the address X in diameter or in radius.

Motion of the Axes Not Participating in Polar Coordinate Interpolation

The tool moves on these axes as it moves in normal case, independently of switched-on state of polar coordinate interpolation.

Programming Circular Interpolation in the Course of Polar Coordinate Interpolation

When polar coordinate interpolation is switched on, a circle can be specified by radius in the way known already, or by programming the circle center coordinates. Having chosen the latter case, the addresses I, J and K have to be used in harmony with the selected plane, according to the following:

G17 X_P_ the address of a rotary (virtual) axis_ I_ J_
G18 Z_P_ the address of a rotary (virtual) axis_ I_ K_
G19 Y_P_ the address of a rotary (virtual) axis_ J_ K_

Use of Tool Radius Compensation in the Case of Polar Coordinate Interpolation

The instructions G41, G42 can be used in the ordinary way when polar coordinate interpolation is switched on. Be careful to it that the tool position code in the rotary tool compensation group be Q=0. The following restrictions must be considered regarding its application:
Switch-on of polar coordinate interpolation (instruction G12.1) is possible in the state G40 only. If G41 or G42 has been switched on in the state G12.1, G40 will have to be programmed before switching the polar coordinate interpolation (the instruction G13.1) off.

Programming Restrictions in the Course of Polar Coordinate Interpolation

The instructions listed below cannot be used when polar coordinate interpolation is switched on:
– plane change: G17, G18, G19;
– coordinate transformations: G52, G92;
– change of workpiece coordinate system: G54, ..., G59;
– positioning in the machine coordinate system: G53;
– G28 reference point return, G30 Pp: 2. 3. 4. return to the reference point;
– G31 measurement with deletion of remaining travel.

Feed in the Course of Polar Coordinate Interpolation

In the switched-on state of the polar coordinate interpolation, feed is interpreted, in the way ordinary in the case of the orthogonal interpolation, as tangential speed: it is the relative speed of the workpiece and the tool.
In the course of polar coordinate interpolation the control goes along a path specified in orthogonal coordinate system moving a linear axis and a rotary axis. As the tool center point approaches the the axis of rotation of the circular coordinate, the rotary axis should take bigger and bigger steps in a unit of time so as the tangential speed will be constant. On the other hand, the speed of the circular axis is limited by the maximum permissible speed of the rotary axis defined by parameter. For this reason, near the origin the control decreases the tangential feed step by step so that the speed of the rotary axis does not increase beyond all limits.
This figure illustrates the case of programming straight lines (1, 2, 3 and 4) parallel with the axis X. Δx displacement in

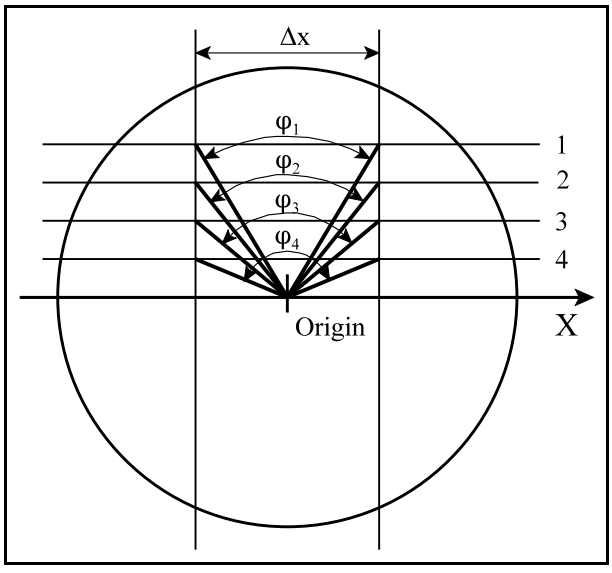


Fig. 4.5-3

unit of time relates to the programmed feed. In the case of different straight lines (1, 2, 3 and 4), different angular displacements (φ_1, φ_2, φ_3, φ_4) relate to the Δx displacement. Apparently, the closer the machining gets to the origin, the larger angular displacement the rotary axis has to make in unit of time in order to keep the programmed feed.
If, in such cases, the angular speed exceeds the value set in the parameter N0305 Max Feed for the rotary axis, the control will decrease the tangential feed step by step.

An example:
This figure shows an example for use of polar coordinate interpolation. The axes participating in the interpolation are the following: axis X (linear axis) and axis C (rotary axis). The ***a x i s X*** i s programmed ***in diameter***, while the ***axis C in radius***.

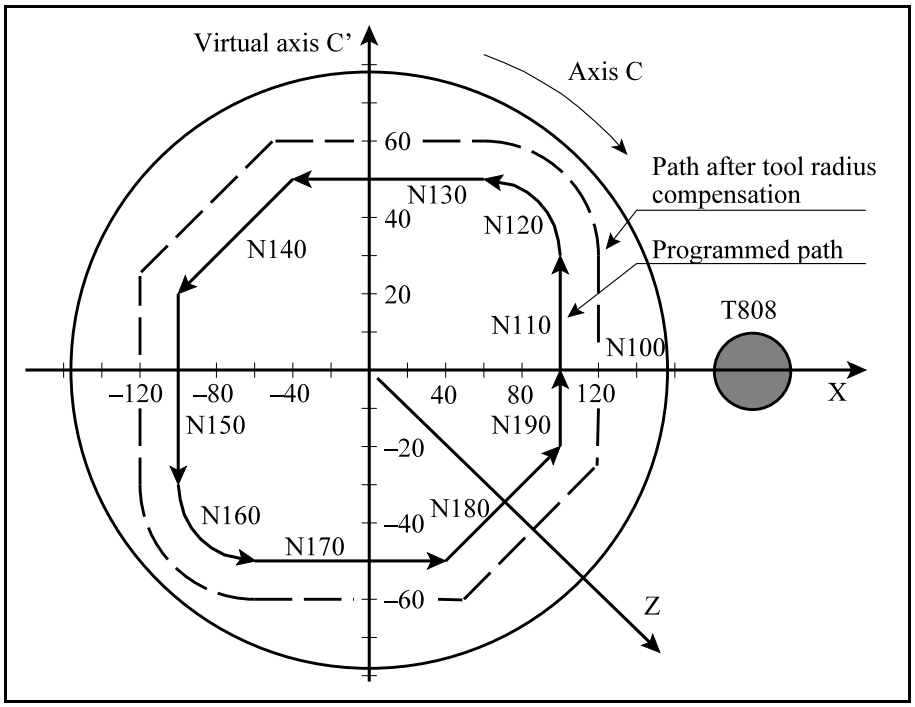


Fig. 4.5-4

```
...
N050 T808
N060 G59                    (The starting point of the coordinate
                            system G59 in the direction X is the
                            axis of rotation of C)
N070 G17 G0 X200 C0         (Selection of the plane X, C;
                            positioning to the coordinate X≠0, C=0)
N080 G94 Z-3 S1000 M3
N090 G12.1                  (Polar coordinate interpolation On)
N100 G42 G1 X100 F1000
N110 C30
N120 G3 X60 C50 I-20 J0
N130 G1 X-40
N140 X-100 C20
N150 C-30
N160 G3 X-60 C-50 R20
N170 G1 X40
N180 X100 C-20
N190 C0
N200 G40 G0 X150
N210 G13.1                  (Polar coordinate interpolation Off)
N220 G0 Z100                (Tool retraction)
...
```

4.6 Cylindrical Interpolation (G7.1)

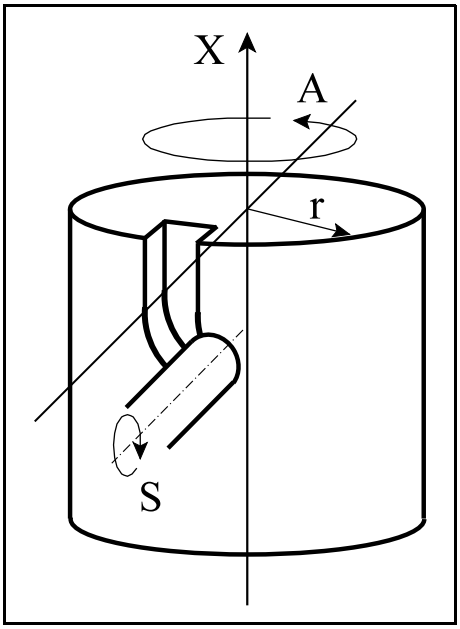


Fig. 4.6-1

If a cam should be milled on the mantle of a cylinder, cylindrical interpolation will be applied. In this case, the center line of the cylinder and the axis of rotation of a rotary axis must coincide.
The program is written by programming ***the linear axis parallel with the center line of the cylinder*** and ***the circular axis rotating the cylinder***. In the program, angular displacement of ***the rotary axis*** is given in ***degree*** which will be converted by the control, along the mantle, as a function of cylinder radius into the arc length measured in mm or inch. The displacement resulted from the interpolation will be reconverted by the control into angular displacement for the rotary axis.
The ***feed F*** specified in the course of cylindrical interpolation is always taken into account ***along the mantle of the cylinder***.
The instruction of cylindrical interpolation On

G7.1 Qr switches the cylindrical interpolation on, where

Q: the address of the rotary axis participating in the cylindrical interpolation;
r: the radius of the cylinder.

For example, if the axis A is the rotary axis participating in the cylindrical interpolation and the cylinder radius is 50 mm, the cylindrical interpolation can be switched on by the instruction G7.1 A50.
In the subsequent part of the program, the path to be milled on the cylinder mantle can be described by specifying linear and circular interpolation. The coordinate for the linear axis must always be given in mm or in inch, while that of the rotary axis in degree (°).
The instruction of cylindrical interpolation Off

G7.1 Q0 switches the cylindrical interpolation off, i.e. the code G is the same as it was in the case of switching on, but at the address of the rotary axis 0 must be written.

The cylindrical interpolation switched on by the instruction G7.1 A50 in the abovementioned example, can be switched off by the instruction G7.1 A0.
After the turn-on, the end of the program or reset, the control will always arrive at the state of cylindrical interpolation Off.
The instruction G7.1 must be specified in a separate block.

Selection of the Linear and the Rotary Axes

Before switching-on the cylindrical interpolation, a linear axis and a rotary axis have to be selected, these ones will be the axes participating in cylindrical interpolation.
Selection of axes is carried out using plane selection instructions G17, G18 and G19.

G17 X_P_, or Y_P_ and the address of a rotary axis_
G18 X_P_, or Z_P_ and the address of a rotary axis_
G19 Y_P_, or Z_P_ and the address of a rotary axis_

The cylindrical interpolation interprets G2 and G3 circular interpolation directions and the direction of tool radius compensation (G41 and G42) on the basis of the plane selected and the linear axis related to the plane. The circular axis will be the other axis of the plane. A parallel axis

can also be selected.
For example, let the axis Z be parallel with the center line of the cylinder, and the axis C be the axis of rotation of the cylinder. In this case, the linear axis and the rotary axis can be assigned using either the instruction

G17 X_ A_

or the instruction

G18 Z_ A_

before switching the cylindrical interpolation on.

Circular Interpolation

It is possible to define circular interpolation in cylindrical interpolation mode, however by specifying the radius R only.
In the case of cylindrical interpolation, circular interpolation is not possible by giving the circle center point (I, J, K).
The circle radius is always interpreted in mm or in inch, never in degree.
For example, circular interpolation between axes Z and C can be specified in two ways:

G17 X_ A_	G18 A_ X_
G2 X_ A_ R_	G3 A_ X_ R_
G3 X_ A_ R_	G2 A_ X_ R_

If the intension is to define the same path by G17 and G18 plane selection, the circle directions will interchange compared to each other.

Application of Tool Radius Compensation in the Case of Cylindrical Interpolation

The instructions G41, G42 can be used in the usual manner in the switched-on state of cylindrical interpolation. The following restrictions are in effect regarding its application:
– Switching the cylindrical interpolation on (instruction G7.1 Qr) is possible in the state G40 only.
– If G41 or G42 has been switched on in the cylindrical interpolation mode, G40 must be programmed before switching the cylindrical interpolation off (instruction G7.1 Q0).

Programming Restrictions in the Course of Cylindrical Interpolation

The following instructions are not available in the switched-on state of cylindrical interpolation:
– plane selection: G17, G18, G19;
– coordinate transformations: G52, G92;
– change of workpiece coordinate system: G54, ..., G59;
– positioning in the machine coordinate system: G53;
– circular interpolation by specifying the circle center point (I, J, K);
– drilling cycles.

An example:
This figure illustrates a path to be milled 3 mm deep on the mantle of a cylinder with radius of R=28.65 mm. The milling tool T is parallel with the axis Z.

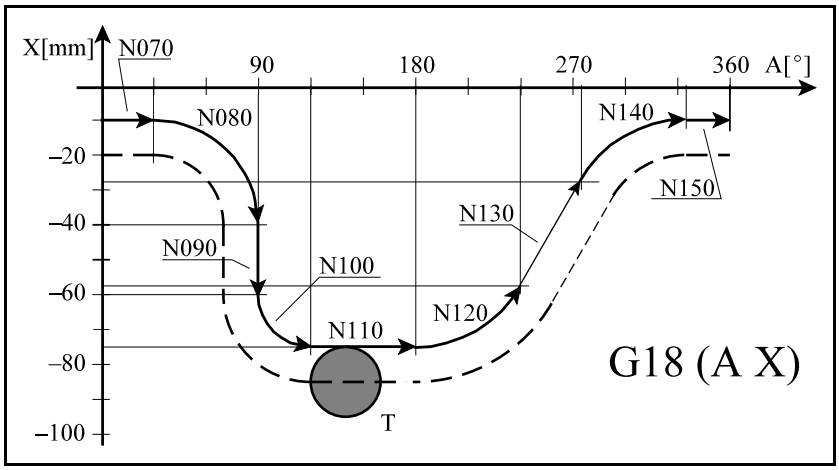


Fig. 4.6-2

On the mantle of the cylinder, the displacement corresponding to the 1 degree (1°) is:

$$28.65mm \cdot \frac{1^\circ}{180^\circ} \cdot \pi = 0.5mm$$

Arrangement of the axes shown in the figure corresponds to the plane selection G18.

```
(CYLINDRICAL INTERPOLATION)
...
N30 G0 Y0
N40 G18 X-20 A0          (G18: selection of the plane A-X)
N40 G1 Z25.65 F500       (3 mm depth of cut)
N50 G7.1 A28.65          (Switching the cylindrical
                         interpolation On, rotary axis: A,
                         cylinder radius: 28.65 mm)
N60 G1 G42 X-10 F500 D1
N70 A30
N80 G2 X-40 A90 R30
N90 G1 X-60
N100 G3 X-75 A120 R15
N110 G1 A180
N120 G3 X-57.5 A240 R35
N130 G1 X-27.5 A275
N140 G2 X-10 A335 R35
N150 G1 A360
N160 G40 X-20 (A380)
N170 G7.1 A0             (Switching the cylindrical
                         interpolation Off)
N180 G0 Z100
...
```

4.7 Smooth Interpolation (G5.1 Q2)

The programmer can choose between ***two machining modes in case of linear interpolation G01:***

Machining Mode 1:

In the case of such parts or such portions of a part, where the accurate form defined by the program is essential, e.g. corners and plane surfaces, the machining is executed exactly in accordance with the instructions specified in the program (the control always moves along linear path).

Machining Mode 2:

In the case of such parts or such portions of a part, where the curved path is approximated by the program using straight line segments and smooth surface is required, the control connects the given points using not line segments but curves.

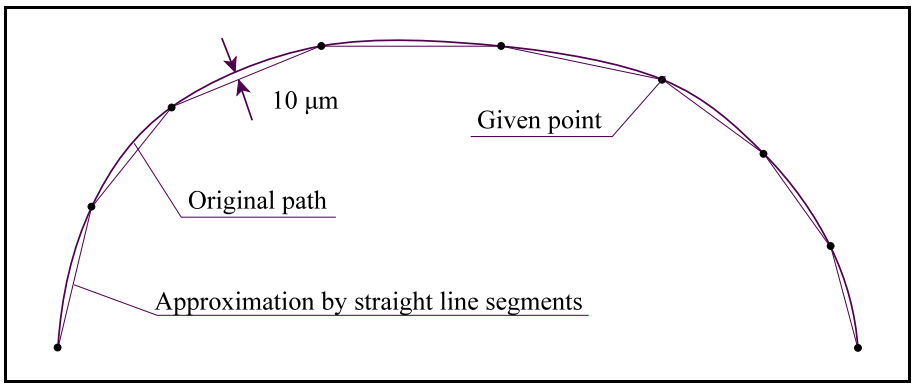


Fig. 4.7-1

In the case of machining complicated surfaces, e.g. in the course of die making, the program usually approximates the path using small straight line segments. The length of the straight line segments is defined by CAD programs so that the difference between the straight line and the curve will not be greater than a given value (e.g. 0.01 mm). When the surface is approximated by straight line segments, the given straight line segment will be short where the radius of curvature is small, and it will be longer where the radius of curvature is large.
When such surfaces are being machined in high-speed high-precision (HSHP) mode, the control attempts to track the course of the path programmed as accurately as possible. As a consequence of this, the surface will not be smooth, deflections between straight line segments will be seen. This is a normal side-effect of the HSHP mode of machining.

The smooth interpolation has been introduced to eliminate the abovementioned effect.

The instruction of smooth interpolation On

G5.1 Q2 X0 Y0 Z0

switches the smooth interpolation on. This instruction switches on the of multiple block preprocessing mode and the high-speed high-precision (HSHP) mode, too.
Smoothing the path between the blocks G1 is executed along the axis specified in the instruction G5.1. Instead of the axes X, Y and Z, axes parallel with them can also be assigned.
In this case, the points specified in the blocks G1will be connected by no straight line but by cubic Bézier curve. It is needed in the case, when the surface has to be ‘smoothed’ in order to eliminate

deflection between chords describing the surface.
Switching the smooth interpolation on increases the speed of program execution too at the same time, because the axes can move at higher speed along a path smoothed.
The instruction of smooth interpolation and HSHP Off

G5.1 Q0

switches the smooth interpolation and the high-speed high-precision machining mode off.
The instruction of smooth interpolation Off

G5.1 Q1

switches the smooth interpolation off, but leaves the high-speed high-precision machining mode switched on.
After the turn-on, the end of the program or reset, the control always ***switches*** the smooth interpolation ***off***.

Specification of Axes Participating in Smoothing
If axis address is not programmed in the block G5.1 Q2, smoothing will be executed along all the linear axes programmed.
The value (for example 0) written at the axis addresses in the instruction G5.1 Q2 does not produce any effect, motions will not be executed on them by the control; the control will specify the only axes participating in smoothing.
If the program creates the surface by moving in the plane XZ for example, and it takes step only along the axis Y, the instruction G5.1 Q2 X0 Z0 has to be written in the program, because smoothing has to be executed in the plane XZ only.
If the program creates the surface by moving along three axes, for example along a 45° straight line in the plane XY, while it moves the axis Z too, the instruction G5.1 Q2 X0 Y0 Z0 has to be written in the program.

Conditions of Switching the Smooth Interpolation on are as follows:

G94: feed per minute

The control executes smooth interpolation if the following cases exist at the same time:
– there are blocks containing linear interpolation G1;
– in the state G40;
– in the continuous cutting state G64;
– in the state G80.

The control cancels smooth interpolation temporarily in the following cases:
– if it executes interpolation codes different from the G1 (e. g. G0, G2, G3, G33).In this case, motion along linear or circular path will be executed according to the code;
– if it executes G43 or G44;
– if it goes to a G or M code cancelling the block preprocessing (for example G53), motion along linear path will be executed in the block preceding that code.

Parameters Influencing the Smooth Interpolation
Existence of the following parameters produces automatic switching on or off the smooth interpolation.

Parameter N3100 Max. Distance of a Block
If the length of a line segment specified by the code G1 is greater than the value given in the parameter N3100 Max. Distance of a Block, the control will cancel smoothing on this segment

and executes linear interpolation.
If longer planes should be milled along a certain distance, this parameter will have to be set in accordance with the length of these planes.

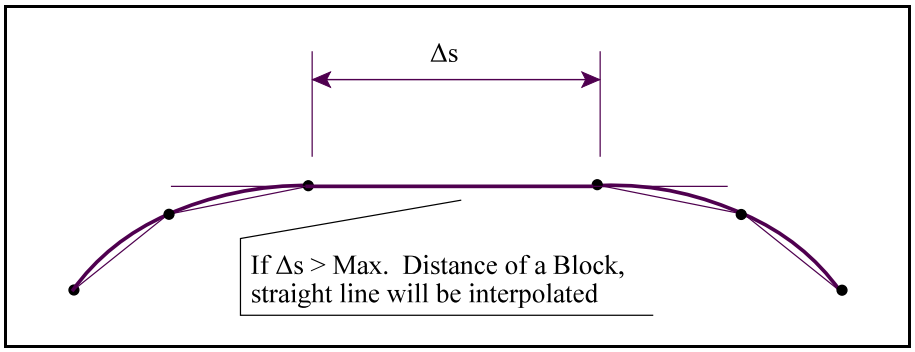


Fig. 4.7-2

Parameter N3101 Min. Distance of a Block
If the length of a line segment specified by the code G1 is smaller than the value given in the parameter N3101 Min. Distance of a Block, the control will cancel smoothing on this segment and executes linear interpolation.
If a minute stair has to be milled along a certain distance, this parameter will have to be set in accordance with the height of this stair.

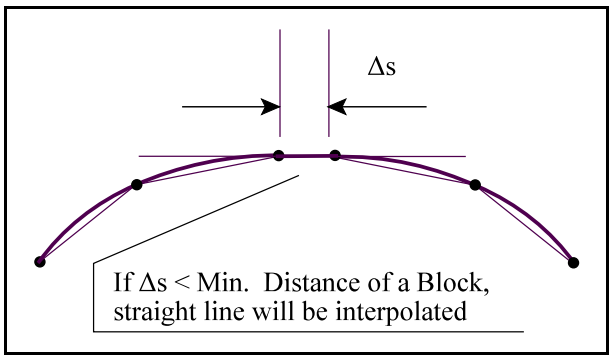


Fig. 4.7-3

Parameter N3102 Angle Cancelling Smooth
If the value of the angle included by two consecutive line segments specified by code G1 is greater than the value set in the parameter N3102 Angle Cancelling Smooth, the control will stop smoothing for these two line segments.
When a sharp deflection line of a given angle should be machined on the workpiece surface, this parameter can be used for setting the stop of smoothing and for machining the deflection line.

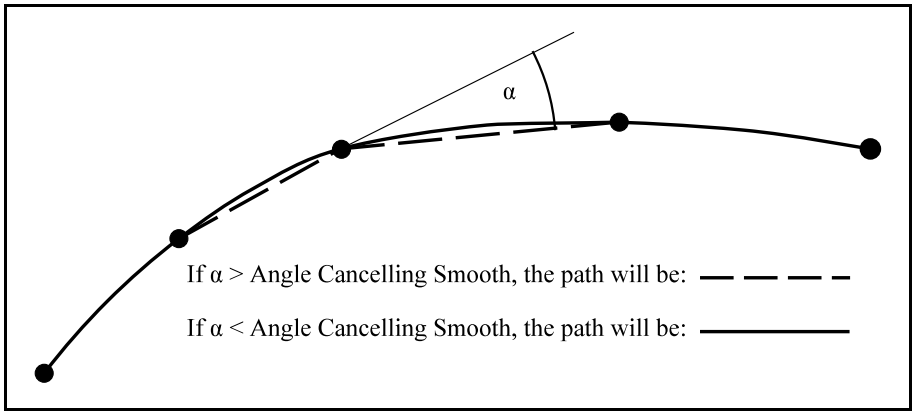


Fig. 4.7-4

Parameter N3103 Accuracy

It is used for setting the accuracy level when the high-speed and high-precision (HSHP) path tracking is switched on.

The programmed short linear displacements (blocks G01) per axis will be merged until the magnitude (absolute value) of displacement along any axis is greater than the value set in the parameter, and then the displacements merged in such a way will be issued in one.

If the value of the parameter is 0.01, the smallest displacement the control issues to the machine will be 0.01 mm.

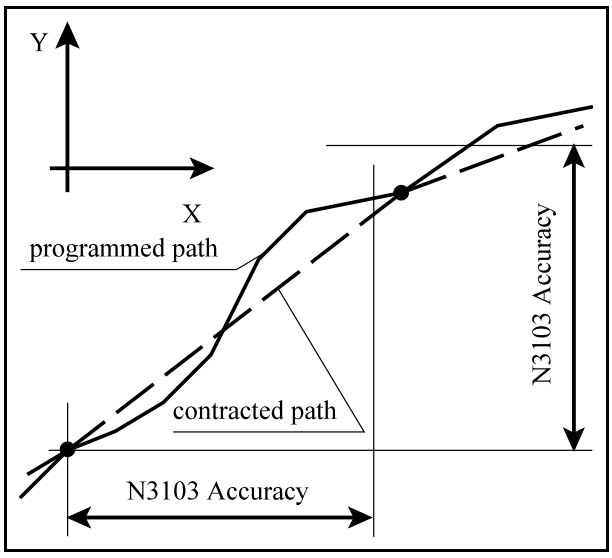


Fig. 4.7-5

Let us see the following program:

```
G90 G1
...
N2500 X25.432 Y47.847
N2510 X25.434 Y47.843
N2520 X25.437 Y47.839
N2530 X25.439 Y47.835
N2540 X25.440 Y47.831
N2550 X25.442 Y47.827
```

In the example above, the displacement along the axis Y regarding the blocks N2500 and N2530 is 47.835−47.847=−0.012, and that is already greater than the value set in the parameter. For this reason, the control moves in a way as if rows N2510 and N2520 would have been deleted from the program:

```
G90 G1
...
N2500 X25.432 Y47.847
N2530 X25.439 Y47.835
N2540 X25.440 Y47.831
N2550 X25.442 Y47.827
```

The value specified here also affects feedrate, since the smaller the value specified, the smaller the feedrate with which the tool moves on the short sections.

In per-block mode, the control will stop always after block merge.

Parameter N3104 Max. Tolerance of Smooth
The number written in this parameter tells the maximum permissible deviation of the smoothed path from the straight line in the course of smoothing the line segment programmed. If the tolerance is greater than the given value, the control will not smooth along the given segment, but it will interpolate straight line.
It is practical to write in this parameter the accuracy value that was given in CAM program (e.g. 0.01 mm) during generation of the part program.

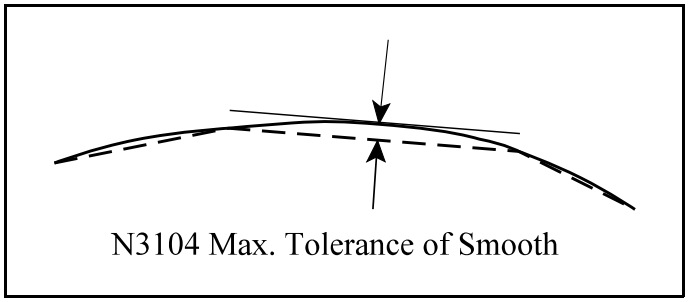


Fig. 4.7-6

Parameter N3105 Min. Tolerance of Smooth
The number written in this parameter tells the minimum permissible deviation of the smoothed path from the straight line in the course of smoothing the line segment programmed. If the tolerance is smaller than the given value, the control will not smooth along the given segment, but it will interpolate straight line.

Note:

When the abovementioned parameters are set properly, not only could the smooth interpolation improve the surface quality, but it will also influence the speed of machining.
Restrictive parameter setting decreases the speed of machining, i.e. lengthens the time per piece, while parameter setting that permits smoothing to a higher degree increases the speed of machining, i.e. reduces the time per piece.

4.7.1 Fine Smoothing (G5.1 Q3)

The difference between fine smoothing and normal smooth interpolation is as follows: while in the case of normal smoothing the path strictly passes through the programmed points, ***in the case of fine smoothing the control moves the programmed points within a given limit*** in order that to be the motion and the path as smooth as possible and the run time is as short as possible.

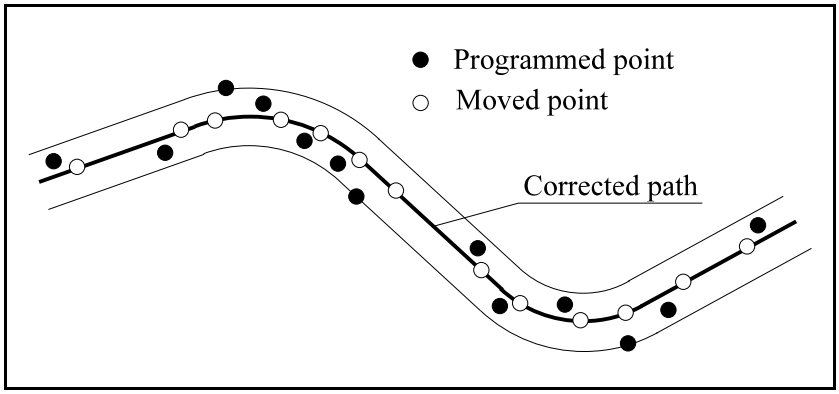


Fig. 4.7.1-1

The instruction

G5.1 Q3 X0 Y0 Z0

switches the fine smoothing on. This instruction switches on the of multiple block preprocessing mode and the high-speed high-precision (HSHP) mode, too

The instruction

G5.1 Q0

switches the fine smoothing and the high-speed high-precision mode off.

The instruction

G5.1 Q1

switches the smoothing interpolation off, but leaves the high-speed high-precision machining mode switched on.

After the turn-on, the end of the program or reset, the control always switches the fine smoothing off.

In the case of fine smoothing, the same rules apply to

the axis assignement,
conditions of its switch-on
its suspension
as those in the case of normal smooth interpolation

While fine smoothing is switched on, it is prohibited to switch the length compensation off or on. Otherwise, the control will send the message '2188 G<code> not allowed in Fine smooth mode'.

Correct

```
...
G43 X Y Z B C
G5.1 Q3 X0 Y0 Z0 B0 C0 (fine
smoothing on)
G1 X Y Z B C
....
....
G5.1 Q0 (fine smoothing off)
G49
...
```

Wrong

```
...
G5.1 Q3 X0 Y0 Z0 B0 C0 (fine
smoothing on)
G43 X Y Z B C
G1 X Y Z B C
....
....
G49
G5.1 Q0 (fine smoothing off)
...
```

In the course of fine smoothing, in per-block mode, the control will stop always at the corrected point, and not at the programmed one.

The parameters

N3100 Max. Distance of a Block
N3101 Min. Distance of a Block
N3102 Angle Cancelling Smooth
N3103 Accuracy

are also apply to the fine smoothing (G5.1 Q3) in accordance with the rules for the smooth interpolation (G5.1 Q2) described in the previous chapter.
For fine smoothing, the following new parameters have been introduced.

Parameter N3106 Tolerance of Smooth2
In the case of Fine smoothing (G5.1 Q3), after moving the programmed end-points of the blocks, the modified path remains within the same accuracy.

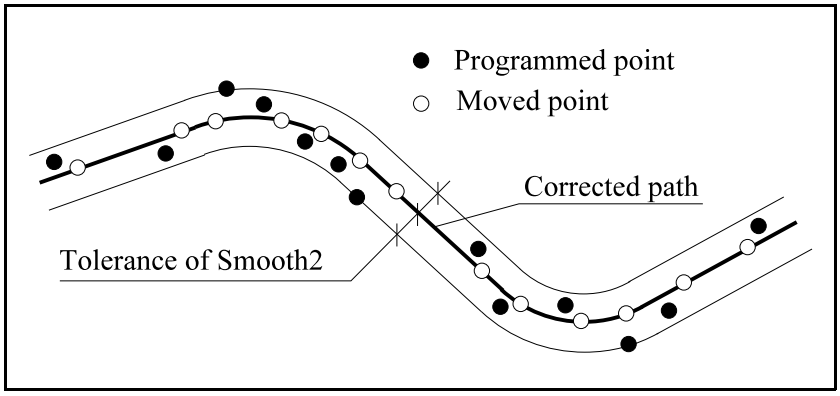


Fig. 4.7.1-2

Note: In the case of the fine smoothing, instead of the parameters N3104 Max. Tolerance of Smooth és az N3105 Min. Tolerance of Smooth, the parameter mentioned above is used by the control.

Parameter N3107 Dist. Cancelling of Angular Calc. in Smooth2
If, because of any computing inaccuracy, the CAM inserts a short section between the points, which causes big change in direction along the path, the direction change can be eliminated by the use of this parameter.
In the course of fine smoothing (G5.1 Q3), the control smooths shorter section than this in any case regardless of the value of the parameter N3102 Angle Cancelling Smooth, i.e. it eliminates the change in path direction. The specified value should not be greater than twice the value of the parameter N3106 Tolerance of Smooth2!
This parameter only applies if its value is greater than the value of the block merging parameter N3103 Accuracy.

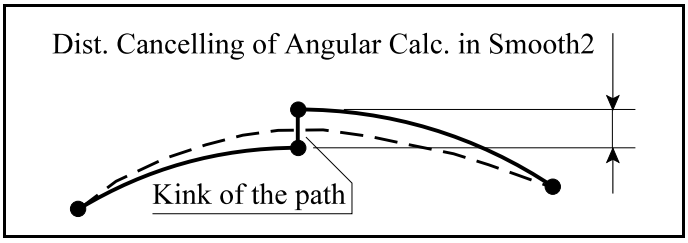


Fig. 4.7.1-3

The fine smoothing G5.1 Q3 can also be used together with the tool centre point control in the course of 5D machining. Description can be found on the page 435 of the chapter '25.18 Smooth Interpolation in the Case of Tool Centre Point Control'.

5 Coordinate Data

5.1 Absolute and Incremental Programming (G90, G91)

The input coordinate data can be specified as absolute and incremental values too. In the case of absolute data specification, the coordinates of the end point must be given to the control, while in the case of incremental data it is the distance to be travelled in the block that must be programmed.

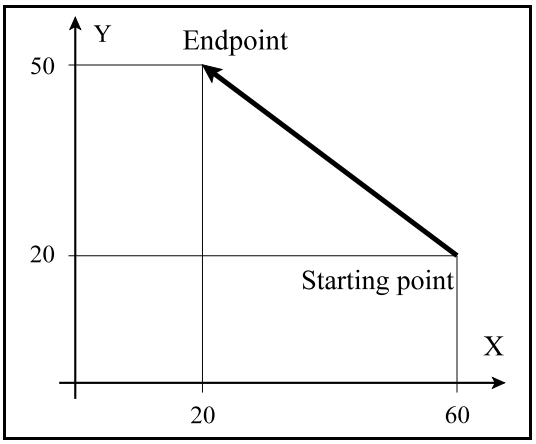
G90: Programming the absolute data specification

G91: Programming the incremental data specification

The G90 and G91 are modal functions. At the time of power-on, the control, according to the bit #7 G91 of the parameter N1300 DefaultG1, decides which of the two states will apply. At the end of the program or in case of reset, the code set in this parameter is effective too.
Motion to an absolute position is feasible after a reference point return only.

An example:
On the basis of this figure, the motion can be programmed in the following two ways:

```
G90 G01 X20 Y50
G91 G01 X-40 Y30
```



5.1-1

The ***operator I*** is effective in the state G90 of absolute data specification. ***It applies to the coordinate only after the address of which it stands***. Its meaning is: incremental data.
Another way to solve the example above is:

```
G90 G01 XI-40 YI30
G01 X20 YI30
G01 XI-40 Y50
```

In the case of using ***multicharacter axis name***, if the name ***ends with number***, e.g. axis X2, the operator I will have to be written after the sign=:

```
X2=I100
```

If the ***addresses U, V and W*** are not assigned for axes, they can be used to indicate incremental motions in the directions X, Y and Z:

	Address of absolute instruction	Address of incremental instruction
Motion instruction in the direction X	X	U
Motion instruction in the direction Y	Y	V
Motion instruction in the direction Z	Z	W

Taking the abovementioned into account, the solution of the example is:

```
G90 G01 U-40 V30
G01 X20 V30
G01 U-40 Y50
```

5.2 Inch/Metric Conversion (G20, G21)

By programming appropriate code G, the input data can be specified either in inch unit system or in metric unit system.

G20: Choosing the inch unit system

G21: Choosing the metric unit system

The unit system chosen will be in effect until a instruction of contrary meaning is issued, therefore the codes G20 and G21are modal ones.

At the time of power-on, the control, according to the bit #3 G20 of the parameter N1300 DefaultG1, decides which of the two states will apply. At the end of the program or in case of reset, the code set in this parameter is effective too.

For example:

```
G21 G0 G54 X20 Y10 Z50 (positioning to X=20 mm, Y=10 mm,
Z=50 mm)
G20 X2 Y3 Z1 (positioning to X=2 inch, Y=3 inch, Z=1 inch)
```

Changing the unit system influences the following items:

– Coordinate and compensation data, (mm/inch);
– Feed rate (mm/min, inch/min, mm/ford, inch/ford);
– Constant cutting speed (m/min, feet/min);
– The values of position, compensation, zero point offset and feed rate are always displayed in the unit system that has been chosen.
– When macro variables (offset, position data etc.) are being read, data will be read out in the unit system chosen.
– Incremental jog and handwheel motion step;
– Feed rate of the manual move (jog).

5.3 Programming in Diameter or Radius

In the case of using lathe functions in the milling channel, the cross-section of the workpiece machined are usually a circle and the workpiece can be controlled by diameter measuring, therefore it could be practical to specify the dimensions on certain axes in diameter. For each axis, it can be specified in the bit #0 DIA of the parameter N0106 Axis Properties whether the control interprets the dimension in diameter or in radius:

in the case of **programming in radius**:

#0 DIA=0

in the case of **programming in diameter**:

#0 DIA=1

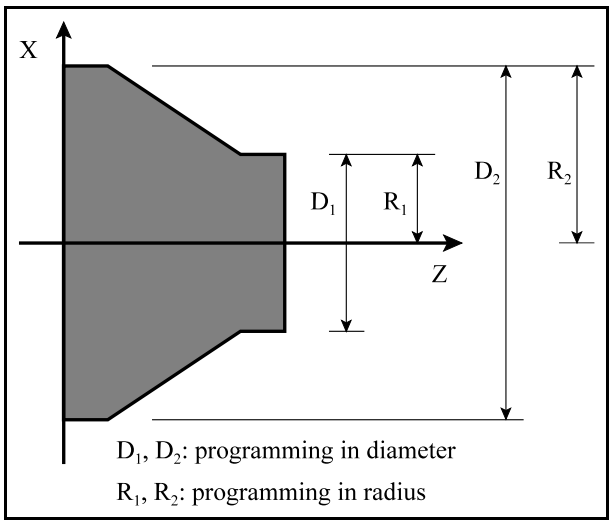


Fig. 5.3-1

If the parameter is set for programming in diameter, the following will have to be taken into account:

Case	Note
Absolute motion instruction	Specified with diameter value
Incremental motion instruction	Specified with diameter value (on the basis of the figure $D_2 - D_1$)
Coordinate offset and zero point offset	Specified with diameter value
Tool length compensation	Specified with diameter value
In canned cycles, parameters concerning axes managed in diameter, such as cutting depth	Always specified with radius value
The value of R and I, J and K in case of specifying circular interpolation	Always specified with radius value
Displaying the axis position	As diameter value
Feed rate in direction of the axis managed in diameter	Always in radius/rev or radius/min
Step value in jog or handwheel mode	Step of 1μm will be taken in diameter if one increment is chosen

5.3.1 Switching between in Radius and in Diameter Programming (G10.9)

It is specified by the value of

the bit #0 DIA of the parameter N0106 Axis Properties

whether the programming on a given axis will be done in radius or in diameter. If the value of the bit

=0: the axis will be programmed in radius;

=1: the axis will be programmed in diameter.

The mode of programming can be switched anywhere in the part program. It is specified by the value of

the bit #5 MGD of the parameter N0106 Axis Properties

whether the switching will be executed from PLC or by code G. If the value of the bit

=0: the switching will be executed through PLC signal;

=1: the switching will be executed by the code G10.9.

The manner of the switching from PLC is defined by the machine builder.

Hereunder, the switching by the code G10.9 will be described.

The instruction

G10.9 v

switches between programming in radius and programming in diameter, where

v: address of arbitrary linear axes, e.g. X, Y ... etc.

If the value written at the axis address:

=0: the axis will be programmed in radius;

=1: the axis will be programmed in diameter.

For example, in basic case, the axes X and Y on a milling machine are programmed in radius.

If, on a given section of the part program, it is practical to specify the value of X and Y in diameter, the following will have to be written in the part program:

```
... (X, Y are specified in radius)
...
G10.9 X1 Y1 (programming the diameter)
... (X, Y are specified in diameter)
...
G10.9 X0 Y0(programming the radius)
... (X, Y are specified in radius)
...
```

If, on a lathe, the axis is programmed in diameter, but in the case of milling to be performed using polar interpolation and axes X and C, the intension is to program the X in radius, then:

```
... (X is specified in diameter, turning)
...
G10.9 X0 (programming the radius)
G12.1 (polar interpolation On)
... (X, is specified in radius, milling)
...
G13.1 (polar interpolation Off)
G10.9 X1 programming the diameter)
... (X is specified in diameter, turning)
...
```

The code G10.9 must be given always in a separate block!

When the program ends or reset is done, the programming mode set by the code G10.9 will be deleted, and the control returns to the basic state set in the bit #0 DIA of the parameter N0106 Axis Properties.

5.4 Data Specification using Polar Coordinates (G15, G16)

The values of the programmed coordinates can also be input as radius given in mm or in inch, and angle given in degree.

G16: Data specification using polar coordinates on

G15: Data specification using polar coordinates off

G15 and G16 are modal functions. After power-on, at the end of the program or in case of reset, the control turns to the state G15.

Data specified in polar coordinates are interpreted in the plane defined by the codes G17, G18 and G19. In the course of data specification, the control considers the address of the first axis of the plane to be the radius, and the second axis to be the angle.

G17 X_P (radius) Y_P (angle)
G18 Z_P (radius) X_P (angle)
G19 Y_P (radius) Z_P (angle)

X_P, Y_P, Z_P could be the basic axes X, Y, Z or axes parallel with them.

In the case of angle specification, ***counter-clockwise*** is the ***positive*** direction of the angle and ***clockwise*** is the ***negative***

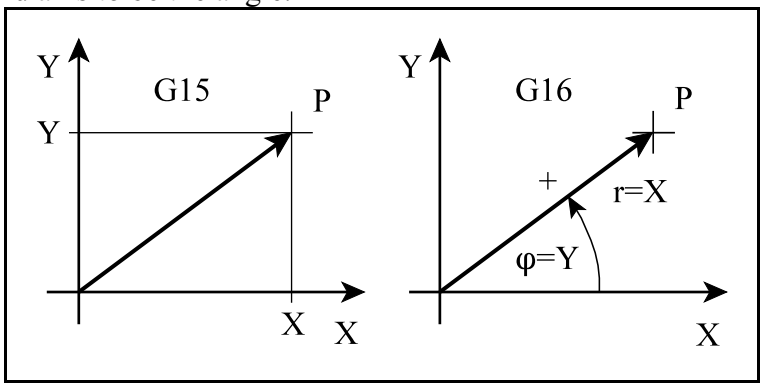


Fig. 5.4-1

direction of the angle.
The control considers data of other axes being outside of the selected plane to be data specified in Cartesian coordinates.

The radius and the angle can be specified as absolute data and incremental data as well.

When ***the radius is specified as absolute data***, the origin of the actual coordinate system will be the origin of the polar coordinate system:

```
G90 G16 X100 Y60
```

Both the angle and the radius are absolute data, the tool moves to the point of 100 mm radius and of 60° angle.

```
G90 G16 X100 YI40
```

The angle is an incremental data. The tool moves to the point of 100 mm radius being further on by 40° compared to the previous angle position.

When ***the radius is specified as incremental data***, the control moves the given radius from the axis positions specified at the beginning of the block in the direction of the angle given:

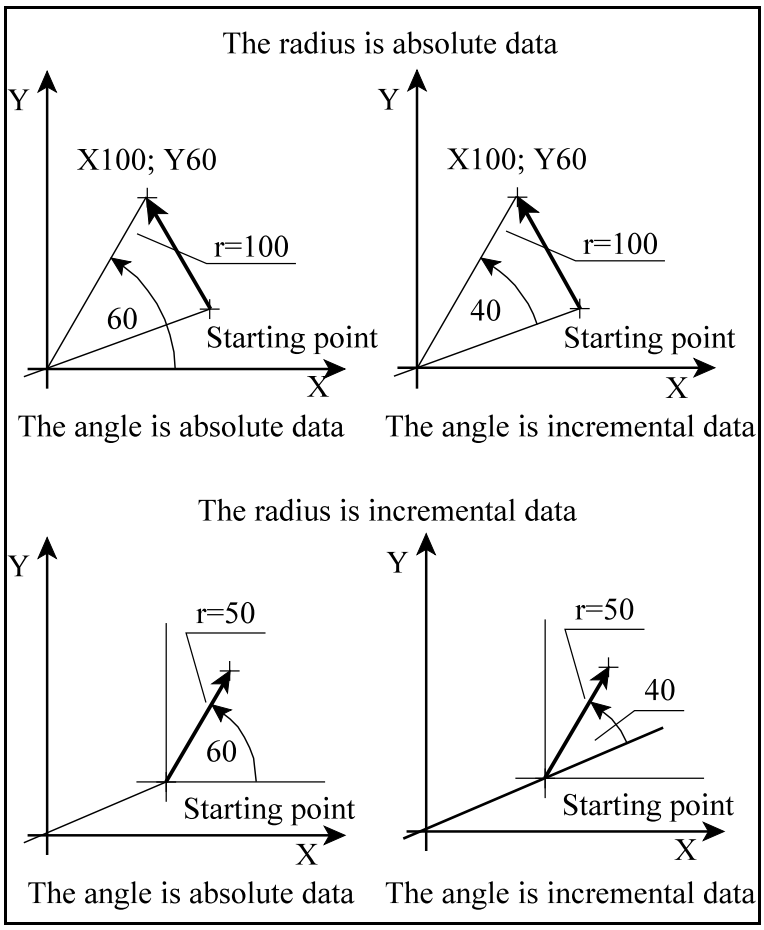


Fig. 5.4-2

```
G90 G16 XI50 Y60
```

The control measures the radius of 50 mm from the starting point and moves to the absolute angle of 60°.

```
G91 G16 X50 Y40
```

The control measures the radius of 50 mm from the starting point and moves to the angle of 40° measured from the starting angle.

Programming a circle is also possible when the data specification in polar coordinates G16 is on. The circle can be specified both by radius and by I, J and K. But in the latter case, the addresses ***I, J and K*** will ***always*** be considered by the control to be ***Cartesian data***.
When the origin of the actual coordinate system coincides with the center of the circle, multi-revolution circle or spiral can also be programmed using data specification in polar coordinates:

```
G17 G16 G90 G02 X100 Y-990 Z50 R-100
```

In the block above, a clockwise spiral of 2 3/4 revolutions is specified. When a multi-revolution

circle is to be programmed, attention has to be paid on programming negative polar angle in case of direction G2, and on programming positive polar angle in case of direction G3.
The addresses in the following instructions are not considered by the control to be polar coordinate data even in the case when the state G16 is switched on:
– G10 coordinates in setup instruction;
– G52 coordinate offset;
– G92 coordinate setup;
– G28, G30 coordinates of an intermediate point;
– G53 positioning in the machine coordinate system;
– G68 the central point of rotation of the coordinate system;
– G51 the central point of scaling;
– G50.1 the central point of reflection.

An example: Milling a hexagon

```
N1 G90 G17 G0 X120 Y120 F120
N2 G16 G1 X100 Y60
N3 Y120
N4 Y180
N5 Y240
N6 Y300
N7 Y0
N8 Y60
N9 G15 G0 X120 Y120
```

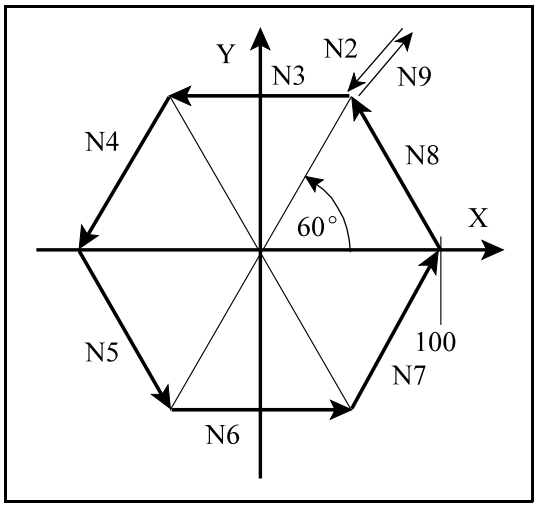


Fig. 5.4-3

5.5 Specification and Accuracy of Coordinate Data

The control interprets the decimal point according to the unit system applied:
– X2.134 means 2.134 mm or 2.134 inch;
– B24.236 means 24.236 degrees, when angle data is specified at the address B.
Use of the decimal point is not mandatory:
– X325 means 325 mm, for example.
The leading zeros can be eliminated:
– .032=0.032
The following zeros after the decimal point can be eliminated:
– 0.320=.32
Coordinate data can be specified with accuracy of up to 15 decimal digit.

5.6 Managing the Rotary Axis Roll-Over

This function can be used in case of axes that rotate, i.e. when an axis address (for eample C) is assigned to a rotary axis.
Managing the roll-over means that the position on the given axis is registered by the control not in between plus and minus infinities, but in between 0° and 360° for example, taking the axis periodicity into account.

Designation of an Axis to be Rotary Axis and the Effect Produced

This designation has to be done by bit setting #1 ROT=1 in the parameter N0106 Axis Properties . For rotary axis,

– the control ***does not do the inch/metric conversion***;
– **the roll-over management can be enabled**.

Enabling the Roll-over Management

This function is activated by bit setting #0 REN=1 in the parameter N0107 RollOver Control . If the value of the bit REN:

=0: the control will manage the rotary axis as it manages the linear axes, and filling the further parameters will not produce any effect;
=1: the control will apply the roll-over management for the rotary axis, and the essence of this is as follows.

Specification of the Distance Travelled per Revolution

The distance travelled per one revolution of the axis can be specified in the parameter N0108 RollAmount. Thus, if the axis rotates 360° per one revolution, the value to be written in the appropriate parameter will have to be 360. The system can manage arbitrary periodicity not only that of 360°.
Using the abovementioned parameter settings, the control displays the position of the rotary axis always in the range of 0°- +359.999° independently of the direction of rotation and the number of revolutions the rotary axis performed.

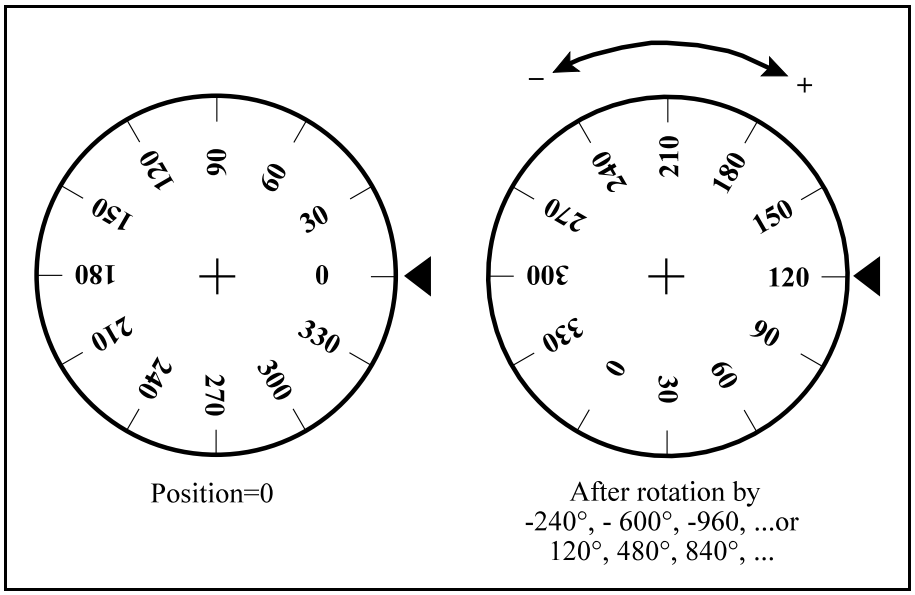


Fig. 5.6-1

In the setting opportunities listed below, the bit state #0 REN=1 of the parameter N0107 RollOver Control and setting of the parameter N0108 RollAmount is supposed.

Rotation to the Specified Position: the Basic Case of Rotation

If the settings of the parameter N0107 RollOver Control are

#2 ABS=0 and #1 ASH=0,

in the case of absolute programming (G90) the rotation will be performed to the specified position. The value of displacement will always be less than the value specified in the parameter RollAmount, i.e. during motion the control always cuts the full revolutions.

An example:

Let the starting position of the axis C be

C=0

In accordance with the instructions

```
G90 C210
G90 C570
G90 C930
...
```

it will always rotate 210 degree (ΔC=210°) in the positive direction, having cut the full revolutions, and the end position will be 201°.

Let the starting position of the axis C be

C=0

In accordance with the instructions

```
G90 C-210
G90 C-570
G90 C-930
...
```

it will always rotate 210 degree (ΔC= –210°) in the negative direction, having cut the full revolutions, and the end position will be 150°(= –201°).

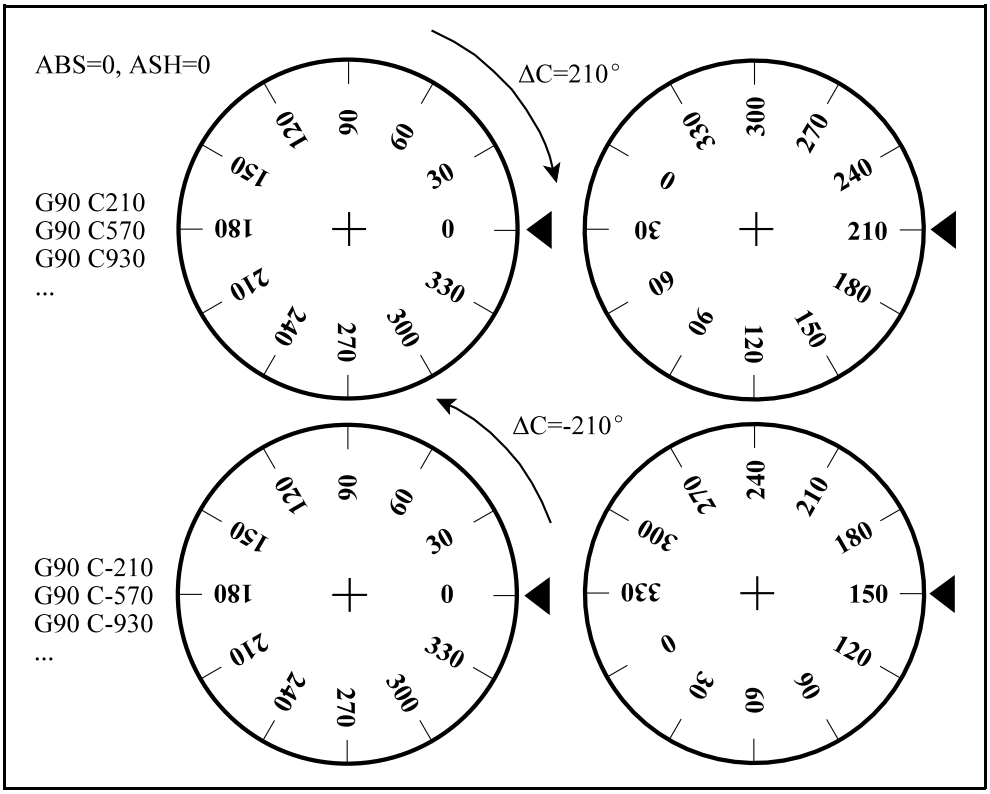


Fig. 5.6-2

Rotation to the Specified Position in the Shorter Way

If the settings of the parameter N0107 RollOver Control are

#2 ABS=0 and #1 ASH=1,

in the case of absolute programming (G90) the rotation will be performed to the specified position and in the direction related to the shorter way. The value of displacement will always be less than the value specified in the parameter RollAmount, i.e. during motion the control always cuts the full revolutions.

An example:

Let the starting position of the axis C be

C=0

In accordance with the instructions

```
G90 C210
G90 C570
G90 C930
...
```

Let the starting position of the axis C be

C=0

In accordance with the instructions

```
G90 C210
G90 C570
G90 C930
...
```

it will always rotate 150 degree (ΔC= – 150°) in the negative direction (in the shorter way), having cut the full revolutions, and the end position will be 210°.

Let the starting position of the axis C be

C=0

In accordance with the instructions

```
G90 C-210
G90 C-570
G90 C-930
...
```

it will always rotate 150 degree (ΔC=150°) in the positive direction (in the shorter way), having cut the full revolutions, and the end position will be 150°(= –210°).

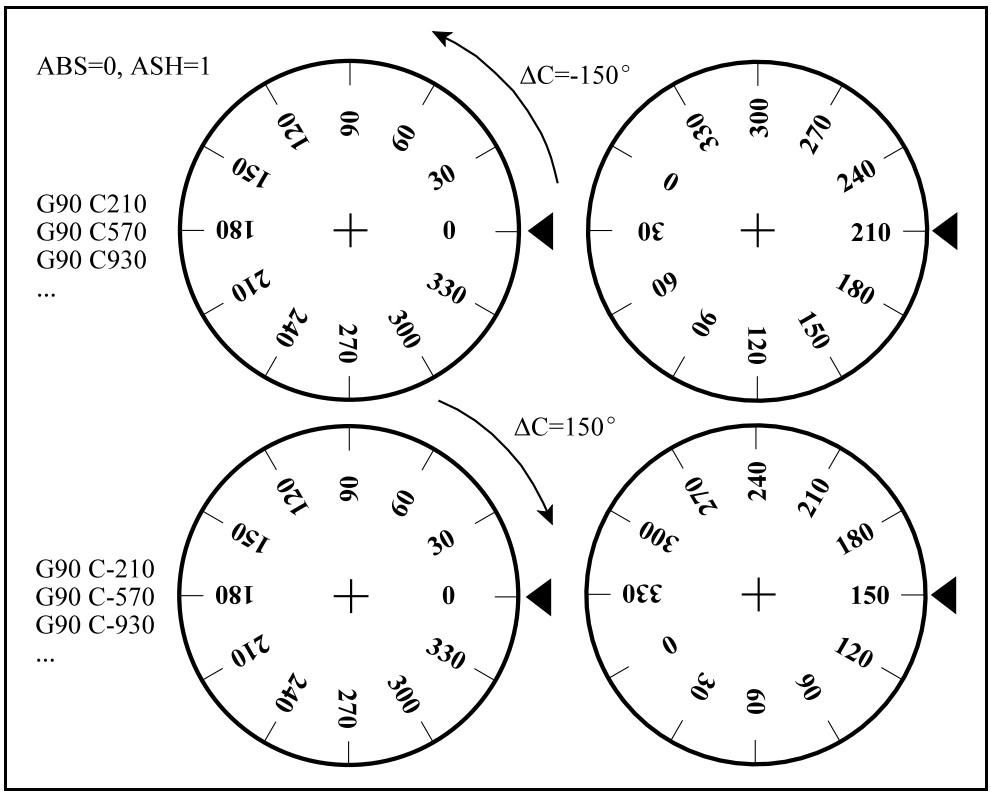


Fig. 5.6-3

Rotation to the Absolute Value of the Specified Position in the Direction Defined by the Sign

If the settings of the parameter N0107 RollOver Control are

#2 ABS=1 and #1 ASH=0,

in the case of absolute programming (G90) the rotation will be performed to the absolute value of the specified position and in the direction defined by the sign. The value of displacement will always be less than the value specified in the parameter RollAmount, i.e. during motion the control always cuts the full revolutions.

An example:

Let the starting position of the axis C be

C=0

In accordance with the instructions

```
G90 C30
```

```
G90 C390
G90 C750
...
```

it will always rotate 30 degree (ΔC=30°) in the positive direction, having cut the full revolutions, and the end position will be 30°.

Let the starting position of the axis C be

C=0

In accordance with the instructions

```
G90 C-30
G90 C-390
G90 C-750
...
```

it will always rotate 330 degree (ΔC= −330°) in the negative direction, having cut the full revolutions, and the end position will be 30°.

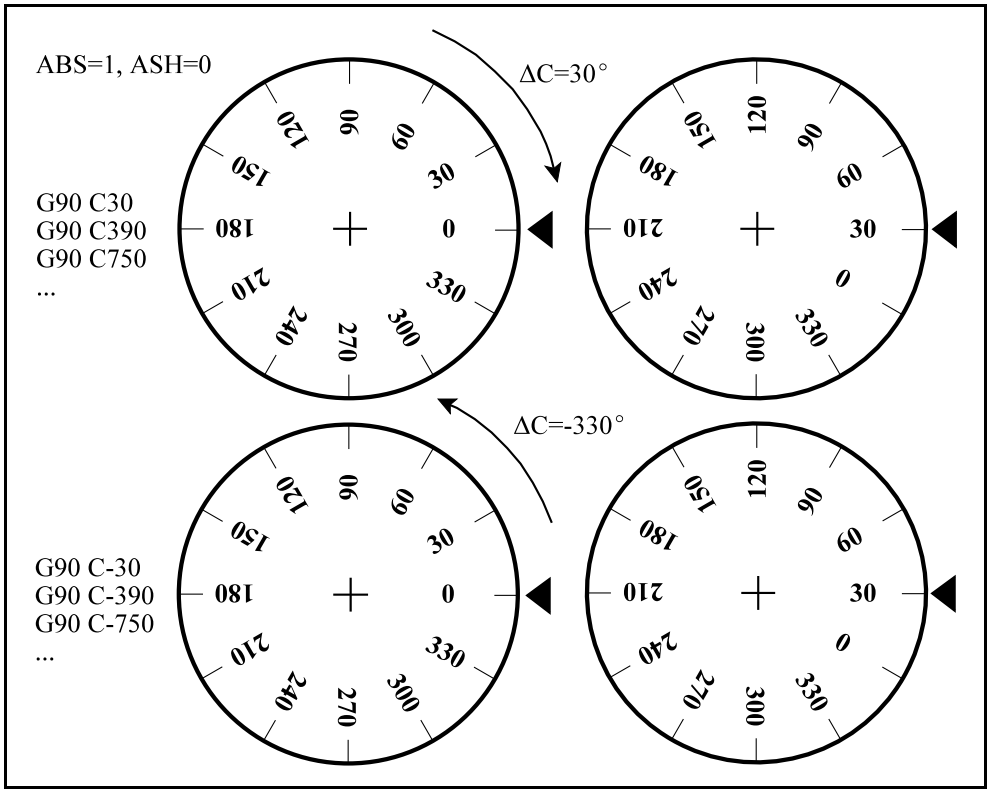


Fig. 5.6-4

☞ ***Note:*** The 0 (zero) is a positive number! Therefore, if position 0 is programmed, the motion will be performed in the positive direction.

Let the starting position of the axis C be

C=0

In accordance with the instruction

```
G90 C0
```

it will rotate 330 degree (ΔC=330°) in the positive direction, and the end position will be 0°.

If the motion from 30° to 0° in the negative direction is required, the block

```
G90 C-360
```

will has to be programmed. In this case, it will rotate 30 degree (ΔC= −30°) in the negative

direction, and the end position will be 0°.

Motion of a Rotary Axis in the Case of Incremental Programming

In the case of programming incremental data specification, the motion will be performed in the direction defined by the programmed sign.

If the setting of the parameter N0107 RollOver Control is

REN=0,

the control does not apply the parameter Roll Amount for the travel programmed incrementally, so multi-rotation displacement can be programmed using incremental specification.

Let the starting position of the axis C be

C=0

During execution of the blocks

```
G91 C30 (displacement 30 degree)
G91 C390 (displacement 390 degree)
G91 C750 (displacement 750 degree)
...
```

the control does not cut the full revolutions. On the other hand, the position of the axis C on the position display will always be 30 degree at the end of the motion, because the value set in the parameter RollAmount will always be valid for displaying the position.

If the setting of the parameter N0107 RollOver Control is

REN=1,

the control applies the parameter Roll Amount for the travel programmed incrementally too, so displacement greater than one rotation cannot be programmed even if using incremental specification.

Let the starting position of the axis C be

C=0

During execution of the blocks

```
G91 C30 (displacement 30 degree)
G91 C390 (displacement 30 degree)
G91 C750 (displacement 30 degree)
...
```

the control cuts the full revolutions. On the other hand, the position of the axis C on the position display will always be 30 degree.

6 Feed

6.1 Rapid Traverse

Positioning is executed by rapid traverse activated by the instruction G0. Apart from the G0 positioning, the positioning phases of the instructions G53, G28,G30 and the cycles are executed by rapid traverse too. The value of the positioning rapid traverse for each axis ***is set in parameter by the builder of the machine, in mm/min, inch/min or degree/min***. The rapid traverse values for the axes can differ from each other.
When several axes perform rapid traverse simultaneously, the value of the resultant feed will be calculated by the control in such a way so that neither of the speed components projected on the axes will exceed the rapid traverse value given in parameter and valid for that given axis, and the positioning will be performed in a minimum period of time.
The value of the rapid traverse can be modified by the rapid traverse percent (override) switch. Operation of the rapid traverse override is defined in the PLC program by the builder of the machine. The description of operation is contained in the manual provided by the builder of the machine.
The value of the rapid traverse override does not exceed 100%.
The rapid traverse will always be stopped by the position 0% of the rapid traverse percent switch .
If there is no valid reference point, the decreased rapid traverse values defined by the builder of the machine will be valid for the axes in the parameter field until reference point is done.
In the case of moving the slide by the slide-actuating (jog) buttons, the rapid traverse differs from the positioning rapid traverse, and it has a value different for each axis and set in parameter too. Evidently, its value is less than that of the positioning rate in order that the human response time can be calculated in for stopping.

6.2 Cutting Feed

Feed is programmed
at the address **F**.
The programmed feed will be valid in blocks of linear interpolation (G01) and blocks of circular interpolation (G02 and G03).
The unit of feed is determined by ***the codes G94 and G95***.
The feed is calculated by the control tangentially along the programmed path.

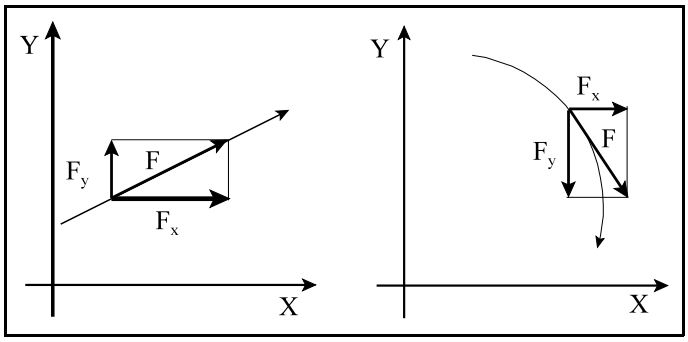


Fig. 6.2-1

F : tangential feed (programmed value)
F_x: feed component in the direction X
F_z: feed component in the direction Z

$$F = \sqrt{F_X^2 + F_Y^2}$$

The programmed feed can be modified using the feed percent (override) switch, excluding the G63, the percent switch and the inhibition of the stop states.
Operation of the feed override is defined in the PLC program by the builder of the machine. The

description of its operation and value limit are contained in the manual provided by the builder of the machine.
The feed is a modal value.
After switch on, the control takes the value defined in parameter. In the state G94 it gets the initial value of the feed from the parameter N0300 Default F G94, while in the state G95 from the parameter N0301 Default F G95.
For a given machine, the maximum programmable feed is limited for each axis by the builder of the machine in the parameter field. The value set here is always interpreted in minute dimension. When the switch DRY RUN is in switched-on position, this value, at the same time, is the speed of the feed motions.
The value of feed can be given with an accuracy of up to 15 decimal digit. Decimal point can be used. It is always a positive number.

6.2.1 Feed per Minute (G94) and Feed per Revolution (G95)

In the program, the unit of feed can be specified by the codes G94 and G95:

G94: feed per minute
G95: feed per revolution

The term 'feed per minute' refers to the feed specified in the units of mm/min, inch/min or deg /min.
The term 'feed per revolution' refers to the feed accomplished in a revolution of the spindle and specified in the units of mm/rev, inch/rev or deg/rev.
They are modal values.
After power-on, reset or program end, the bit #0 G95 of the parameter N1301 DefaultG2 determines whether the control will get into the state G94 or G95.
The state G94/G95 does not affect the rapid traverse which has to always be interpreted in 'per minute' dimension.

6.3 Feed Control Functions

The feed control functions are necessary
– in the case of ***machining corners***,
– in such cases when it is required by the technology so that the ***switches override and stop*** must be ***inhibited***.

In the case of machining corners in the mode of continuous cutting, the slides, because of their inertia, are not able to follow the path instructions issued by the control. Depending on the feed, the tool will round the corner more or less.
If sharp corners are required on the workpiece, the control must be told to decelerate at the end of the motion, to wait until the axes stop, and to start the subsequent motion following this only.

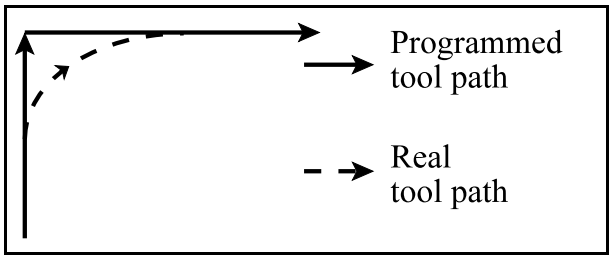


Fig. 6.3-1

6.3.1 Exact Stop at the End of the Block (G9)

This function is ***not a modal one***, it is valid in that block only in which it was programmed.
At the end of the bloc in which it was specified, the control decelerates after execution of

interpolation, ***it stops and waits for the in-position signal of the measuring system***.
If the in-position signal does not arrive even after expiration of the time set in parameter, the control will send the message '2501 Position error'.
This function is used for precision machining sharp corners.

6.3.2 Exact Stop Mode (G61)

This is a modal function. It can be cancelled by the instructions G62, G63 and G64.
The control decelerates ***at the end of each block***, ***it stops and waits for the in-position signal of the measuring system***, and starts the subsequent interpolation cycle following this only.
If the in-position signal does not arrive even after expiration of the time set in parameter, the control will send the message '2501 Position error'.
This function is used for precision machining sharp corners.

6.3.3 Continuous Cutting Mode (G64)

This function is a modal one. ***This is the state the control gets into after power-on, reset or program end***. It can be cancelled by the codes G61, G62 and G63.
In this mode of operation, the motion does not stop after execution of the interpolation, the axes do not wait for the in-position signal of the measuring system, but the interpolation of the next block starts immediately.
In this mode of operation, sharp corners cannot be machined because the control rounds them at transitions.

6.3.4 Override and Stop Inhibition Mode (G63)

This function is a modal one. It can be cancelled by the codes G61, G62 and G64.
In this mode of operation, ***the percent switches of feed and spindle (override) as well as stopping the feed are ineffective***. The control interprets the percent values as 100% independently of the positions of the switches. It does not decelerate after execution of the interpolation, but it starts the next interpolation cycle immediately.
This mode of operation can be used for machining threads.

6.3.5 Automatic Feed Override at Inner Corners (G62)

This is a modal function. It can be cancelled by the instructions G61, G63 and G64.
In the case of machining inner corners, the force acting on the tool increases on the sections before and after the corner. In order that the tool does not flutter and the surface remains appropriate, the control reduces the feed automatically on the sections before and after the inner corners, when the G62 is switched on.

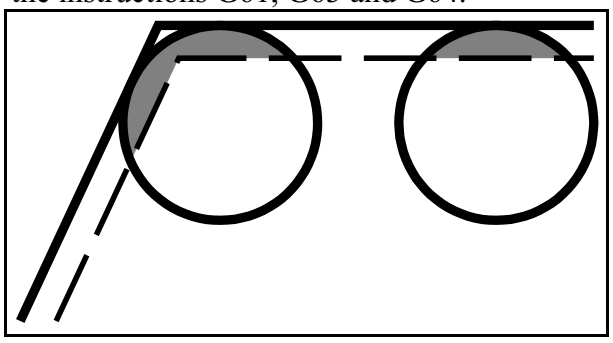


Fig. 6.3.5-1

The corner override is effective under the following conditions:
– if the planar tool radius compensation is switched on (G41, G42);
– between the blocks G1, G2 and G3;
– in the case of motions in the selected plane;

– if the tool machines the corner inside;
– if the angle of the corner less than an angle specified in parameter;
– before and after the corner, at a distance specified in parameter.
The feed override function acts for all the following four possible transitions: linear - linear; linear - circular, circular - line, circular- circular.

The value of the internal angle φ can be set in the parameter N1409 CornAngle, in the angle range of 1° - 180°.
Deceleration and acceleration will be commenced and finished at the distance L_d before and at the distance L_a after the corner, respectively. In cases of circle arcs, the distances L_d and L_a will be taken into account by the control along the arc.
The distances L_d and L_a can be specified in the parameters N1407 DecDist and N1408 AccDist, respectively.

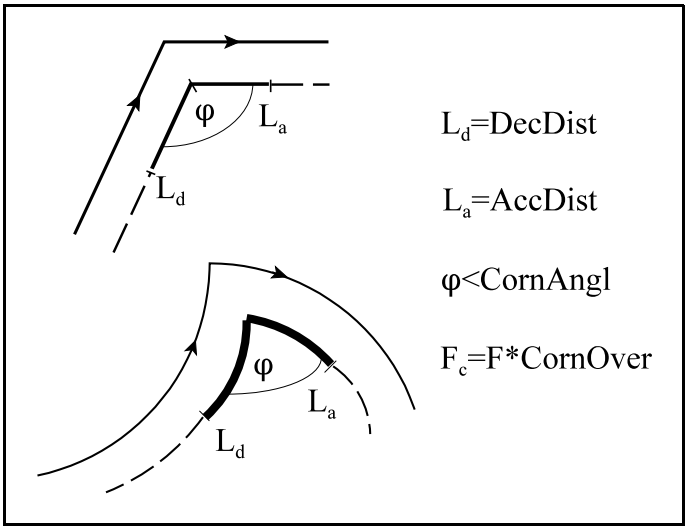


Fig. 6.3.5-2

Writing the ratio between 0 and 1 in the parameter N1410 CornOver there can be specified the value up to which the control has to reduce the feed at inner corners. The feed will be

F*CornOver

where F is the programmed feed. Even the override switch produces effect on the feed obtained, too.
If the aim is to program exact stop in the state G62, G09 will have to be written in the given block.

6.4 Automatic Feed Override in the Cases of Inner Circle Arcs

When the planar tool radius compensation is switched on (G41, G42), the programmed feed during circular interpolation will be effective along the tool center. In the case of machining inside circle arcs, the control reduces the value of the feed automatically so that the programmed feed will be effective along the cutting radius. The value of the feed in the center of the tool radius is:

$$F_C = \frac{R_C}{R} F$$

where F_c: the feed of the center of the tool radius (corrected feed);
R: the programmed circle radius;
R_c: the corrected circle radius;
F: the programmed feed.

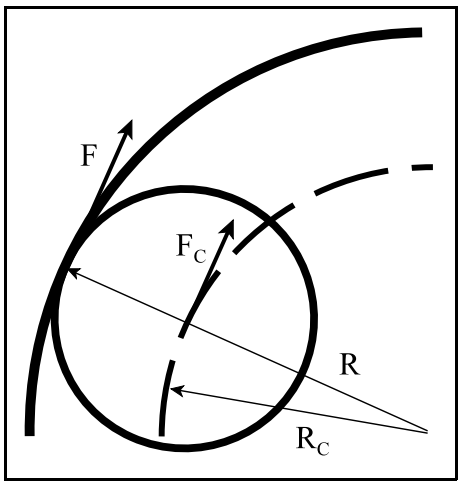


Fig. 6.4-1

The lower limit of automatic feed reduction is determined by the parameter N1406 CircOver, writing in which a ratio between 0 and 1 there can be specified the minimum value of feed reduction, i.e. the condition

$F_c \geq F*CircOver$

will be satisfied. The override for circle radius will be multiplied by the feed override and the corner override, and then it will be issued.

7 Acceleration

Acceleration describes how velocity changes over the course of time. The shorter the time necessary to reach a given velocity is, the great acceleration will be.
The higher acceleration we would like to reach, the stronger motors and drives of higher power are required.
During motion, the value of forces acting on the machine, after all the load on the machine is in direct proportion to the acceleration come into being.
Parameters of acceleration for each axis are set by the builder of the machine on the basis of the two abovementioned considerations

The control accelerates always the value of the tangential (vectorial) feed. It calculates the value of acceleration in such a way so that the value of the axis component of acceleration does not exceed, on none of the axes, the value set for the given axis.
Acceleration of two kinds can be set:
– linear acceleration, and
– bell-shaped acceleration.

In the case of ***linear acceleration***, the value of acceleration is constant during acceleration/deceleration, the control increases the feed at start or reduces the feed at stop in accordance with a linear function.
The value of acceleration can be set in parameter for each axis.

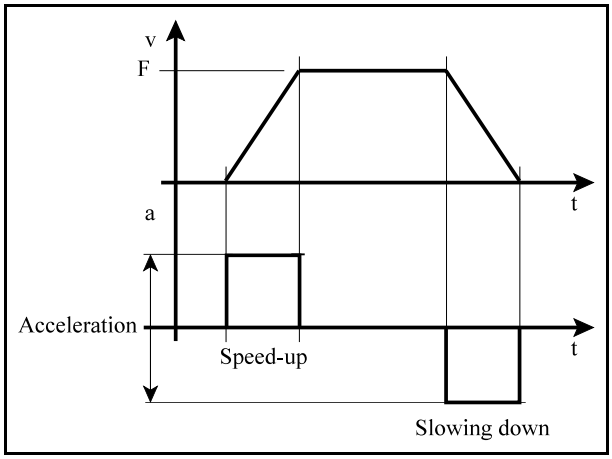


Fig. 7-1

In the case of ***bell-shaped acceleration***, the value of acceleration changes during acceleration/deceleration, it increases linearly until it reaches the set value of acceleration, or it decreases linearly until it reaches the target speed. As a consequence of this, the shape of the feed leading and trailing branches is a bell-shaped curve as a function of time, and because of this the acceleration of this kind is called ‘bell-shaped’.
The time T, during which the control reaches the set value of acceleration, can be set in parameter for each axis.

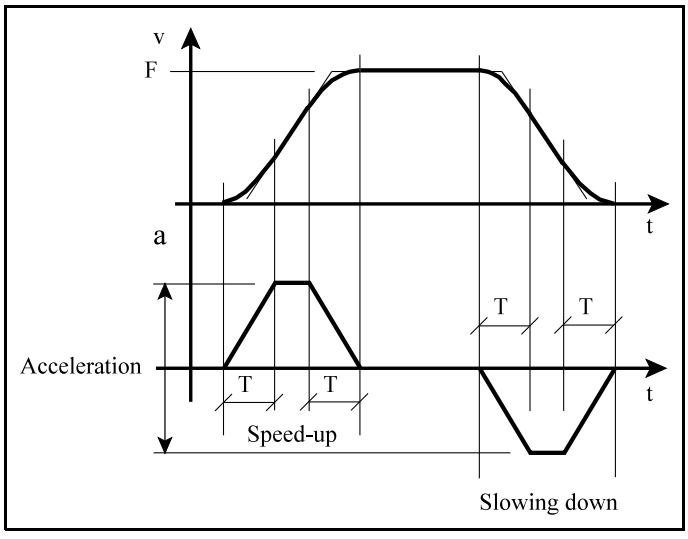


Fig. 7-2

By setting ***acceleration without jerk*** the bell-shaped acceleration can be soften further. Acceleration without jerk can be switched on by the bit position #1 JRK=1 of the parameter N0421 Acc Contr. In this case, already the leading and trailing branches of the acceleration function will be bell-shaped; in other words, there will not be jump in the first derivative (j) of the acceleration (a) either.

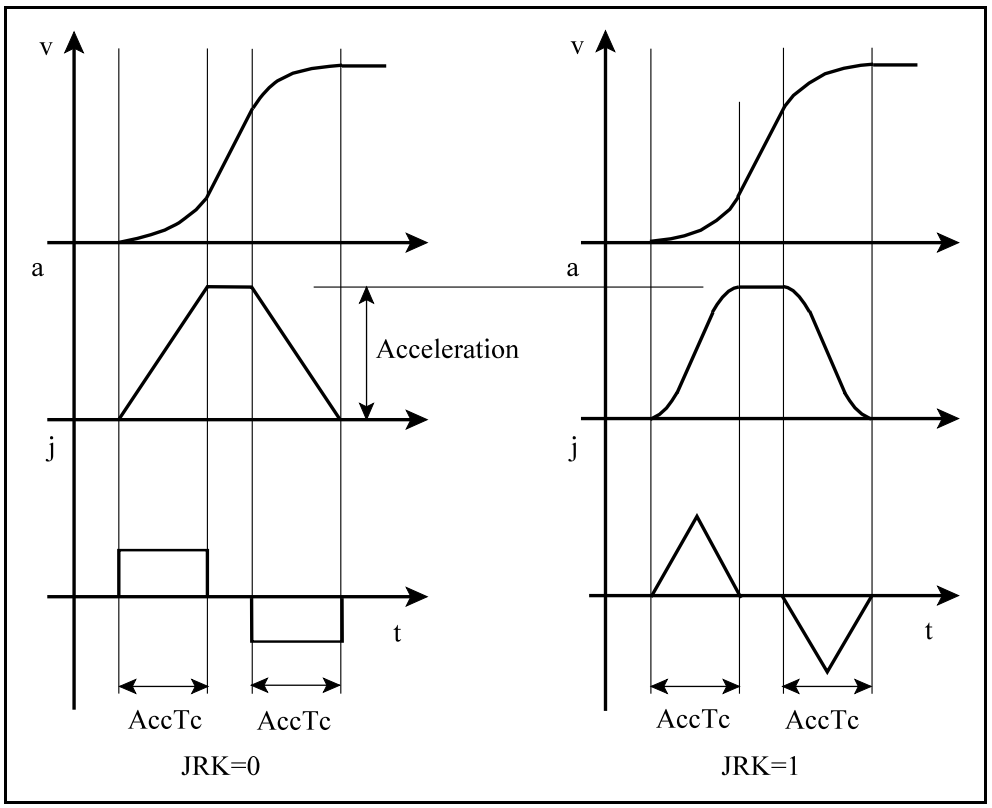


Fig. 7-3

By setting acceleration without jerk JRK=1, higher acceleration can be set on the machine, on the other hand, the start and the stop will be softer.

In the case of high-speed machining, speed feedforward has to be used in order to reach the accuracy required. In this case, bell-shaped acceleration must always be set.

In normal conditions, the control accelerates or decelerates in the following cases:
– when manual actuation is performed;
– in the case of rapid traverse positioning (G0), the motion always starts from the speed 0 at the beginning of the block, and it always decelerates to the speed 0 at the end of positioning;
– in case of feed motions (G1, G2 and G3), in the state G9 or G61, the controlalways decelerates to the speed 0 at the end of the block;
– the control will decelerate if motion is stopped by the button STOP, and it will accelerate if motion is started by the button START;
– the control will stop with deceleration if function is executed after motion, and at the end of the block in BLOCK BY BLOCK mode.

Acceleration to a new feed value greater than previous one will always be started by the control during execution of the block, in which the new feed is given. This process may cover several blocks too, if necessary.
Deceleration to a new feed value less than the previous on will always be started by the control in a proper previous block in such a way so that the control starts the machining with the speed programmed in that block in which the new feed is given.

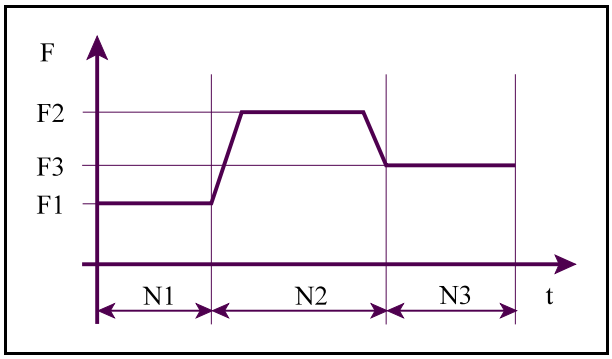


Fig. 7-4

Tangential speed changes are pre-monitored and registered by the control. It is necessary to reach the desired target speed by continuous acceleration covering execution of several blocks as well.
Either at start or at stop, reaching the desired speed may cover several blocks.

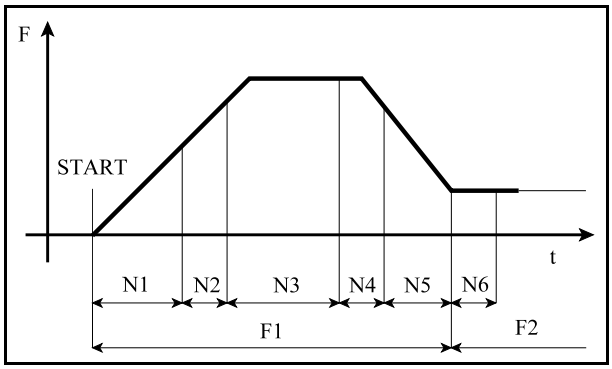


Fig. 7-5

7.1 Automatic Deceleration at Corners in the State G64

In case of continuous cutting, in the state G64, the control tries to follow the path at the programmed feedrate.
If a corner is found between two blocks, the control will have to decelerate the tangential feed.

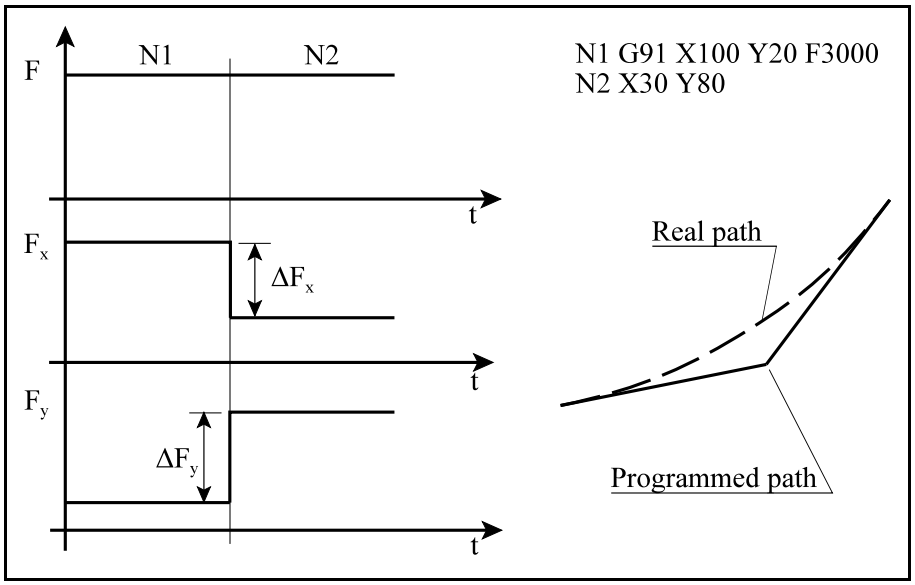


Fig. 7.1-1

If no deceleration is executed at the corner in two subsequent blocks N1 and N2 , the feed differences (ΔF_x and ΔF_y) shown in the figure will occur along the respective axes.

Detection of feedrate changes (corners) and deceleration of feed at the same time are necessary for the following two reasons:

– Feedrate changes for the axes deriving from sudden change in direction of the path may be so big, that without deceleration the drives will not be able to follow them without swinging; as a consequence, accuracy will decrease and mechanical load on the machine will increase extremely.

– If, in the course of cutting, sharp corner is to be formed, but without stop (without programming exact stop G61) since it increases cutting time, it will be necessary to decelerate too. The more the feed is decelerated, the sharper the corner will be.

The control can detect corners by monitoring either the change in direction angle of the path or the change in axis components of the feed. The method to be used for deceleration can be chosen by parameter.

Deceleration at Corners by Monitoring the Change in the Direction Angle of the Path

At the bit position #0 FDF=0 of the parameter N0306 Feed Control, deceleration will be executed by monitoring the change in the direction angle of the path.

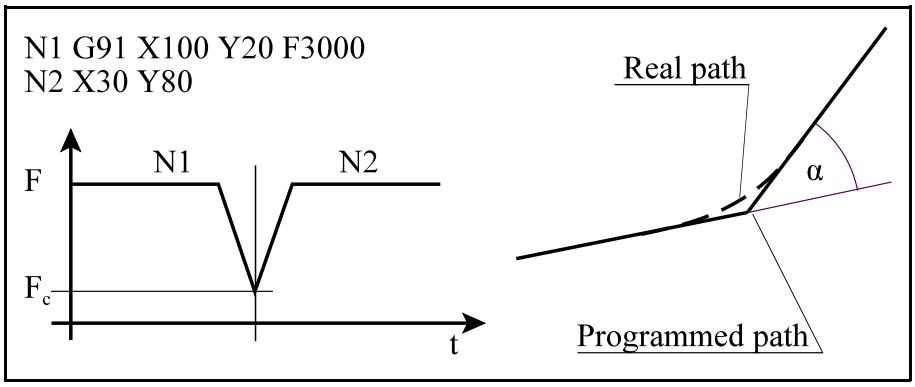


Fig. 7.1-2

If, at the meeting point of the blocks N1 and N2 shown in the figure, ***the value of the angle α exceeds*** the value enabled in parameter, the control will decelerate the feed to the value F_c.
It is the parameter N0307 Crit A Diff , in which the value of the critical angle can be set in degree: α=Crit A Diff. The value in the parameter N0308 Feed Corn determines the value of corner feed to which the control has to decelerate in the case of exceeding the critical angle: F_c=Feed Corn.
The greater the values of the critical angle and the corner feed are, the faster the machining will be, but the bigger the load acting on the machine will be and the more rounded the corner will be.
This set-up is not appropriate for high-speed machining.

Deceleration at Corners by Monitoring the Change in Axis Components of the Feed

At the bit position #0 FDF=1 of the parameter N0306 Feed Control, deceleration will be executed by monitoring the change in axis components of the feed.

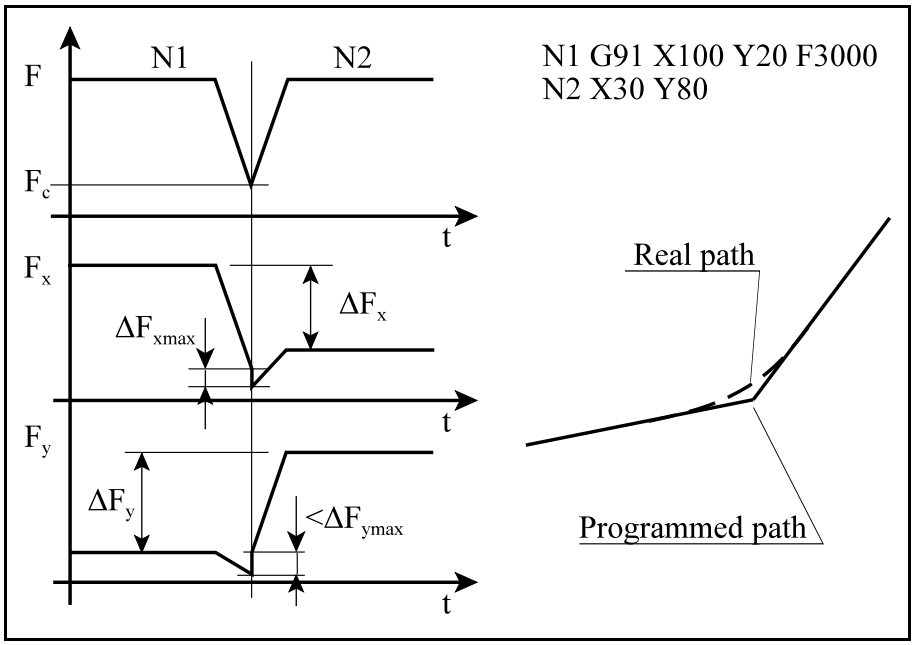


Fig. 7.1-3

If, at the meeting point of the blocks N1 and N2 shown in the figure, the change in axis components of the feed ΔF_x, ΔF_y exceeds the maximum value enabled in parameter, the control will decelerate the tangential feed F to the value F_c.
The control decelerates feed in such a way, so that the value of feed change does not exceed, on none of the axes, the critical feed difference (ΔF_{xmax}, ΔF_{ymax}) enabled for the given axis in parameter; the critical feed difference for a given axis can be specified in the parameter N0309 Crit F Diff : ΔF_{xmax}=Crit F $Diff_x$, ΔF_{ymax}=Crit F $Diff_y$.
The greater the value of the critical feed difference is, the faster the machining will be, but the bigger the load acting on the machine will be and the more rounded the corner will be.
The control bounds the value of the preset critical feed difference from above, on the basis of acceleration set-up.
This set-up must be applied for high-speed machining.

7.2 Limiting the Normal Direction Acceleration

In the course of machining, the control keeps the feed constant along the tangent of the path (in tangential direction). As a consequence, acceleration components do not come into being in tangential direction. The situation is different in normal direction (in the direction perpendicular to the path and the speed). The axial components of the normal direction acceleration may exceed the value permissible for the given axis. In order to avoid this, the speed along the path must be limited in proportion to the curvature of the path.
The maximum permissible value of the normal direction acceleration can be set in the parameter N0402 Normal Acc.

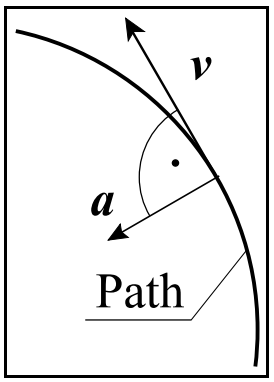


Fig. 7.2-1

Limiting the Normal Direction Acceleration in case of Circular Arcs
In the course of machining circular arcs, the control limits the value of the feedrate F according to the formula

$$F = \sqrt{a \cdot r}$$

where:

a: the lesser one among the acceleration values set for axes participating in circular interpolation ;

r: the radius of the circle.

The circular interpolation will already be started using the rate calculated in this way.
The control does not decrease the feedrate under the value specified in the parameter N0310 Circ F Min, independently of formula above.

☞ ***Warning!***

This function should not be confused with automatic feed override in states G41and G42, in the course of machining inner circular arcs.

Limiting the Normal Direction Acceleration in case of Other Interpolations

Though ***in the case of linear interpolation***, the given line segment (path) does not have curvature, therefore there is no normal direction acceleration component either, but this is true for long straight line segments only. If a ***path is made up of minute straight line segments***, as it is usual in manufacturing tools, the curvature of the path resulted in such a way can be significant, and feed will have to be decreased, as it is illustrated in the figure below:

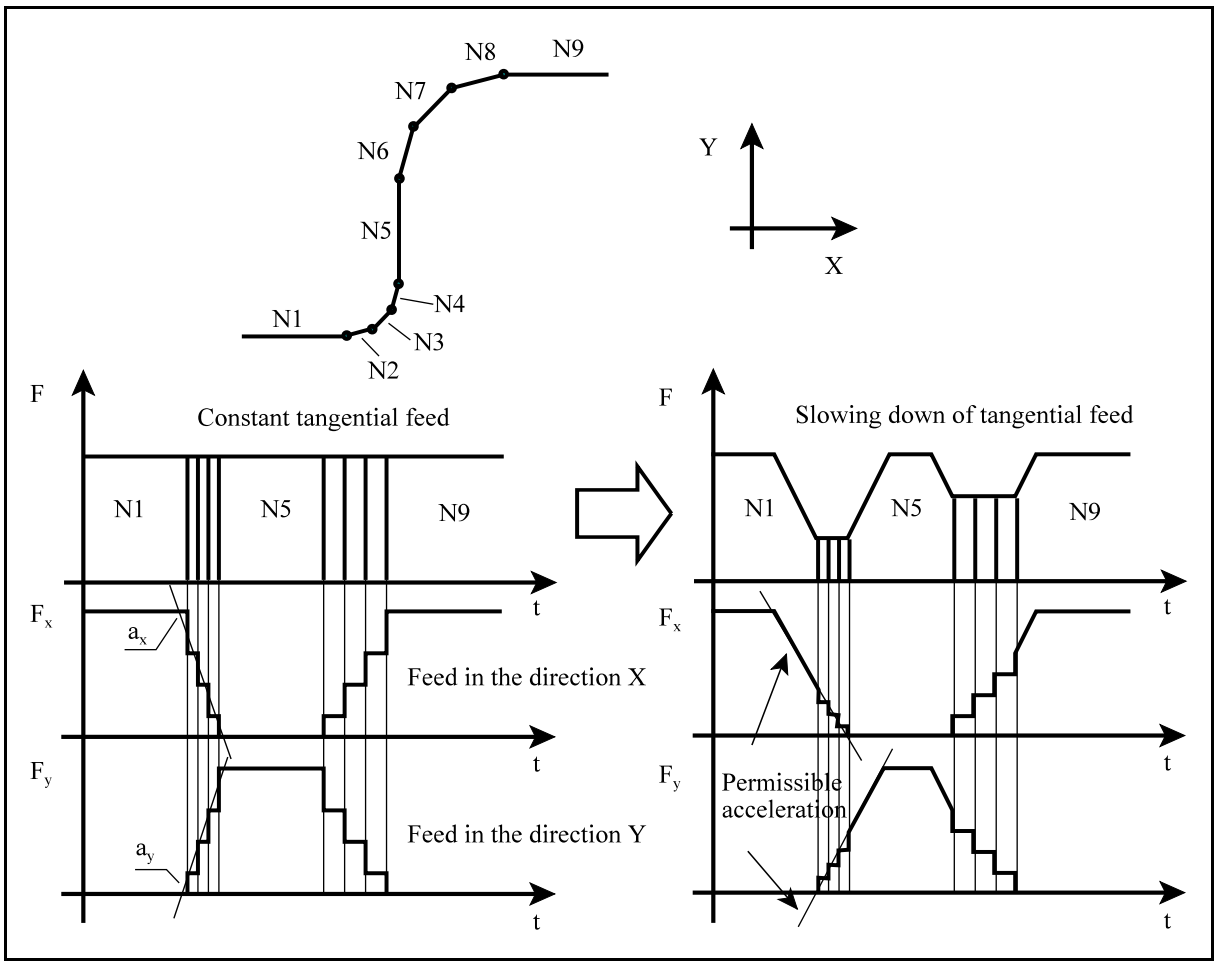


Fig. 7.2-2

In the blocks N2, N3, N4 and N6, N7, N8, the path is made up of minute straight line segments. If the value of tangential feed is kept constant (the left part of the figure above), the gradient of speed change (acceleration in the normal direction) on the axes X and Y may exceed the value permissible for the given axis, because of geometry (direction change) of the path.
For this reason, the control scans the path block by block in order to be able to limit normal direction accelerations. Where, from the geometry, the acceleration components on given axes are greater than the permissible values, the tangential speed must be decreased. The graphs on the right part of the figure shows, how the measure of speed change (the normal direction acceleration) decreases on the given axes in proportion to deceleration of tangential feed.

In the case of the ***smooth interpolation*** G5.1 Q2, the control also examines the normal direction accelerations derived from the path curvature, and it decreases the feed, if necessary.

7.3 Limiting the Acceleration Step Change (Jerk)

On the certain sections of the path, sudden step change, jerk may occur that causes swings, loads the machine mechanically , and appears on the surface machined. This is the case, when in the course of machining a tangent circle follows a straight line segment or a circle arc is followed by a tangent straight line.
The purpose of this function is that the control limits the measure of acceleration step change in the transition point by decreasing the feed.

Limiting the Acceleration Step Change at the Beginning and at the End of Circle Blocks
In the case of entering from a straight line segment into a tangent circle arc at a feedrate F, the value of the acceleration step change can be calculated using the formula

$$a = \frac{F^2}{r}$$

where:

a: the value of the acceleration step change,
r: the radius of the circle.

For example, if the machine, at the feedrate F6000, enters into a circle arc with radius of 10 mm, as it is illustrated in the figure below, the value of the acceleration step change on the axis Y will be:

$$a = \frac{\left(\frac{6000}{60}\frac{mm}{\text{sec}}\right)^2}{10mm} = 1000\frac{mm}{\text{sec}^2}$$

In order that the value of the acceleration step change will not be greater than 250mm/sec^2, the feedrate has to be decreased to the value F=50*60=3000 mm/min, using the equation above.

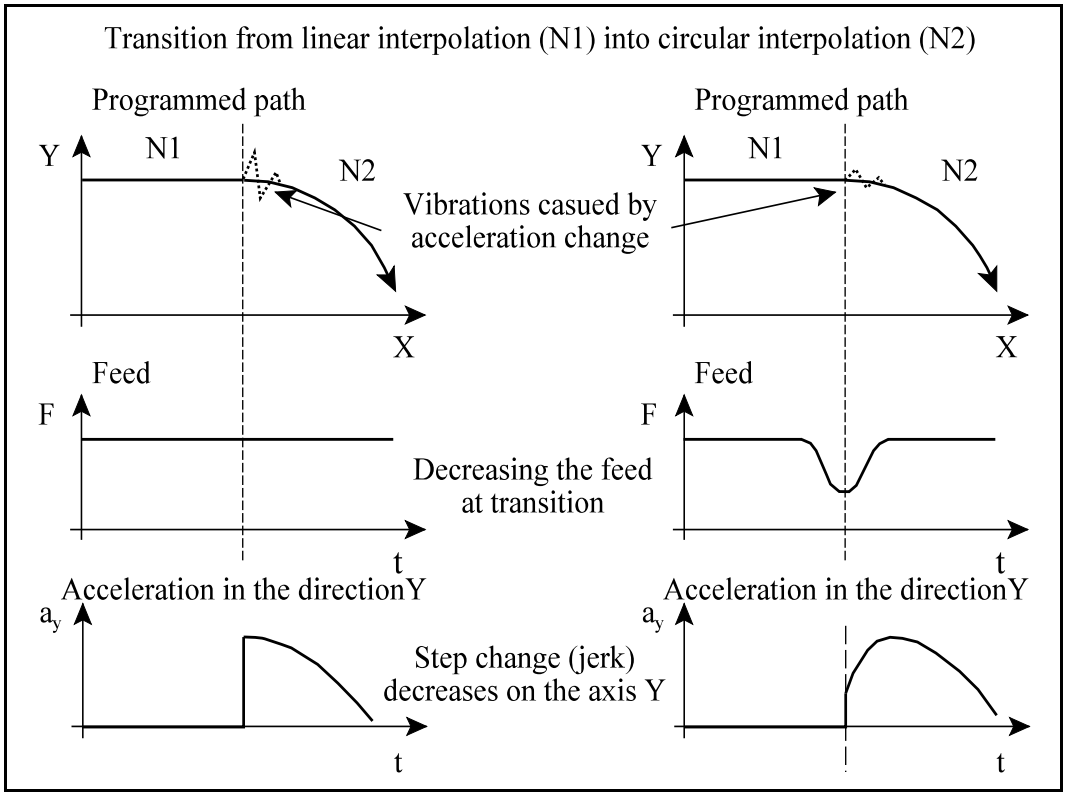


Fig. 7.3-1

In case of circle arcs, the acceleration step change can be limited in the parameter N0404 Acc Diff Circ.

Limiting the Acceleration Step Change in Straight Line Blocks following Each Other

If the path is composed of long straight line segments, the measure of acceleration change will be negligible. In this case, the change in axis components of the feed could limit the feed.

The situation will be different if the path is made up of minute straight line segments. In this case, there could take place the situation, when the feed change on the given axes is small between two straight line segments; because of this the interpolator does not limit the feed, but the values of acceleration step change on the given axes are great. In such cases, the feed must also be limited depending on the permissible acceleration step change.

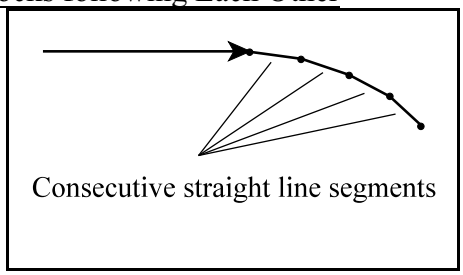


Fig. 7.3-2

In case of straight line segments following each other, the acceleration step change can be limited in the parameter N0403 Acc Diff.

8 Dwell (G4)

Using the instruction

G94 **G4** P....

dwell time can be programmed in second.

Using the instruction

G95 **G4** P....

dwell time can be programmed in number of spindle revolutions.

The accuracy of P is 15 decimal digits.

At the bit state #1 SEC=1of the parameter N1337 Execution Config , dwell is always counted in second, even if in the state G95.

Dwell always means the programmed delay of execution of the next block. It is a non-modal function.

9 Reference Point

The reference point is the point of the axis where the measuring system sends out position 0.
On the axes equipped with incremental measuring devices this point has to be found. This process is called reference point return.
Measuring the workpiece coordinate systems and positioning to an absolute position can be done after finding the reference point. The parametric end-positions and programmed travel limits will be effective after reference point return only.

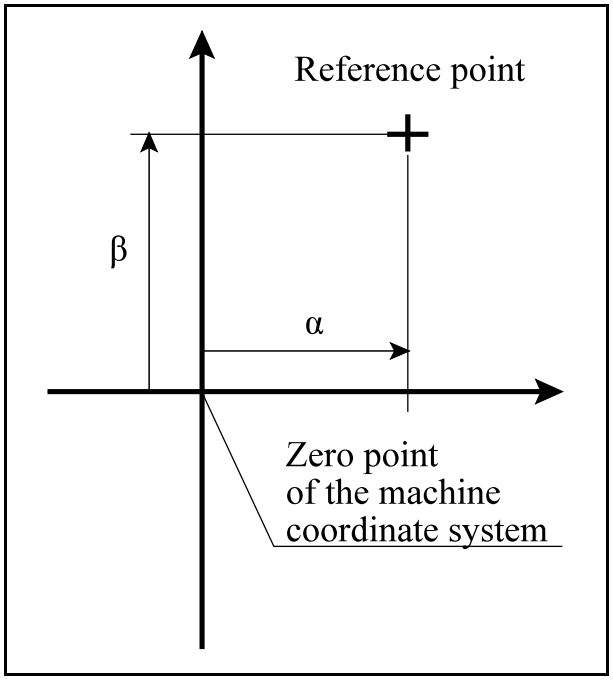


Fig. 9-1

Positions are recorded by the control not as values relative to the reference point, but as values in the machine coordinate system.
The zero point of the ***machine coordinate system*** is specified by the builder of the machine and it is a significant point on the machine tool.
The control records the change positions, center of rotation of the rotary axes etc. in the machine coordinate system. All the compensations (for thread pitch, straightness etc.) of the machine measuring system are recorded in the machine coordinate system, too.

The ***position of the reference point*** is recorded by the control ***in the machine coordinate system too***, and this position is set by the builder of the machine in parameter

If the axis is equipped with ***incremental measuring system***, return to the reference point must be executed after power-on. In the course of reference point return, the slides run onto a switch in the direction specified in parameter, and then, coming down from it, they look for the zero pulse of the measuring system and record existence of the reference point. The value given in parameter will be the position in the machine coordinate system.

If the axis is equipped with ***distance-coded measuring system***, return to the reference point must be executed after power-on. In the course of reference point return, the slides start in the direction specified in parameter, and then they look for two zero pulses and record existence of the reference point. The position of the zero pulse found as second one will be the position in the machine coordinate system.

If the axis is equipped with ***absolute measuring system***, return to the reference point does not have to be executed after power-on.

The reference point is also the point ***in the cases of both distance-coded and absolute measuring systems*** where the measuring system sends out position 0. Generally, this point is ***not***

within the working range of the machine. For this reason, the builder of the machine, by the use of parameterization, shifts this point into the working space, e.g. near to the positive endpoint, and then he measures this shifted reference point to the origin of the machine coordinate system. It is necessary, for example in the case, when the instruction G28 is to be used in the part program, for example for positioning together with tools.

9.1 Automatic Reference Point Return (G28)

The instruction

G28 v

will return the axes defined by the vector v to the reference point. There are two phases of the motion.

Phase 1

At first, the control, taking the coordinates given ***in the actual workpiece coordinate system*** into account as ***intermediate point***, executes rapid traverse along the axes defined by the vector v to the intermediate point specified by the vector v. The specified coordinate values can be either absolute or incremental ones. When the intermediate point is reached, the planar tool radius compensation is deleted.

Phase 2

Then, from the intermediate point, reference point return will be executed simultaneously on the axes specified by the vector v.

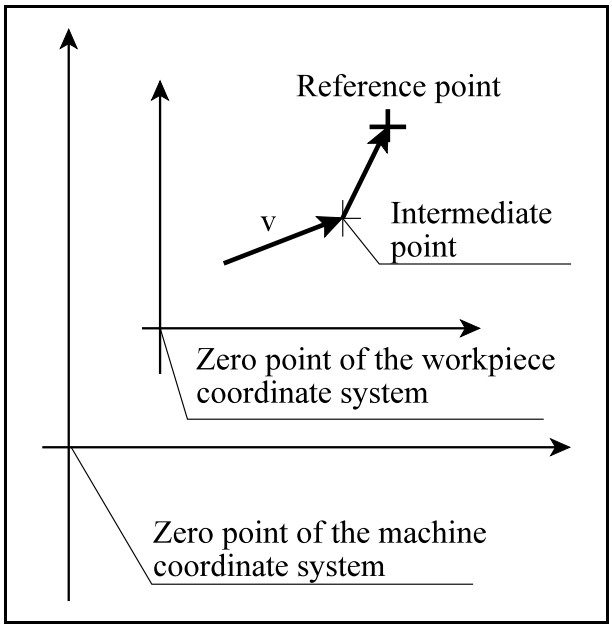


Fig. 9.1-1

If reference point return on a given axis did not occur yet, it would occur according to the mode determined by the manual reference point return. In this case,

– if the axis is equipped with ***incremental measuring system***, the reference point position specified in parameter will be the machine position at the end of the motion;
– if the axis is equipped with ***distance-coded measuring system***, the position of the second zero pulse will be the machine position at the end of the motion.

If reference point return on a given axis occurred already, or the axis is equipped with absolute measuring system, the axis executes rapid traverse to the reference point position specified in the machine coordinate system.

The code G28 is not a modal one.
For example:

```
G90 G28 X100 Y50 (the intermediate point: X=100, Y50)
```

If the position X is X=20 and the position Y is Y=50:

```
G91 G28 X100 Y50 (the intermediate point: X=120, Y=100)
```

☞ *Note*:
– If there is no valid reference point yet, incremental values must be assigned to the intermediate coordinates of v existing in the instruction G28.

9.2 Return to the 2nd, 3rd and 4th Reference Point (G30)

Three additional significant points called ***2nd, 3rd and 4th reference point*** can be specified ***in parameter***, in the ***machine coordinate system***.
These reference points are used on the machine to store change positions, e.g. tool change position, pallet change position etc.
Motion to these change positions is permitted after execution of reference point return only.

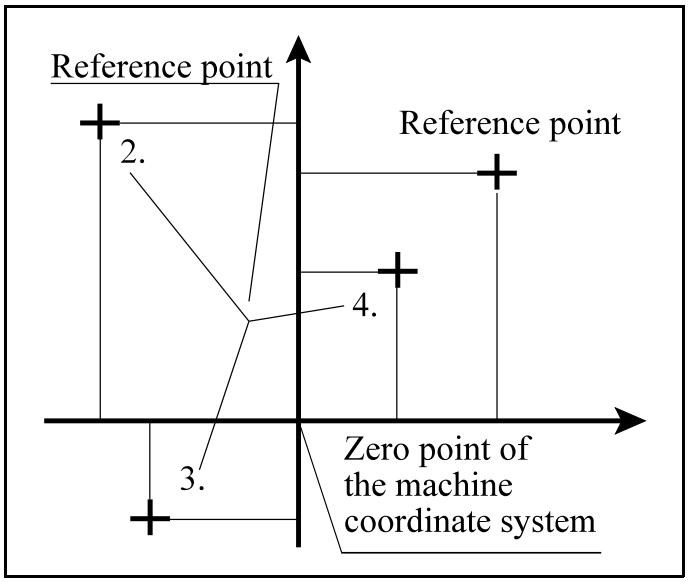


Fig. 9.2-1

The series of instructions

G30 v P

sends the axes coordinates of which are specified at the addresses of the vector v, to the reference point specified at the address P.

P2: the 2nd reference point
P3: the 3rd reference point
P4: the 4th reference point

There are two phases of the motion as it was the situation in the case of the instruction G28.
At first, the control, taking the coordinates specified by the vector v into account as intermediate point, executes rapid traverse of linear motion ***to the intermediate coordinates*** specified by the vector v. The specified coordinate values can be either absolute or incremental ones. The motion is always executed in the actual coordinate system. When the end point of the linear motion is reached, the planar tool radius compensation vector is deleted.
In the second phase, the axes specified by the vector v execute rapid traverse from the intermediate point ***to the reference point selected at the address P***.
Travelling to the reference point is carried out by ignoring compensation vectors (length, offset, 3D radius); they do not have to be deleted before issuing the instruction G30, but the control will implement them in the course of programming further motions. The planar tool radius compensation resets automatically in the first motion block.
It is not a modal code.

The instruction G30 v P1moves the machine to the reference point, its effect and the effect of the instruction G28 is the same.

For example:

```
G90 G30 X100 Y50 P3 (the intermediate point X=100, Y50
                     moves to the P3)
```

If the position X is X=20, and the position Y is Y=50:

```
G91 G30 X100 Y50 P4 (the intermediate point X=120, Y=100
                     moves to the P4)
```

10 Coordinate Systems and Plane Selection

In the program, a position the tool is to be moved to is specified by coordinate data. When three axes (X, Y and Z) are used, the position of the tool is expressed by three coordinate data: X____, Y____ and Z____.

As many axes are there on the machine, so many coordinate data express the tool position. The coordinate data have to always be interpreted in a given coordinate system.
The control differentiates the following three coordinate system:

1. The machine coordinate system
2. The workpiece coordinate system
3. The local coordinate system

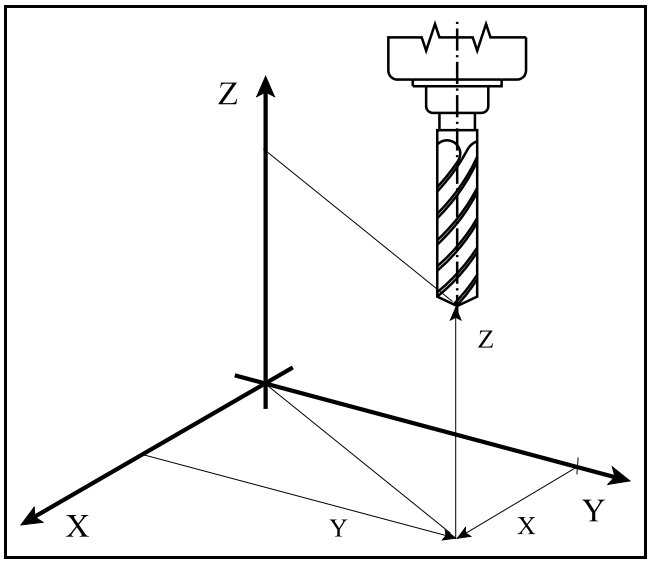


Fig. 10-1

10.1 Machine Coordinate System

The control stores positions in the machine coordinate system and not as ones relative to the reference point.
The zero point of the machine coordinate system is specified by the builder of the machine and it is a significant point on the machine tool.
The control records the change positions, center of rotation of the rotary axes etc. in the machine coordinate system. All the compensations (for thread pitch, straightness etc.) of the machine measuring system are recorded in the machine coordinate system, too.

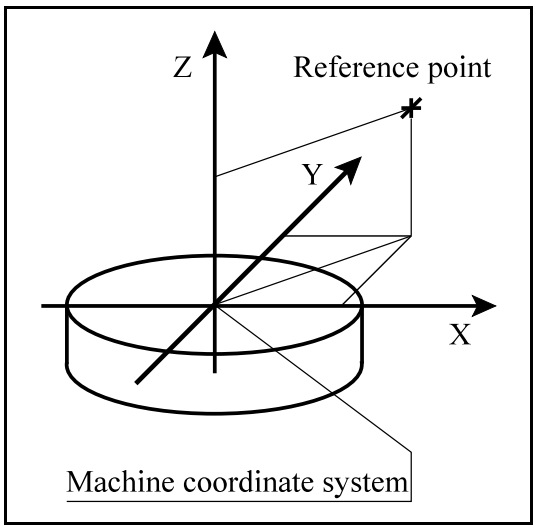


Fig. 10.1-1

The ***position of the reference point*** is recorded by the control ***in the machine coordinate system too***, and this position is set by the builder of the machine in parameter.

The position of the machine coordinate system cannot be changed by the use of any instruction or offset.

10.1.1 Positioning in the Machine Coordinate System (G53)

The instruction

G53 v

moves the tool to the point of coordinate v in the machine coordinate system.

– Independently of the state of G90 and G91, coordinates of v are always interpreted as ***absolute coordinates***.
– After the address of the coordinates or in the case of using address U, V and W for incremental specification, the operator I will send the error message '2097 Illegal incremental moving on ... axis'.
– Motion is always ***rapid travel***, similarly to the case of G00.
– Positioning is always executed by ignoring the selected and set tool compensations (length and radius).

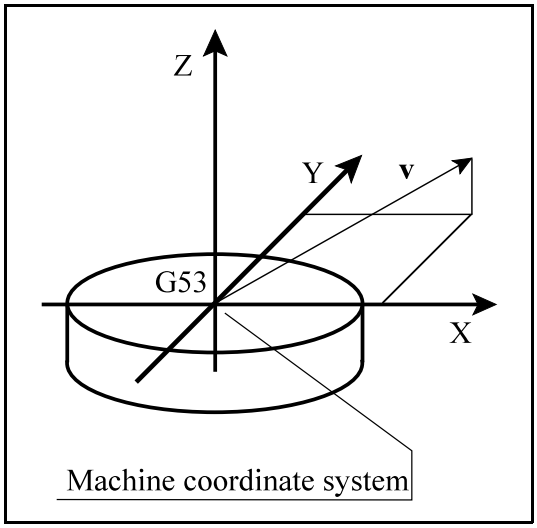


Fig. 10.1.1-1

The instruction G53 can be executed after reference point return only. The instruction G53 is a ***one-shot*** instruction, it is effective only in the block in which it is given.

An example: The effect of the instruction

```
G53 X200 Y150 Z20
```

is motion to the specified point in the machine coordinate system.

☞ ***Warning!*** *The instruction G53 suspends the read ahead of the blocks(buffering). Therefore, the instruction G53 in itself without coordinate specification can also be used to suspend reading ahead of the blocks, that is to empty the buffer.*

10.2 Workpiece Coordinate Systems

The coordinate system in which the part program has to be written is named workpiece coordinate system. The control stores the origin of the workpiece coordinate system relatively to the machine coordinate system.

The origin of the workpiece coordinate system is fixed to an appropriate point on the workpiece. This point can be one of the corners of the workpiece, the center point of a hole or a collar etc. Setting can be carried out in the following ways:

within the machine by manual measurement, or by measurement using probe; or
outside the machine.

In the latter case, the values measured outside have to be input in the memory of the control. Data input can be done manually, or from program by the use of NC instructions.

Relationship between the Workpiece Coordinate Systems and the Channels

Offsets of the workpiece coordinate systems refer to the given axes. Since each axis is assigned in parameter to a given channel, therefore, each channel has different workpiece offset table.

If one or more axes are interchanged between two channels, the axes will take their zero point offset away into the new channel. In such cases, after axis interchange, it is practical to calla new

workpiece coordinate system together with absolute positioning, and to continue the machining after that.

10.2.1 Selecting the Workpiece Coordinate System (G54...G59)

In basic version, 6 different workpiece coordinate systems are stored by the control. The ***offsets*** of the workpiece coordinate systems ***relative to the origin of the machine coordinate system*** must be given for each axis separately.

The case, when there is no common zero point offset, illustrated in the figure below.

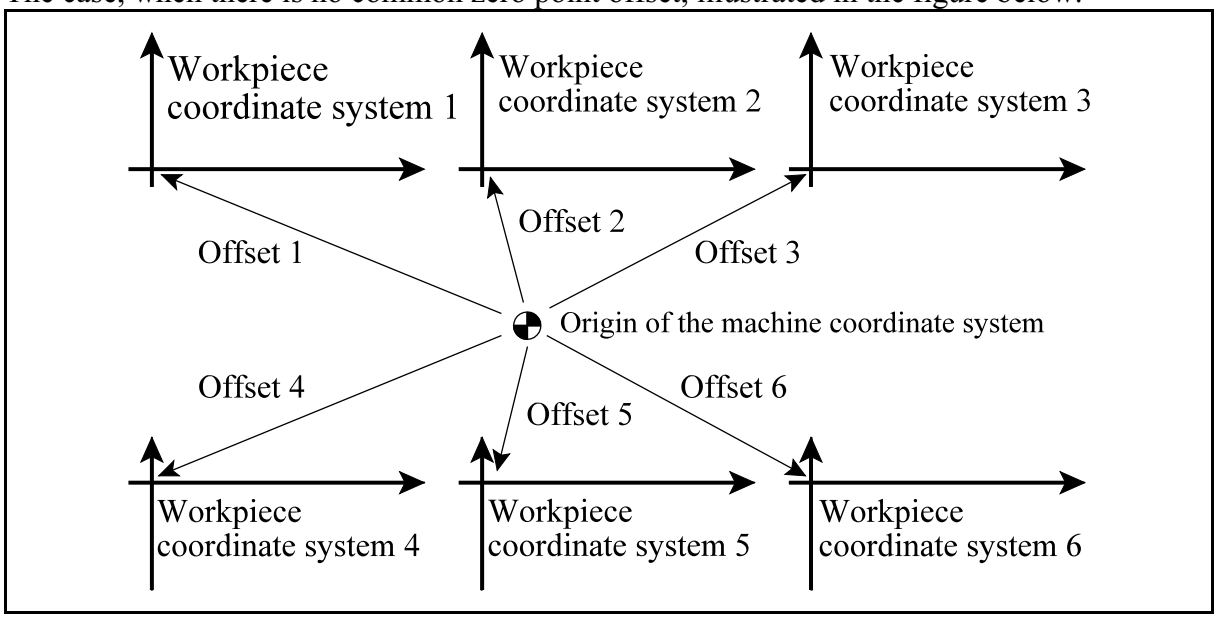


Fig. 10.2.1-1

All the workpiece coordinate system can be offset relative to the origin of the machine coordinate system. ***A common zero point offset will shift the origins of all the workpiece coordinate systems*** relative to the machine coordinate system.

The following figure shows the case, when there is common zero point offset.

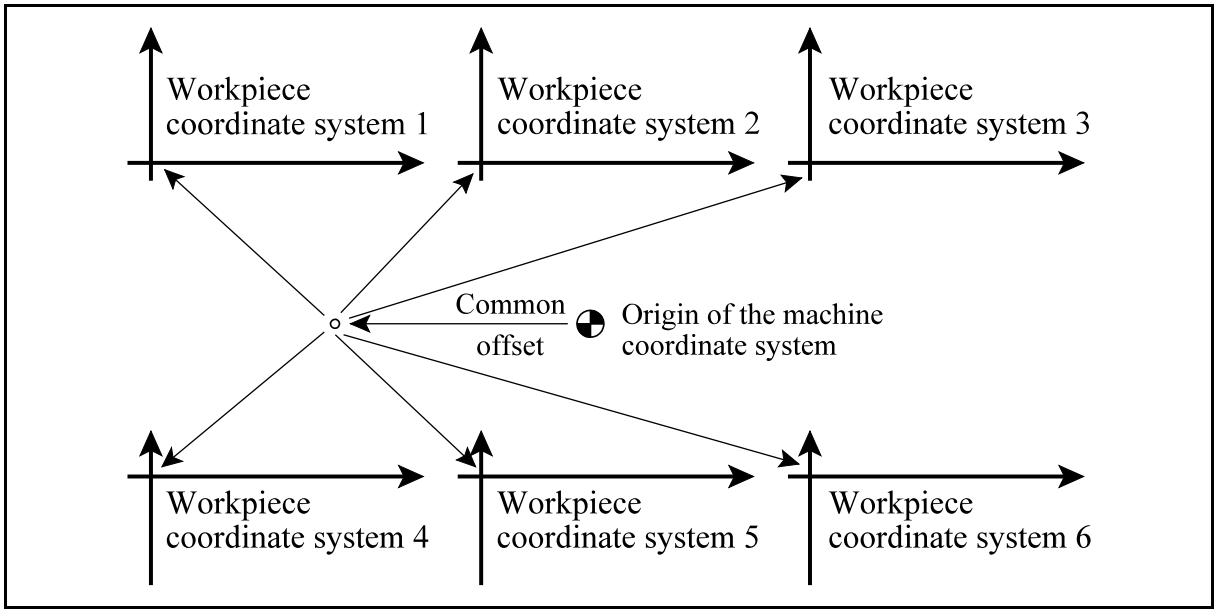


Fig. 10.2.1-2

The instructions G54 ... G59 are those, by the use of which the selection can be done from among the various workpiece coordinate systems.

G54: Workpiece coordinate system 1
G55: Workpiece coordinate system 2
G56: Workpiece coordinate system 3
G57: Workpiece coordinate system 4
G58: Workpiece coordinate system 5
G59: Workpiece coordinate system 6

These functions are modal ones.
After power-on, reference point return, reset or program end, the coordinate system G54 will be selected.
The absolute coordinate data of the interpolation blocks are taken into account by the control in the actual workpiece coordinate system.
For example, in the case of the instruction

```
G56 G90 G00 X60 Y40,
```

positioning to the point

X=60, Y=40

of the workpiece coordinate system 3 will be executed.

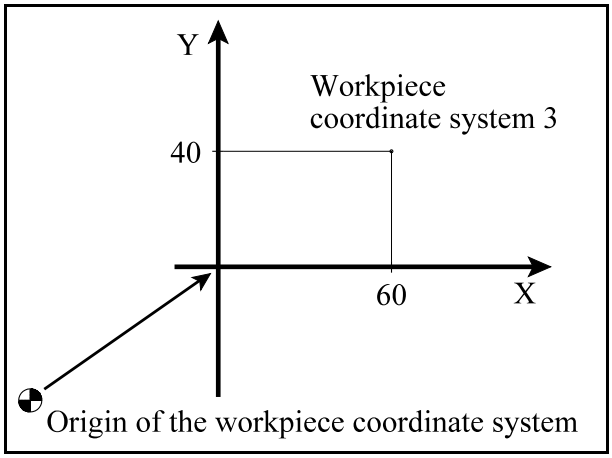


Fig. 10.2.1-3

After changing the coordinate system, the position of the tool will be displayed in the new coordinate system. For example, there are two workpieces on the work table. To the reference point of one of the workpieces, the first system, i.e. the workpiece coordinate system ***G54*** is positioned, the ***offset*** of which in the machine coordinate system is

X=300, Y=800.

To the reference point of the other workpiece, the second system, i.e. the workpiece coordinate system ***G55*** is positioned, the ***offset*** of which in the machine coordinate system is

X=1300, Y=400.

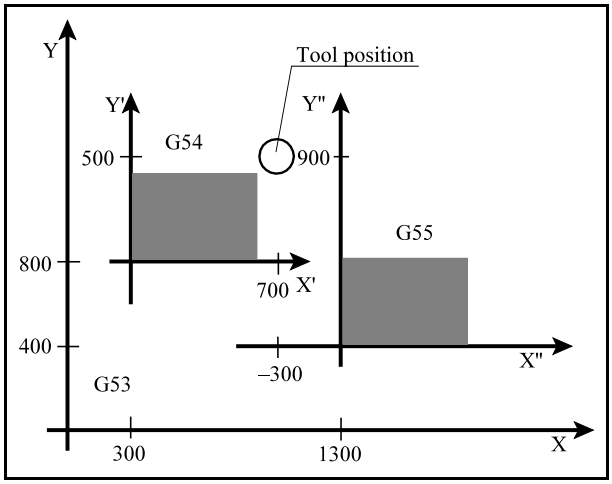


Fig. 10.2.1-4

The ***position of the tool*** in the coordinate system ***G54*** X', Z' is

X'=700, Y'=500.

By the effect of the instruction ***G55, the position of the tool*** in the coordinate system X", Z" will be interpreted as

X"=–300, Y"=900.

10.2.2 Selecting the Additional Workpiece Coordinate Systems (G54.1 P)

Optionally, 99 further workpiece coordinate systems can be used in the control. These coordinate systems are augmentations to the 6 basic coordinate systems so they are called additional workpiece coordinate system.

The ***common zero point offset*** shifts the additional workpiece coordinate system too. The additional workpiece coordinate systems ***can be rotated*** just as the basic coordinate systems can be.

The instruction

G54.1 Pp

is used for selection of a additional workpiece coordinate system, where the number of the additional workpiece coordinate system can be specified at the address P:

P = 1, 2, ..., 99

It is a modal function.

☞ ***Warning!*** The address P can serve various purposes. Therefore, the use of the address P in the block must be unequivocal:

```
G0 G54.1 P16 X100 Z20 M98 P1 (NOT CORRECT! Two Ps in one block)
G0 G54.1 P16 X100 Z20 (unequivocal)
```

10.2.3 Compensating the Angular Position of the Workpiece

Compensation of the angular position of the workpiece is needed when the workpiece cannot be aligned parallel to the main axes. Hereafter, this compensation is called ***misalignment compensation***.

Interpretation of misalignment compensation

In the general zero point offset table below, the columns have the following meanings:

X, Y, Z: the zero point offset along the 3 main axes,
α, β, γ: the angles of the misalignment compensation, namely, α in the plane G19, β in the plane G18, γ in the plane G17.

	X	Y	Z	G17(γ)	G18(β)	G19(α)
G54						
...						
G54.1 P						
...						

The rotations are always about the main axes. The order of rotations is as follows:
Rotation 1: rotation about X axis at an angle α,
Rotation 2: rotation about Y axis at an angle β,
Rotation 3: rotation about Z axis at an angle γ.

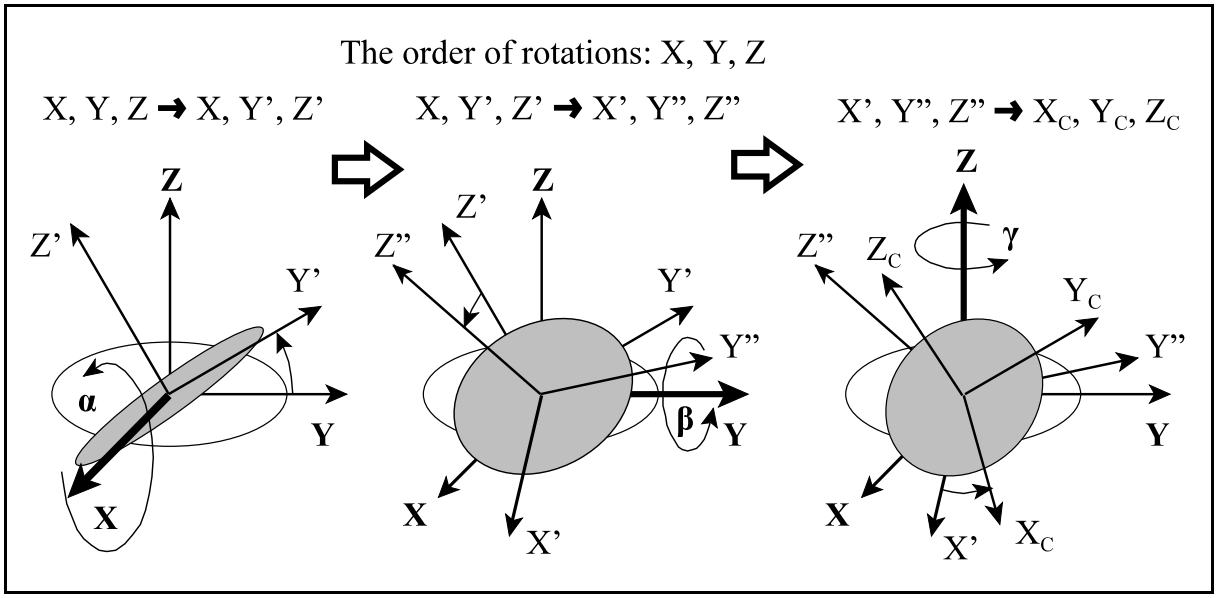


Fig. 10.2.3-1

The direction of α, ß and γ is always the direction of the rotational transformations about the corresponding main axis, according to the right-hand rule.

The misalignment compensation transformations ***are always taken into account in the current workpiece coordinate system, in the case of motion commands***.
It never applies to commands on motion to a machine position (G53, G28, G30)..

In the case of manual moving, the controller takes into account the switch 'In accordance with misalignment compensation' (CP_WMCAXF PLC flag), i.e., it applies rotation about all three axes.

If the endpoint programmed is **v**[x, y, z], rotation takes place in accordance with the equation

$$\mathbf{v'}=\mathbf{M_Z}(\gamma)*\mathbf{M_Y}(\beta)*\mathbf{M_X}(\alpha)*v$$

where:

v'[x', y',z'] are the coordinates rotated, and

$\mathbf{M_X}(\alpha)$, $\mathbf{M_Y}(\beta)$, $\mathbf{M_Z}(\gamma)$ are the matrices of rotations about the X, Y and Z axes:

$$M_X = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix} \quad M_Y = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \quad M_Z = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Applying misalignment compensation for 3-axis machines

In the case, when the misalignment compensation takes place only about the axis of the tool (about the Z axis in figure below):

– The position of the tool endpoint will be correct, and the hole will be parallel with the Z axis,

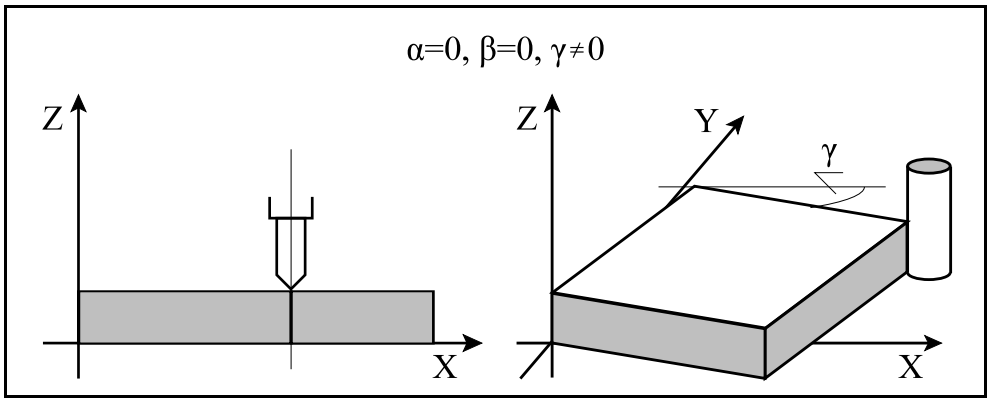


Fig. 10.2.3-2

– When milling a side, the tool mills the side perpendicular to the XY plane.

In the case, when the misalignment compensation takes place about the axis perpendicular to the tool (about the Y axis in figure below):

– The position of the tool endpoint will be correct, but the hole will not be perpendicular to the surface of the workpiece,

– When milling a side, the tool mills the side perpendicular to the XY plane, i.e., the side of the workpiece will not be perpendicular to the upper plane of the workpiece.

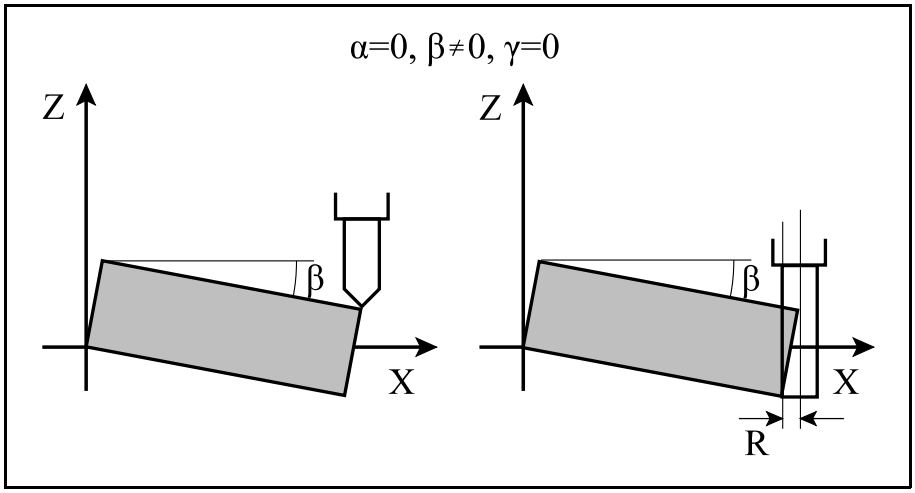


Fig. 10.2.3-3

10.2.4 Setting the Offset of the Workpiece Coordinate Systems (G10 L2)

Offset, rotation and common zero point offset of the workpiece coordinate systems can also be set using program instruction.
Setting is executed by the instruction

G10 L2 P v I J K,

where

P = 0	setting the common zero point offset;
P = 1...6	selecting the workpiece coordinate system G54, ..., G59;
v (X, Y, Z, ...):	offset values for the axes.

Axis offsets are always entered as Cartesian values; length data in mm or in inch, angle data in degree.

I:	γ the angle of rotation in the plane G17;
J:	β the angle of rotation in the plane G18;
K:	α the angle of rotation in the plane G19.

The angle of rotation must always be specified in degree.
For common zero point offset, rotation cannot be specified.
G10 is a one-shot instruction.
In the absolute data setting instruction state **G90,** the value written at the coordinate addresses or at the address I, J and K will be put in the appropriate offset register.
In the incremental data setting instruction state **G91** or in the case of using the operator I, the data written at the addresses will be added to the content of the appropriate offset register. The operator I can be used for coordinate addresses only, but not for the address I, J and K.

10.2.5 Setting the Offset of the Additional Workpiece Coordinate Systems (G10 L20)

Offset and rotation of the additional workpiece coordinate systems can also be set using program instruction.
Setting is executed by the instruction

G10 L20 P v I J K,

where

P = 1...n — selecting the workpiece coordinate system G54.1 P1, G54.1 P2, ..., G54.1 Pn;

v (X, Y, Z, ...): — offset values for the axes.

Axis offsets are always entered as Cartesian values; length data in mm or in inch, angle data in degree.

I: — γ the angle of rotation in the plane G17;

J: — β the angle of rotation in the plane G18;

K: — α the angle of rotation in the plane G19.

The angle of rotation must always be specified in degree.

G10 is a one-shot instruction.

In the absolute data setting instruction state **G90,** the value written at the coordinate addresses or at the address I, J and K will be put in the appropriate offset register.

In the incremental data setting instruction state **G91** or in the case of using the operator I, the data written at the addresses will be added to the content of the appropriate offset register. The operator I can be used for coordinate addresses only, but not for the address I, J and K.

10.2.6 Creating a New Work Coordinate System (G92)

A The instruction

G92 v

establishes a new workpiece coordinate system in such a way that a designated point, e.g. the tool tip will be the point of coordinate v of the new workpiece coordinate system. Afterwards, any following absolute instruction will have to be interpreted in this new workpiece coordinate system, and positions will also be displayed in this coordinate system. The coordinates specified in the instruction G92 will always be interpreted as absolute Cartesian values.

For example, if the tool is at the point of the coordinates

X=150, Y=100

in the actual X,Z coordinate system, the instruction

```
G92 X90 Y60
```

creates the new X',Y' coordinate system in which the tool will be at the point of the coordinates

X'=90, Y'=60.

The axis components of the offset vector **v'** between the coordinate systems X,Y and X',Z' will be the following:

v'_x=150-90=60, and

v'_y=100–60=40.

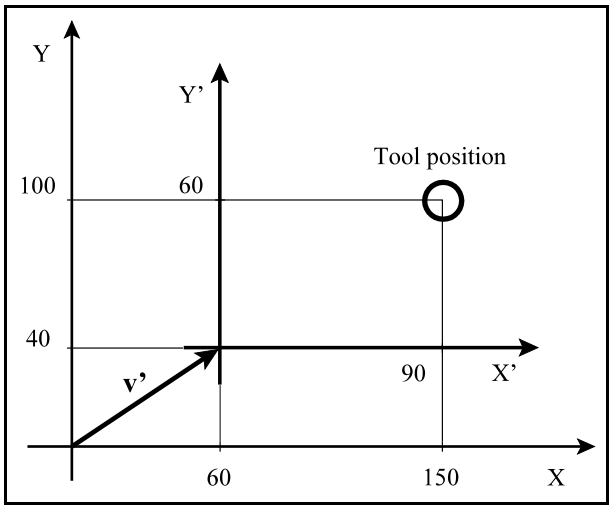


Fig. 10.2.6-4

The instruction G92 is valid in each workpiece coordinate system, i.e. the offset **v** calculated in one of them will be taken into account in the other ones too.

The offset of the workpiece coordinate system set by the instruction G92 will be cancelled by power-on, at the end of the program and by reset.
The instruction ***G92 cancels the tool radius compensation vector***, it will not be included in calculation of the offset.
The instruction ***does not cancel the length compensation vector***, the offset will always be calculated for the position of the tool tip.
The instruction ***G92 cancels*** the local coordinate system's ***offsets*** programmed by the instruction ***G52*** on the axes that are given in the instruction.
During motion, if the valid workpiece coordinate system is rotated, the offset vector specified in the instruction G92 will be taken into account as rotated vector.

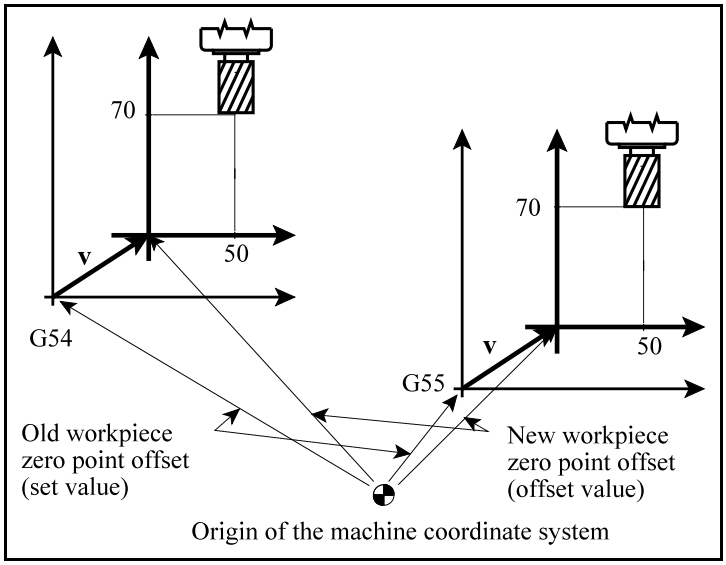


Fig. 10.2.6-5

10.3 Local Coordinate System (G52)

When writing part programs, it is sometimes more convenient to specify the coordinate data in a so-called local coordinate system instead of the work coordinate system.
The instruction

G52 v

creates a local coordinate system.
If the coordinate **v** is specified **as absolute value**, the origin of the local coordinate system will coincide with the workpiece coordinate system's point of coordinate **v**.
If the coordinate **v** is specified **as incremental value**, the origin of the local coordinate system will be shifted with the offset **v**.
Afterwards, any following motion instruction specified with absolute coordinates will be executed in this new coordinate system. Positions will also be displayed in this new coordinate system.
The values of the coordinates ***v*** will always be handled as ***Cartesian*** data.
The instruction

G90 G52 v0

cancels the offsets at the point of coordinate v.
On power-on, at the end of the program and on reset the offset values set by the instruction G52 will be cancelled.

If the tool is at the point of the coordinates

X=150, Y=100

in the actual X,Z workpiece coordinate system, the instruction

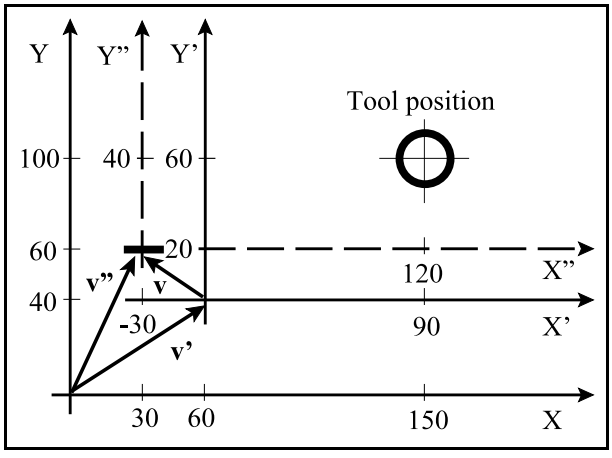


Fig. 10.3-1

```
G90 G52 X60 Y40
```

creates the new X',Y' coordinate system in which the tool will be at the point of the coordinates X'=90, Y'=60

The axis components of the offset vector **v'** between the coordinate systems X,Y and X',Y' will be the following by the instruction G52: v'_x=60, and v'_y=40.

A new coordinate system X",Y" can be created in the following two ways:

By ***absolute*** data specification:

The instruction

```
G90 G52 X30 Y60
```

moves the origin of the coordinate system X",Y" to the point of coordinates X=30, Y=60 ***in the workpiece coordinate system X,Y.*** The components of the vector ***v"*** will be produced by the specification of $v"_x$=30, $v"_y$=60.

By ***incremental*** data specification:

The instruction

```
G91 G52 X-30 Y20
```

shifts the origin of the local coordinate system X',Y' by the values X'=-30, Y'=20. The components of the vector ***v*** will be produced by the specification of v_x=-30, v_y=20. The vector v" showing the position of the new local coordinate system in the workpiece coordinate system X,Y is **v"**=**v'**+**v**. Its components are the following: $v"_x$=60+(-30)=30, $v"_y$=40+20=60.

The position of the tool in the coordinate system X",Y": X"=120, Y"=40.

The offset of the local coordinate system is valid in all the workpiece coordinate systems.

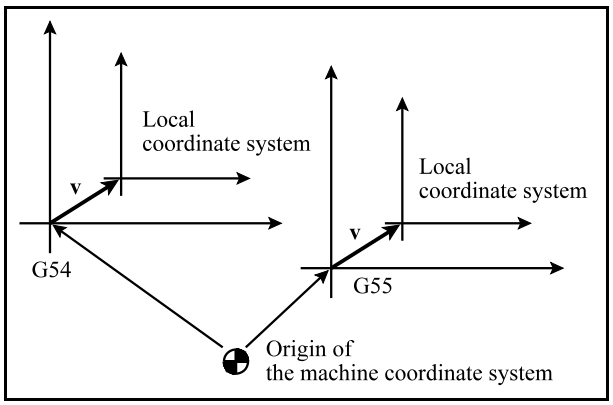


Fig. 10.3-2

Programming the instruction ***G92*** on the axes for which values were specified, ***cancels*** the offsets created by the instruction ***G52***, as if the instruction G52 v0 would have been issued.

If the tool is at the point of coordinates X=200, Y=120 of the workpiece coordinate system X,Y, by the instruction

```
G52 X60 Y40
```

its position in the local coordinate system X',Y' will be X'=140, Y'=80. Then, by the instruction

```
G92 X110 Y40
```

the position of the tool in the new workpiece coordinate system X",Y" will be X"=110, Y"=40. So, the instruction G92 deletes the

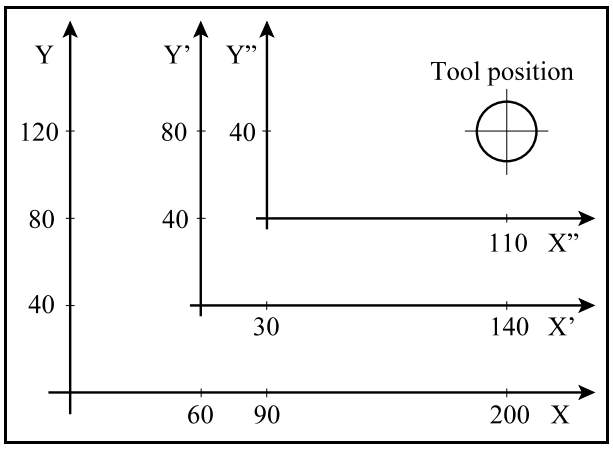


Fig. 10.3-3

local coordinate system X',Y' as if the instruction G52 X0 Y0 would have been issued.

10.4 Plane Selection (G17, G18, G19)

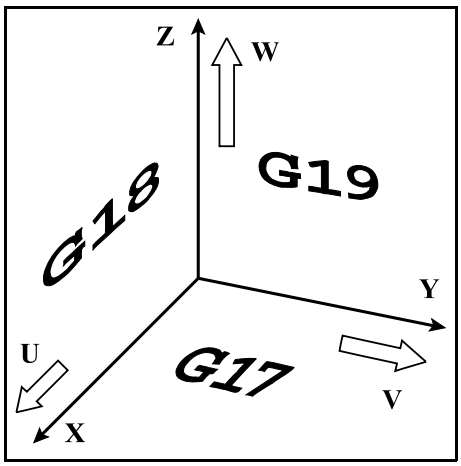


Fig. 10.4-1

The plane in which
– circular interpolation,
– data specification using polar coordinates,
– planar rotation of the coordinate system,
– planar tool radius compensation,
– positionings of drilling cycles,
– turning cycles
are performed, can be selected using the following codes G:

G17 the plane X_pY_p
G18 the plane Z_pX_p
G19 the plane Y_pZ_p,

where: X_p: the axis X, or the axis parallel with it;
Y_p: the axis Y, or the axis parallel with it;
Z_p: the axis Z, or the axis parallel with it.

The plane selected is called main plane.
It depends on axis addresses programmed together with the instruction G17, G18 or G19 in one block, which one of the parallel axes will be selected:
For example, if X and U, Y and V, Z and workpiece are parallel axes, then:

the axis XY will be selected by the instruction G17 X____ Y____;
the axis XV will be selected by the instruction G17 X____ V____;
the axis UV will be selected by the instruction G17 U____ V____;
the axis XW will be selected by the instruction G18 X____ W____;
the axis YZ will be selected by the instruction G19 Y____ Z____;
the axis VZ will be selected by the instruction G19 V____ Z____.

If G17, G18, G19 is not given in a block, the plane selection will remain unchanged:
G17 X____ Y____ the plane XY
U____ Y____ the plane remain the plane XY.

If axis address is not given in the block G17, G18 and G19, the control will select the main axes:

G17 selects the plane XY;
G17 X selects the plane XY;
G17 U selects the plane UY;
G17 V selects the plane XV;
G18 selects the plane ZX;
G18 W selects the plane WX.

The motion instruction does not affect the plane selection; due to the instruction

```
G90 G17 G00 Z100
```

the plane XY will be selected and the axis Z moves to the point of coordinate 100.
Changing planes within a program can be done more than once.

After power-on, program end or reset, it will be decided by the bits #1 G18 and #2 G19 of the parameter N1300 DefaultG1 which plane will be valid.
It can be set in the parameter N0103 Axis to Plane which axis address will be used by the control

for main and parallel axes.
In this manual there are many references to the first axis and the second axis of the selected plane. The figure below illustrates how to interpret them.

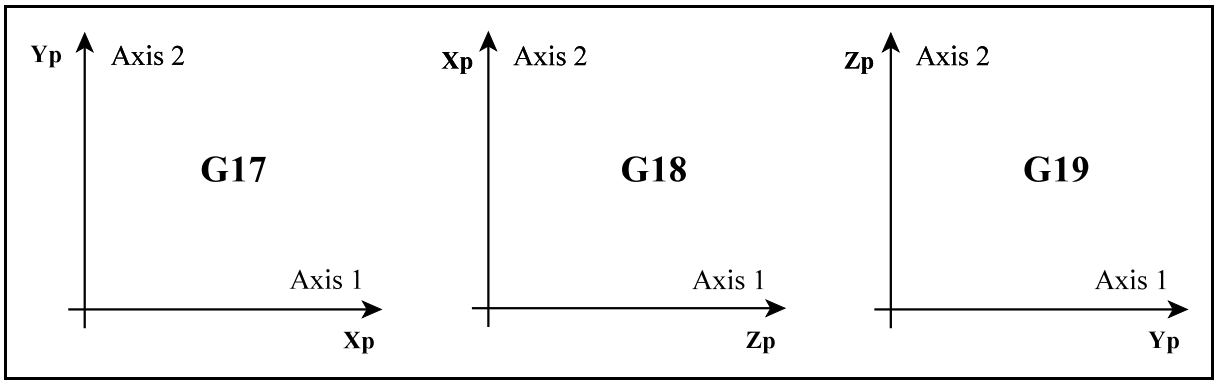


Fig. 10.4-2

11 Spindle Functions

11.1 Spindle Speed Command (Code S)

With writing a maximum 8-digit number at the address

S nnnnnnnn

the code will be transmitted by the control to the PLC.
Depending on the design of the given machine tool, the PLC can interpret the address S either as a code, or as a value with the dimension of revolution/minute.
If motion instruction and spindle speed (S) are programmed in the same block, the function S will be executed during or after carrying out the motion instruction. The builder of the machine determines the manner of execution.
The speed values specified at the address S are modal ones. After power-on the control starts with the code S0.
In each gear ratio range, the spindle has a minimum and a maximum limit. These limits are determined by the builder of the machine too in the parameter field, and the control does not allow the speed to come out from the range.
The control can manage maximum 8 speed range.

11.1.1 Referring to Several Spindles. Extending the Address S

The control can manage maximum 16 spindles.
If there are several spindles on a machine or in one channel, the address S will not be enough to discriminate the spindles. The control provides two ways for managing several spindles.

Referring to a Spindle at Addresses S and P

The first way is to give the spindle number at the address P in addition to the address S. By the instruction

Snnnnnnnn Pp

the code S and the spindle number written at the address P will be transmitted by the control to the PLC.
The program language uses the address P for various purposes, for example for waiting, for called subprogram etc. For this reason, the spindle references at the address P must be written in a separate block, otherwise interpretation of the address P will not be unambiguous.

Referring Several Spindles by Extension of the Address S

The another way is to extend the addresses of the spindles. The spindles can also be referred by specifying maximum 3 characters.
The addresses of the spindles must always begin with the letter S. In the parameters N0605 Spindle Name2 and N0606 Spindle Name3, two additional characters may be given which can be letters of the English alphabet: A, B, C, D ... Y, Z or numbers: 0, 1, 2 ... 9. If the spindle name 2 or the spindle name 3 is not used, the value of the parameters will be 0.
Accordingly, the name of a spindle can be SSB, but S1 and S2 may also be used as spindle name.
If the name of a spindle ends with letter, the value related to it can be added. The meaning of

```
SSB12500
```

is: the spindle SSB has to rotate at the speed of 12500/min.
If the name of a spindle ends with number, the sign = will have to be written after the name. The

meaning of

```
S1=8700
```

is: the spindle S1has to rotate at the speed of 8700/min.
In the case of referring with an extended spindle address, the speed value programmed on the address and the number of the referred spindle will be transmitted by the control to the PLC.
In the program, ***only one spindle should be referred within one block***. If several spindles have to be started, the instructions must be written in separate blocks:

```
S1=500 M3 (S1 500/min, clockwise)
S2=1000 M4 (S2 1000/min, counter clockwise)
```

The control always stores the spindles according to their number. ***Numbering and naming the spindles is also global*** and channel-independent.

11.1.2 Assigning Spindles to Channels

The given ***spindles are always assigned to the given channels by the PLC program***.
Assignment means that speed instruction for a given spindle can only be issued from the program running in that given channel. For example, if the spindle S4 is assigned to the channel 2, the address S4 cannot be programmed in the channel 1.
During program run, the PLC program can move a given spindle into an other channel due to M function, for example.
Assigning spindles to channels and moving them into other channels is always determined by the builder of the machine tool.
In each channel, a default spindle can be designated in the parameter N0604 Default Spindle. This designated spindle can always be referred at address S even if it has multicharacter name. For example, let the S2 be the spindle No.2. If in the channel 2 N0604 Default Spindle=2, then the spindle can be referred at both addresses S2 and S in the channel 2.

11.2 Functions M Controlling the Spindle

Built-in Functions M

The control manages the spindles using the following built-in codes M:

M3: Spindle On, clockwise
M4: Spindle On, counter-clockwise
M5: Spindle Off (Stop)
M19: Orientation

The direction is always to be meant from the motor towards the spindle. M03, M04 and M05 can also be written instead of M3, M4 and M5.
These are built-in codes M controlling the spindle because they are transmitted by the control for the PLC to stop the spindle, to change its direction of rotation and to orient it during execution of drilling cycles.

Optional Functions M

In addition to the abovementioned built-in codes M, further codes M can also be designated to manage the spindles. These codes can be set in the parameters N0689 Spindle M Low and N0690 Spindle M High, in one array. The smallest value of the array of the codes M has to be written in the parameter N0689 Spindle M Low, but the greatest value has to be written in the parameter N0690 Spindle M High.
For example, let Spindle M Low be S2=20, and Spindle M High S2=24. Let the functions of the

codes M be the following:

M20: Conversion of the spindle into axis C
M21: Synchronization of the spindle
M22: Synchronization of the spindle together with phase shift
M23: Preparation of the spindle for polygonal turning
M24: Closing the spindle position loop without orientation (Code M for Closing S Loop)

☞ ***Warning!*** *The abovementioned array of the codes M is an example only. The optional functions M of the spindles are always determined by the builder of the machine tool, and for this reason, description of them can be found in the manual of the given machine.*

The optional functions M designated in parameter and the built-in ones are registered by the control as exclusive functions. It means that only one such code M can be written in one block.

The functions M controlling the spindle refer always to the spindle programmed last:

```
S1=1500 M4      (S1 On in M4 direction)
S2=2000 M3      (S2 On in M3 direction)
M5              (S2 Stop)
M19             (S2 Orientation)
S1=0 M5         (S1 Stop)
```

The example above shows, that if the spindle S2 was referred already, the further codes M controlling the spindle will refer to the spindle S2.If, however, the spindle S1should be stopped, it is needed to refer to the address of the spindle.

11.3 Managing the Speed Ranges

There could be a ***variable-ratio drive*** between the driving motor and the spindle which can be used ***for changing the spindle speed range***. Maximum 8 speed ranges for each spindle can be managed by the control. The lower the speed range the spindle is in, the higher the torque will be with which it is able to machining.
For each ratio range, there can be set permissible minimum and maximum speeds, below and above of which the control does not allow the spindle speed to decrease and increase, respectively.
The permissible speeds can overlap each other between the ranges.

The case when speed ranges do not overlap each other

For example:

The minimum speed of the range 1: 50/min
The maximum speed of the range 1: 1000/min
The minimum speed of the range 2: 1001/min
The maximum speed of the range 2: 4000/min

In the case above, according to programming the code S900 it is unambiguous that the spindle has to be rotated in the range 1.
If the ranges do not overlap each other, in addition to the value of ***the address S*** and the number of the referred spindle, the control will transmit the code of the range too to the PLC, and the PLC ***will change*** the required range ***automatically***.

The case when speed ranges overlap each other

For example:

The minimum speed of the range 1: 50/min
The maximum speed of the range 1: 1000/min
The minimum speed of the range 2: 800/min
The maximum speed of the range 2: 4000/min

According to programming the code S900 it is not unambiguous to decide which is the range the spindle has to be rotated in, whether the range 1 or the range 2.
If the ranges overlap each other, the programmer will have ***to use M function to select*** the range in which he would like to rotate the spindle.
These functions M are the following:

M11: Selecting the range 1
M12: Selecting the range 2
...
M18: Selecting the range 8

Managing the ranges on a given machine is determined by the builder of the machine tool, and description of it can be found in the manual of the given machine.

11.4 Main Spindle. Selecting the Main Spindle

If there are several spindles on a machine tool or in a channel, it will have to be decided which one of them will be the 'main' spindle. The following functions apply to the spindle designated as ***main spindle***:

enabling the feed;
feed per revolution;
calculation of constant cutting speed;
threading;
rigid tapping;
master spindle of polygonal turning.

The main spindle has to be designated in each channel. In a given channel, a spindle related to an other channel can also be designated as main spindle, for this reason it cannot be programmed in this channel, but the feed per revolution has to be received from this spindle, for example.
The way of selection of the main spindle is determined by the PLC program of the given machine tool, and it is contained ***in the manual provided by the builder of the machine tool***.
Selection can be done using function M, for example:

```
M31 (the spindle 1 acts as main spindle)
M32 (the spindle 2 acts as main spindle)
```

11.5 Controlling the Constant Surface Speed

Function of controlling the constant surface speed can be used for infinitely variable spindle drive only. In this case, the control changes the speed of the spindle in such a way that the tool speed relative to the workpiece surface is always constant and equal to the value programmed.

The control changes always the speed of the spindle designated as main spindle.

The value of the constant surface speed as a function of the input unit system has to be done on the basis of the table below:

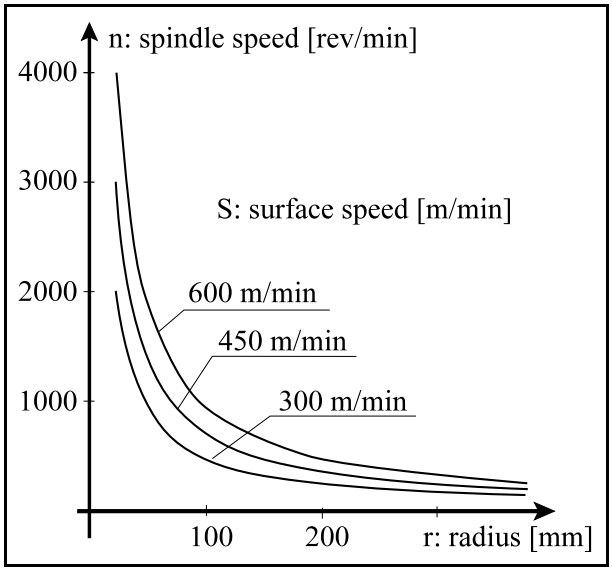


Fig. 11.5-1

Input unit	Unit of the constant surface speed
mm (G21 metric)	m/min
inch (G20 inches)	feet/min

11.5.1 Specifying the Constant Surface Speed Control (G96 S, G97 S)

The instruction

G96 S

switches the constant surface speed control on. At the address S the value of the constant surface speed control has to be given in the unit shown in the table above.

The value of the constant surface speed control must be given always at the address S, and specification of the speed applies to the spindle designated as main spindle.

For example:

```
M32 (designation of the spindle 2 as main spindle)
G96 S300 (the surface speed is 300 m/min)
```

(M32 here is an example only, the way of designation as main spindle is found in the manual on the given machine tool.)

The instruction

G97 S

switches the constant surface speed control off .

At the address S, the desired spindle speed can be specified (in rev/min). In the case when there are several axes, after programming G97 always the address of the main spindle has to be used, instead of S. For example:

```
G97 S2=1200 (The speed of the spindle 2 is 1200/min)
```

– For calculation of constant surface speed, the zero point of the axis, on the basis of the position of which the speed of the main spindle has to be changed, must be set on the axis of rotation of the main spindle.
– The constant surface speed control is effective only after starting the main spindle by M3 or M4.
– The value of the constant surface speed is modal even after its calculation has been cancelled by the instruction G97.

```
G96 S100 M3     (100 m/min or 100 feet/min)
G97 S1500       (1500 rev/min)
G96 X260        (100 m/min or 100 feet/min)
```

– The constant surface speed calculation is effective in mode G94 (feed/min) too.
– If the constant surface speed calculation is cancelled by the instruction G97 and a new speed of the main spindle is not specified, the last speed of the main spindle taken in the state G96 will remain effective.

```
G96 S100  (100 m/min or 100 feet/min)
      .
      .
      .
G97       (Speed related to the resulted diameter X)
```

– In the case of rapid traverse positioning (block G0), the spindle speed is not calculated continuously, but the control sets the speed related to the position actual in the endpoint of the positioning.
– After power-on, the parameter N0686 Default Surf Speed determines the value of the constant surface speed.

11.5.2 Clamping the Speed during Calculation of Constant Surface Speed (G92 S)

The instruction

G92 S

is used for setting the highest speed of the main spindle permissible during constant surface speed control. When the constant surface speed control is switched on, the control disables issuing a main spindle speed greater than the value specified here. In this case, the unit of the S is rev/min. ***The maximum value of the main spindle speed must be given at the address S***, and the speed clamp applies to the spindle designated as main spindle.

– After power-on or if the value of the speed is not clamped by the instruction G92, in the case of constant surface speed control, the maximum value permissible in the given range will be the upper limit of the main spindle speed.
– In the case of constant surface speed control, a lower limit can also be given for the main spindle speed in the parameter N0688 Min Spindle Speed G96, which can be greater than the minimum value of the speed related to the range.
– The value of the maximum speed is modal until a new value is programmed.

11.5.3 Selecting an Axis for Constant Surface Speed Control (G96 P)

In the state G96, the parameter N0687 Default G96 Axis selects the axis, according to the position of which the control calculates the spindle speed.
If an other axis is to be used, the axis according to which the surface speed should be calculated can be specified by the instruction

G96 P.

The ***address P*** is to be interpreted as ***number of axis***.
In the instruction G96, address S can also be programmed together with address P:

```
G96 S300 P4 (surface speed 300 m/min together with the axis 4)
```

The value set at the address P is modal.

11.6 Spindle Speed Fluctuation Detection

The control monitors speed fluctuation of each spindle. It determines the fluctuation as difference between the programmed speed modified by override and by speed limits, and the actual speed measured from signal transmitter.
If the speed of the spindle is out of the tolerance range set in parameter by the builder of the machine tool, the control will send a message to the PLC.
Then, the PLC program sends an error message and takes action to stop the spindle and machining. All this is descripted in the manual provided by the builder of the machine tool.

– The function of speed fluctuation detection will work in the only case if the spindle is equipped with signal transmitter.
– The spindle fluctuation detection is only effective when the spindle rotates (in the state M3 or M4).

11.7 Positioning the Spindles

In case of normal machining, the control issues speed instructions proportional to the programmed speed to the spindle drives. At this time, the spindle drive works in speed control mode.
In cases of certain technological tasks it could be necessary to bring a spindle into a defined angular position. This process is called spindle positioning or indexing.
Prior to positioning, the control switches the spindle in the mode of position control. Practically, it means that henceforth the control does not issue speed instruction proportional to the code S, but it measures the spindle position by the use of angular position transmitter (signal transmitter) mounted on the spindle and it issues instruction depending on desired angular displacement, as it is done on the other position-controlled axes. This is the position feedback.
In order that the spindle on a given machine tool can be positioned, an encoder has to be mounted on the spindle, and the spindle drive should have the design providing operation in the mode of position feedback too.

11.7.1 Spindle Orientation

The function of stopping the spindle in a particular angular position is called spindle orientation or oriented spindle stop. It could be necessary in the case of automatic tool change or for carrying out certain drilling cycles, for example.
The bit state #1 ORI=1of the parameter N0607 Spindle Config means that it is possible to orient a given spindle.
The orientation instruction is issued by the function

M19.

If there are several spindles on the machine, in addition to the function M19, the spindle also has to be selected. For example:

```
S2=0 M19
```

Technically, orientation can be done in the following two different ways.

If the spindle cannot be fed back in position control (parameter state of N0607 Spindle Config #2 INX=0), orientation can be carried out by turning on the position switch mounted on the machine.
If the spindle can be fed back in position control (parameter state of N0607 Spindle Config #2 INX=1), the instruction M19 will cause the control to seek zero pulse of the spindle encoder. Then, the control closes the position control loop automatically.
This, at the same time, means the reference point return of the spindle too, i.e. the spindle can be sent to an absolute angular position after orientation

11.7.2 Stopping the Spindles and Closing the Position Control Loop

The parameter state N0607 Spindle Config #2 INX=1means that the position control loop can be closed.
In this case, in the parameter N0823 M Code for Closing S Loop there can be given a code M due to which the spindle stops, closes the position control loop, but does not go to the orientation position (does not seek the zero pulse of the encoder).
For example, if the value of the parameter is 24, closing the loop will occur due to the instruction

```
M24.
```

If there are several spindles on the machine, in addition to the function M, the spindle also has to be selected. For example:

```
S2=0 M24
```

It is the builder of the machine tool who can give information about the code and work of the function.
This function can accelerate execution of rigid tapping cycles, for example.

11.7.3 Programming of the Positioning the Spindles

The parameter state N0607 Spindle Config #2 INX=1means that the position control loop can be closed. This is the only case when positioning the spindles is possible.
The parameter will be set by the builder of the machine tool in the case if the function is implemented on the given spindle.

Positioning on the basis of axis name

Each spindle may have a maximum 3-character-long axis name which can be referred to after closing the position control loop.
In the parameter N0817 Spindle Axis Name1, it is obligatory to set the letter A, B or C. In the second character (parameter N0818 Spindle Axis Name2) and on the third character (parameter N0819 Spindle Axis Name3) there can be given letters A, B, C, G ...Y, Z of the English alphabet or numbers 0, 1, 2 ... ,9.
For example:

CS: the name of the spindle axis 1
CS2: the name of the spindle axis 2
ABC: the name of the spindle axis 3

If the name ends with number, the sign = will have to be used. Certainly, the name given should not coincide with other names.
After the address, ***position has to be given in degree***. Positioning is executed at the rapid traverse speed set for the spindle axis. The unit of the rapid traverse is 1/min.

Prior to the ***positioning*** by ***absolute*** data specification, M19 has to be programmed. For example:
The instruction lines

```
S2=0 M19
G90 CS2=30
```

moves the spindle CS2 to the position of 30 degree, rotating the spindle in positive direction.
In the case of absolute positioning, the control ***cuts the full turns*** from the data greater than 360 degree, for example, if

```
S2=0 M19
G90 CS2=750
```

are given, the control will rotate the spindle to the 30 degree. The control rotates ***always in the direction of the shorter travel***. If

```
S2=0 M19
G90 CS2=270
```

are given, the control will rotate to the 270 degree, in negative direction.

Prior to the ***positioning*** by ***incremental*** data specification, M19 has not to be programmed, it is enough to close the position loop only. If the loop is closed by the M24, then:
in the case of the following data specifications

```
S2=0 M24
G91 CS2=3600
```

or

```
S2=0 M24
CS2=I3600,
```

the spindle will make 10 revolutions in the given direction (here in positive direction). In the case of incremental data specification, the control ***does not cut the full turns and will rotate in the direction given by the sign***.

Indexing the Spindles using function M

Spindles can be indexed using codes M. Indexing means revolving the spindle to a discrete preset positions.
For this, the following parameters have to be set:
The direction of indexing at the bit #7 IDS of the parameter N0607 Spindle Config: =0 positive, =1 negative.
The initial value of the codes M in the parameter N0820 Start M of Spnd. Pos. and the the number of the codes M in the parameter N0821 No. of M Code for Spnd. Pos..
The angle of indexing in the parameter N0822 Basic Angle of Spnd. Pos..

An example:
The spindle can be fixed by 18 degrees starting from the orientation position. The setting is as follows:
Spindle Config: #7 IDS=0: indexing in positive direction
Start M of Spnd. Pos.=201 (m=201)
No. of M Code for Spnd. Pos.=20 (n=20)
Basic Angle of Spnd. Pos.=18 (φ=18)
The following table explains the different codes M:

Code M	The revolved angle α
Mm (M201)	α=φ=18°
M(m+1) (M202)	α=2φ=36°
M(m+2) (M203)	α=3φ=54°
....	
M(m+n) (M220)	α=nφ=360°

On the basis of the settings above, let's have the example below:

```
M19        (orientation)
M205       (revolving by 90°)
...        (drilling)
M210       (revolving by 180°)
...        (drilling)
```

Before issuing a code M for positioning, the spindle should be oriented by M19. Thus the spindle gets the position α=0.
A hole has to be drilled at the position 90°, the spindle revolves to the position α=5φ=5*18=90° by the instruction M205.
The next hole has to be drilled at the position 270°. When code M is used, the spindle can be moved only incrementally, for this reason M210 has to be programmed, because the displacement will be 10*18°=180° in this case.

11.7.4 Position-correct Synchronization of Two Spindles

Synchronization of two spindles means their rotation at the same speed and their being in a preset phase displacement to each other during rotation.
Position-correct synchronization of two spindles is done by the PLC program, using generally a code M. The method of synchronization is contained in the documentation provided by the builder of the machine tool.
For synchronization of two spindles, it is required of both spindles to be equipped with encoder, and that the position control looping can be realized.
During synchronization, ***the maximum speed of spindles*** is limited by ***the rapid traverse speed*** set for the spindle axis. Generally, this speed is lower than the maximum spindle speed.
In the course of synchronization, there are differentiated master and slave spindles. Always the slave spindle is synchronized to the master one.

The process of synchronization is as follows:
– if the speed of the master spindle is higher than the lower one of the rapid traverse speeds specified for the two (master and slave) spindles, the master one will slow down to the proper speed;
– the master spindle closes the position control loop;
– the slave spindle accelerates up to the master spindle speed in the same or opposite direction of the master spindle rotation;
– the slave spindle closes the position control loop;
– then, the slave spindle pulls its zero pulse over the zero pulse of the master spindle, or it executes a phase shift from the zero pulse of the master spindle on a distance given in

encoder pulse and specified in the parameter N0685 Spindle Phase Shift.

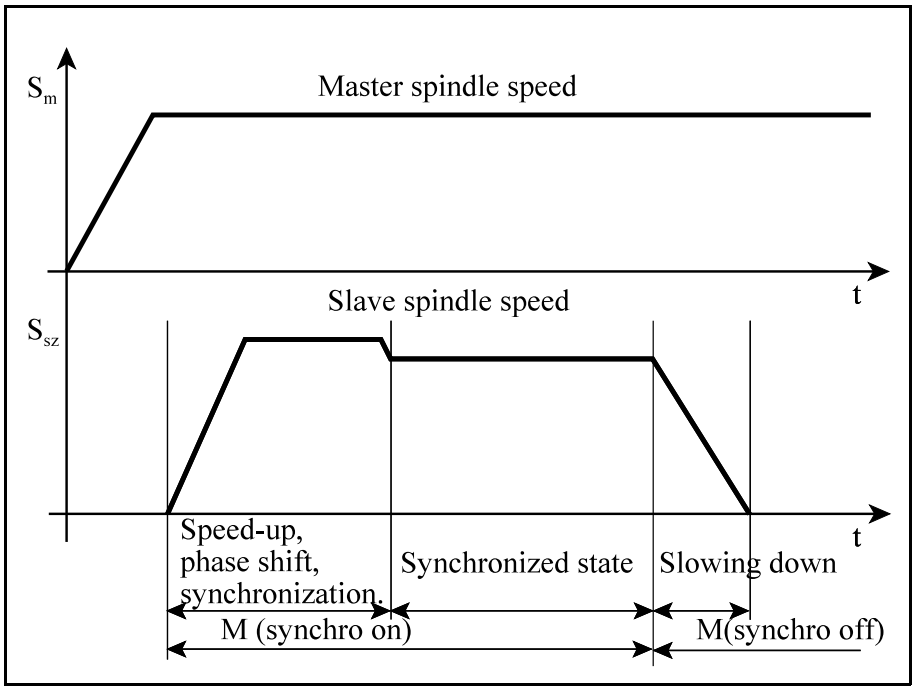


Fig. 11.7.4-1

An example:
It is required to machine the another side of a workpiece cut in the main spindle (marked as S1) of a sub-spindle lathe. The material the main spindle uses is bar. First, the sub-spindle being the slave spindle (marked as S2) should be synchronized to the main (master) spindle, the slave spindle should clamp the workpiece, and then cutoff should be executed.
Let M21 be synchronization realized by synchronous run of the zero pulses and let M22 be pulse shift synchronization.
If pulse shift is not necessary, the program detail will be the following:

```
S1=3000 M3
S2=0 M21   (synchronizing the S2 to S1)
...        (clamping the workpiece by the chuck of the S2)
...        (cutting off)
S2=1200 M4 (turning the synchronization off, rotating the S2)
```

If a shaped workpiece should be clamped, the code M22 will have to be used and phase shift will have to be set in the parameter N0685 Spindle Phase Shift.

☞ ***Warning!*** *The example above is a sample only. In a particular case, the process is as it is directed by the builder of the machine tool.*

11.7.5 Turning the Position-controlled Operating Mode off

After orientation, positioning or synchronization, the following function must be used for turning the spindles off from the position-controlled operating mode:

M3, M4 or M5

For example:

```
S1=0 M19   (position-controlled mode on, orientating the S1)
CS1=60
...
S1=0 M5    (position-controlled mode off, spindle stands)
```

or

```
S1=2400 (position-controlled mode off, spindle on)
```

11.8 Converting Spindle into Axis and Axis into Spindle

Spindle axes can be used for machining with limitations only because they can be positioned only, and they cannot be involved in interpolation with other axes. Furthermore, spindle axes use low-resolution and high-speed encoder, while a rotary table requires high-resolution encoder operating at low speed.
Therefore, in the course of machining it could be necessary to convert a spindle into axis or an axis into spindle. The spindle of a lathe has to be converted into axis C in order to mill on the face of the workpiece using polar coordinate interpolation or to engrave something onto its curved surface using cylindrical interpolation. Then, the axis C has to be reverted to spindle in order to execute further turning operations.
If the construction of the machine makes it possible, the rotary table B of a horizontal machining center can be converted into spindle in order to execute turning operation on the workpiece.

Appropriate mechanical and electronic construction of the machine tool is necessary to convert a spindle into axis or an axis into spindle. The manual of the given machine contains information about whether such conversion is possible on the machine. Always the PLC executes conversion in accordance with capabilities of the given machine.
Usually, the process of conversion of ***a spindle into axis*** is as follows:
– stopping the spindle if it rotates;
– stopping drive operation;
– disconnecting the spindle from the drive;
– converting the encoder into a high-resolution one;
– adjusting the drive parameters;
– turning the drive on again;
– connecting the axis to the drive and makes its display visible.
Usually, the process of conversion of ***an axis into spindle*** is as follows:
– waiting until the axis stops;
– stopping drive operation;
– disconnecting the axis from the drive and makes its display invisible;
– converting the encoder into a low-resolution one;
– adjusting the drive parameters;
– turning the drive on again;
– connecting the spindle to the drive.

An example:
On a lathe, the spindle S1 should be converted into the axis C1 for milling, and then, the axis C1 should be reverted to the spindle S1for further turning:

```
S1=0 M20        (converting S1 into C1)
G28 G91 C1=0    (reference point return on the axis C1)
...             (milling by the use of the axis C1)
S1=3000 M3      (reverting C1 to S1, rotation by 3000)
```

```
...                (turning by the use of S1)
```

☞ ***Warning!*** *The example above is a sample only. In a particular case, the process is as it is directed by the builder of the machine tool.*

12 Function T

12.1 Tool selection instruction (T kód)

In the part program, the tool number is referred by ***code T***. The code T should be given with a maximum ***8 decimal digit***:

Tnnnnnnnn

The leading zeros can be eliminated.
If motion instruction and tool number (T) is programmed in the same block, the function T will be executed during or after execution of the motion instruction. The builder of the machine determines the manner of execution.
The tool number will be given to the PLC program.

12.2 Programming the Tool Change

There are basically two ways of referring to tool change in the part program. These two ways depend on the machine construction. The tool call technique applicable in the part program is defined by the builder of the machine tool.

Case A: Tool change on code T

Tool change on the machine can be carried out manually or using turret type tool changer. In such a case, referring to the code

Tnnnnnnnn :

– in the case of manual change, in the screen there will be displayed the number of the tool to be changed; this tool has to be inserted into the spindle and after start the machining should continue;
– in the case of turret-type tool changer, due to the code T the machine changes the tool automatically.

Practically, referring to a tool number causes prompt tool change in the block in which T was specified.
In the case of tool change on code T, the following parameter settings are required:

Bit position #0 M06=0 of the parameter N1338 Block No Search - for block search
Bit position #1 TLC=1 of the parameter N2901 Search Config - for tool life management

Case B: Tool change on the function M6

Tool preparation on the machine is needed for tool change. The steps of preparation are as follows:

– Finding the tool to be changed in the tool magazine. In this step, referring to the address
Tnnnnnnnn
in the part program triggers motion of the appropriate tool to the change position. This action goes on in the background, parallel with the machining.
– Sending the slides to the change position.
– Executing the tool change by the function
M6
in the program. (M06 can also be used.) The control waits for executing the tool change until the tool T being under preparation arrives at the change position. Then it inserts the new tool into the tool holder. From this point, machining can be continued.

– Inserting the previous tool back into the tool magazine. This action goes on in the background, parallel with the machining.
– Beginning to find the new tool in the tool magazine.

In the case of tool change on the function M6, the following parameter settings are required:
Bit position #0 M06=1 of the parameter N1338 Block No Search - for block search
Bit position #1 TLC=0 of the parameter N2901 Search Config - for tool life management

An example:

```
T12 M6     (changing the tool T12)
T15        (calling the tool T15, the PLC searches in the
           course of machining)
...        (machining by the use of the tool T12)
M6         (changing the tool T15)
T8         (calling the tool T8, the PLC searches in the
           course of machining)
...        (machining by the use of the tool T15)
T9         (calling the tool T9, the PLC searches in the
           course of machining)
M6         (changing the tool T8)
...        (machining by the use of the tool T8)
```

If instruction T and instruction M6 are written in one block, the PLC will usually execute function T at first, and then function M6, i.e. it will execute tool change. After tool change, M6, it is practical to program tool call in the next block in order to minimize secondary time of the machine.

☞ ***Warning!*** *The example above is a sample only. In a particular case, the process is as it is directed by the builder of the machine tool.*

12.3 Tool Management

Tool management function has to be set in the control in the following cases:
– when tool life analysis should be applied;
– when referring to the tools stored in the magazine from the part program is done not according to position code, but according to tool code;
– when tool change applied on the machine requires random magazine handling. Random magazine handling is the case when the tool being in the spindle will be inserted not in the pocket from where it was taken out, but in the pocket of the incoming tool.

Tool management data are organized into ***tables*** stored in the control. ***In case of power-off***, data in the tables are ***reserved.***

12.3.1 Tool Management Table

The tool management table is a chart into which there can be written the ***code T***, i.e. the ***type number*** of the tools used in the control; this code T or type number will be referred to in the part program.
In this table there can be specified characteristics of the tools, in particular there size (normal or extra), their participation in tool life analysis (yes or no), and the element of the compensation memory assigned to them. Further technological data, e.g. speed, feedrate etc. may also be assigned to a given tool.
The tool management table is global, i.e. it is common for each channel.

The element of the tool management table:

Data number	Type number (T)	Tool info	Figure number	Tool life status	Tool life counter	Tool life
1	10002001	UENCV		Expired	150	150
2	10002001	SENCV		Used	131	200
3	10002001	SENCV		Not yet used	0	170
4	150	SDLCV	2	Not		
5	3210	UENTV		Broken	1h42m26s	2h30m00s
6	3210	SENTV		Used	1h34m14s	1h50m30s
...						

Information about the elements of the column of the tool management table is summarized below. The memory area required by the elements is indicated next to the elements.

1 Data number (DWORD)

Serial number in the table. It cannot be edited, the number of the table row is defined by parameter. It is the serial number of the tool management table, i.e. ***the data number on the basis of which tools are registered*** in the cartridge table.

2 Type number (DWORD)

A tool can be referred to by the Type number at the address T from the part program. ***If a group of the tools is involved in tool life analysis, all the tools of the group have to be marked with the same Type number (code T)***. The Type number can be given by maximum 8 digits:

T: From 1 up to 99 999 999.

In the course of calling for tool (code T), there will always be selected the data-numbered tool the tool life counter of which shows the greatest value, but not expired yet. According to the table above, referring to

```
T10002001
```

the tool with data number 2 will be selected. If the tool life counter of all the tools with the same type number shows the same value, the tool with the smaller data number will be selected.

3 Tool info (DWORD)

The Tool info contains the following bit information:

#0 **I** (=0, Invalid): An entire row of the tool management table is invalid, it can be deleted.
V (=1, Valid): An entire row of the tool management table is valid.

#1 **C** (=0, Count): Tool life analysis is executed for use.
T (=1, Time): Tool life analysis is executed for time.

☞ ***Warning!*** *For the tools with the same Type number, the same Tool info C or T must be set.*

#2 **N** (=0, Normal): The tool is of normal size (it occupies room of one tool).
L (=1, Large): The tool is of large size (it occupies room of several tools).

#3 **E** (=0, Enabled): An entire row of the tool management table is editable.
D (=1, Disabled): An entire row of the tool management table is not editable.

#4 If Tool life status says that the given tool is not involved in the tool life analysis, and this bit is
U (=0, Unsearchable): this tool will not be searched by the control;
S (=1, Searchable): this tool will be searched by the control.

4 Figure number (DWORD)

If an extra wide tool is defined by letter L in the column Tool info, the figure of the tool will have to be determined. Figures of extra wide tools (pocket occupation) are included in the tool figure table.

The Figure number indicates the relevant row of the tool pattern table.

If the value of the Figure number is 0, the tool will occupy room of one pocket. If a normal tool is defined by letter N in the column Tool info, the value of the Figure number will be invalid.

5 Tool life status (DWORD)

The tool life status can be coded as follows:

=0: Not performed

The tool is not involved in the tool life analysis, but if there is ***S*** (searchable) in the column Tool info, the tool will be taken into account by the control for Type number T, otherwise not.

=1:Not yet used

This status will be displayed if the tool is involved in the tool life analysis, but it was not used yet, i.e. the value of the tool life counter =0. It will be taken into account by the searching algorithm.

=2: Used

This status will be displayed if the tool is involved in the tool life analysis and the value indicated by the tool life counter is less than the tool life specified. It will be taken into account by the searching algorithm.

=3: Expired

This status will be displayed if the tool is involved in the tool life analysis and the value indicated by the tool life counter has reached the tool life specified. This tool will not be searched anymore by the algorithm.

=4: Broken

This status will be displayed if the PLC indicates breakage for the given tool. For the searching algoritm, this status is equal to the status ***Expired***. This also applies to the tools which are not involved in the tool life analysis.

6 Tool life counter (DWORD)

Execution of tool change will be counted if tool life analysis is switched on and set on C in the column Tool info. After execution of code M06 or code T, the value indicated by the counter will be increased by 1. When the value reaches the value specified in the column Tool life, the status of the tool will be ***Expired***. It is decided by a parameter whether the counting is done by code M06 or by code T.

If the tool life analysis is enabled in the column Tool info and ***tool life monitoring is executed for time***, the time will be measured when

it is in state Start;
the value of override is not 0;
the spindle is rotating;
and feed motion is being executed.

If the value of time measured by the tool life counter reaches the value of tool life, the tool life status of the tool will be ***Expired***.

7 Tool life (DWORD)

A tool life valid for a given tool can be set for ***number*** or for ***time***. When the ***tool life counter*** of a tool ***reaches*** this specified value, the status of the tool will be ***Expired*** in the column Tool life.

Further elements of the table:

Data number	Type number (T)	Tool info	Noticing tool life	H	D	S
1	10002001	UENCV	20	1	1	12500
2	10002001	SENCV	12	2	3	11400
3	10002001	SENCV	15	31	31	13000
4	150	SDLCV		12	13	5400
5	3210	UENTV	5m30s	326	326	2500
6	3210	SENTV	4m10s	63	63	2700
...						

8 Noticing tool life (DWORD)

If the ***difference between the values of the Tool life and the Tool life counter*** preset for a tool reaches the value specified here, a ***noticing signal*** will be sent by the control to the PLC.

9 H: Number of the length compensation cell (DWORD)

In the milling machine channel, it is the number of the length compensation cell related to the tool (code H) if the tool is involved in tool life monitoring.

10 D: Number of the radius compensation cell (DWORD)

In the milling machine channel, it is the number of the radius compensation cell related to the tool (code D) if the tool is involved in tool life monitoring.

11 G: Number of the geometry compensation cell (DWORD)

In the lathe channel, it is the number of the geometry compensation cell related to the tool if the tool is involved in tool life monitoring.

12 W: Number of the wear compensation cell (DWORD)

In the lathe channel, it is the number of the wear compensation cell related to the tool if the tool is involved in tool life monitoring.

13 Spindle speed (DWORD)

It is the spindle speed related to the tool, in 1/min.

Further elements of the table:

Data number	Type number (T)	Tool info	F	Customize, bit-type	Customize 1	...
1	10002001	UENCV	3000			
2	10002001	UENCV	2800			
3	10002001	UENCV	3300			
4	150	SDLCV	650			
5	3210	UENTV	1800			
6	3210	UENTV	1700			
...						

14 Feed (double)

Feed rate related to the tool in the dimensions of mm/min, inch/min or mm/revolution, inch/revolution.

15 Customize, bit-type data (DWORD)

For each type of tool, 8 bit-type data can be specified for any use.

16 Customize data (double)

It is the column to store floating-point data quantity of which is specified in parameter (maximum 20).

12.3.2 Cartridge Management Table

The control can manage several magazines. The ***cartridge management table*** registers the ***data numbers of the tools being in the pots of the magazines***, i.e. the number of the appropriate row of the tool management table.
The cartridge management table is global, too; it is common for each channel.

Multi-pot Magazines

At most ***4 different multipot magazines*** can be defined by parameter. They are numbered from 1 up to 4.
There can be defined by parameter the type of the magazines, i.e. if they are chain-type or matrix-type ones.
In addition to this, there is defined by parameter

– in the case of ***chain-type magazine***, initial pot number of the magazines (it is not obligatory to begin with 1), and the quantity of the pots in the magazine;

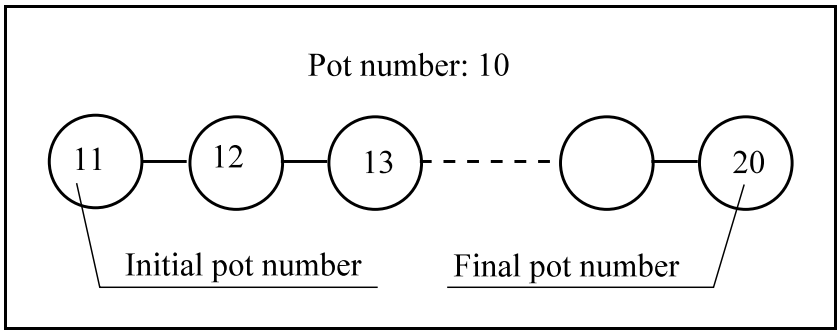


Fig. 12.3.2-1

– in the case of ***matrix-type magazine***, initial pot number (it is not obligatory to begin with 1), and the quantity of the rows and the columns in the matrix.

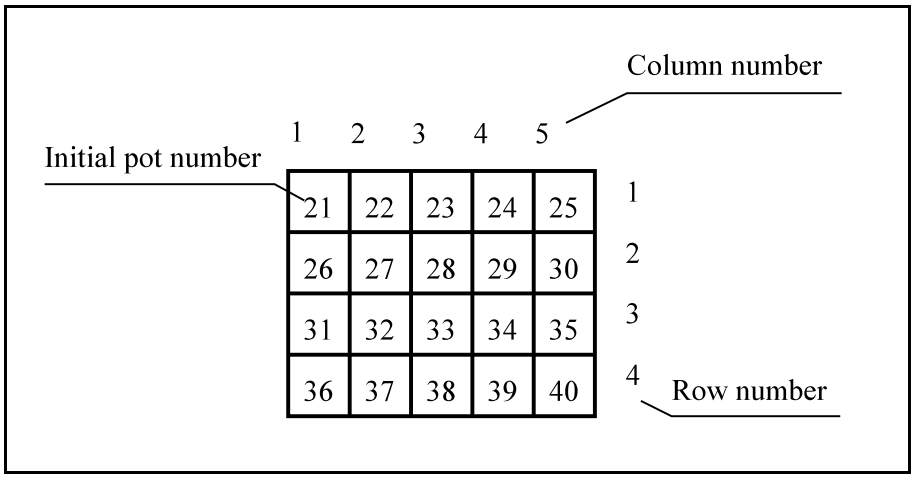


Fig. 12.3.2-2

The figure above illustrates how the control interprets numbering the pots, rows and columns in the matrix-type magazine; according to this, numbering begins from the left upper corner and

ends at the right lower corner.

It depends on the mechanical construction of the machine, which changer arms accesses to which magazines, or which change arms services which spindles.

Special one-pot magazines

In addition, each spindle into which tool can be inserted will be assigned by parameter to the magazine table. These are called ***spindle magazine***.
Depending on the changer mechanism, to the spindles there can also be assigned a ***stand-by magazine*** in which a prepared tool taken out from the magazine waits for being changed, i.e. for being inserted to the spindle.
The spindle magazines and the stand-by magazines are ***one-pot*** magazines.
The first spindle has the number 10, the second spindle has the number 20 and so on.
The numbers of the stand-by magazines assigned to the spindles are 11, 21 and so on.

Elements of the cartridge management table:

Serial number	Magazine number	Pot number	Data number
1	1	1	0
2	1	2	4
3	1	3	3
...	...	...	...
24	1	24	1
25	2	1	12
26	2	2	28
27	2	3	0
...	...	...	...
40	2	16	62
...	...	...	...
41	10	1	2
42	11	1	0
43	20	1	5
44	21	1	6
...	...	...	...

Information about the cartridge management table is summarized below:

1 Serial number

It shows the serial number of the entire cartridge management table . It is equal to the total number of the tool pot on the machine.

2 Magazine number (DWORD)

In these cells, the number of the one-pot and multi-pot magazines preset by parameter are shown.

3 Pot number DWORD

Within a magazine, the pot numbers increase by one. The initial pot number of the magazines can be given by parameter..

4 Data number DWORD

It is the data number of the tool management table, i.e. its serial number; it shows the tool being in the pot.

When the value of the data number is 0, there is no tool in the pot.

12.3.3 Tool Pattern Table

The shape of the tools occupying magazine room larger than one pot can be given in this table. If a tool is oversized, the following has to be done in the tool management table:

–the bit #2 of the ***Tool info*** column must be set to ***L***, and

–it must be written in the ***Figure number*** column, in which row of the tool pattern table the tool figure has been specified.

The tool pattern table is global, it is common for each channel. The quantity of the elements of the tool pattern table is at most 20. It means, that the system is capable of managing maximum 20 tools of different figure.

Figure number	**Left dir.**	**Right dir.**			
1					
...					

1 Figure number DWORD

Serial number in the tool pattern table. The number given in the column Figure number of the tool management table indicates the appropriate row of the tool pattern table.

2 Left dir. Right dir.

The number written in these columns says, how many pots are occupied by the tool in the given direction in the dimension

½ pot.

Certainly, into the pots occupied by a tool partially, an other tool cannot be inserted. However, in the adjacency of these pots there can be inserted such tool occupying room of ½, 1 ½, 2 ½ pots in the opposite direction.

Interpretation of the different directions is as follows:

To the left: descending pot numbers (both in chain-type and matrix-type magazines)

To the right: ascending pot numbers (both in chain-type and matrix-type magazines)

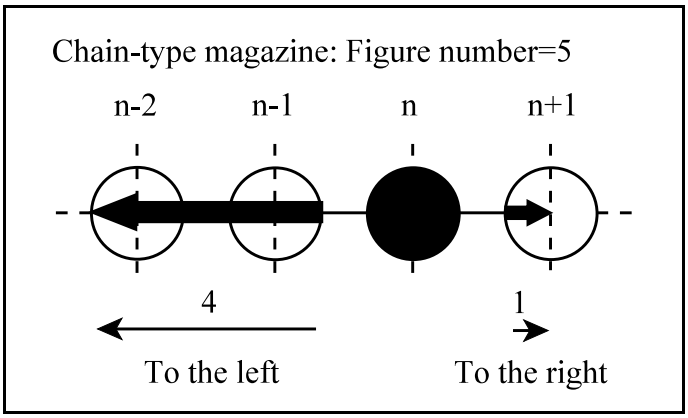


Fig. 12.3.3-1

According to the figure above, the room occupation by the tool must be specified as it is illustrated below:

Figure number	To the left	To the right			
...					
5	4	1	0	0	
...					

12.4 Modifying Data of the Tool Managemenet Table from Program (G10, G11)

The data of tool management can be modified from the part program using the instruction G10.

12.4.1 Registering Data of the Tool Managemenet Table from Program

The data of tool management can be registered from the program too. Registration can be executed row-by-row (N) in the table.
The instruction

G10 L75 P1

written in separate row switches on the registration of tool management data.
Then, the data number, i.e. the serial number of the tool management table

N

comes next, to be written in separate row.
Due to this instruction, all the data of the row N will be deleted.

☞ ***Warning!*** *During registration, all the data of the entire row N will be deleted, and then the row will be filled with the data specified during registration.*

Then, in separate row, adding values to the table columns follows:

T Q A B C L I H (J) D (K) S F
P0 R
P1 R
...
Pn R

New data number

N

and so forth:

...

Registration of the data of tool management is ended by the instruction

G11.

Explanation of notation:

N: data number, serial number of the tool management table (from 1 up to value specified in the parameter N2902 Tool Management Table Length), integer
T: type number of the tool. (T) from 0 up to 99.999.999, integer
Q: tool info bits (data of bit format, for example 10001)
A: figure number of the tool (from 0 up to 20), integer
B: tool life status from 0 up to 4, integer
C: tool life counter from 0 up to 99.999.999, integer
L: maximum tool life from 0 up to 99.999.999
I: warning tool life from 0 up to 99.999.999, integer
H: the number of the tool length compensation cell ***in the milling channel*** (from 0 up to value H specified in the parameter N1400 No. of Tool Offsets), integer
or
(**J**: the number of the tool geometry compensation cell ***in the lathe channel*** (from 0 up to value G specified in the parameter N1400 No. of Tool Offsets), integer)
D: the number of the tool radius compensation cell ***in the milling channel*** (from 0 up to value D specified in the parameter N1400 No. of Tool Offsets), integer
or
(**K**: the number of the tool wear compensation cell ***in the lathe channel*** (from 0 up to

value workpiece specified in the parameter N1400 No. of Tool Offsets), integer)
S: the spindle speed (S) related to the tool from 0 up to 99.999.999, integer
F: the feed rate (F) related to the tool, floating-point

P0 R: customize bit-type data (data of bit format, for example 10011)
Pi R: customize floating-point data, from i=1 up to the value specified in the parameter N2903 No. of Custom Columns

12.4.2 Modifying Data of the Tool Management Table from Program

The data of tool management can be registered from the program too.
The instruction
G10 L75 P2
written in separate row switches on the modification of tool management data.
Then, the data number, i.e. the serial number of the tool management table
N
comes next, to be written in separate row.
☞ ***Warning!*** *Modification relates to the data in the row N for which value is specified; the value of other data remains unchanged.*
Then, in separate row, adding values to the table columns follows:
T Q A B C L I H (J) D (K) S F
P0 R
P1 R
...
Pn R
New data number
N
and so forth:
...
Modification of the data of tool management is ended by the instruction
G11.
Explanation of notation is the same as it was in the case of registration.

12.4.3 Deleting an Arbitrary Row of the Tool Management Table from the Program

Entire arbitrary row of the tool management table can be deleted from the program.
The instruction
G10 L75 P3
written in separate row switches on the deletion of rows of the tool management table.
Then, in separate row, the data number, i.e. the number of the row to be deleted in the tool management table follows:
N
N
...
☞ ***Warning!*** *Deletion concerns all the data in the row N.*
Deletion of the rows of tool management is ended by the instruction
G11.

12.4.4 Completing the Cartridge Management Table with Data from Program

All the data numbers (serial numbers) of the tool management table can be assigned (registered) to a magazine in the cartridge management table, and to a pot within the magazine.
The instruction

G10 L76 P1

written in separate row switches on the completion of the cartridge management table with data.
Then, in separate rows

N P R
N P R
...

the data necessary for completion follow.
Completion of the cartridge management table with data is ended by the instruction

G11.

Explanation of notation:

N: number of the magazine (1-4, 10-11, 20-21, ..)
P: number of the pot
R: data number (serial number) of the tool management table containing description of the tool.

In the case of spindle magazines and stand-by magazines 10-11, 20-21, ... , since they are one-pot magazines, it is not necessary to fill the address P; it will not be taken into account by the control.
Assignment R0 means, that the pot specified at the address P is empty.
Numbering the pots of the magazines is defined by the builder of the machine tool using parameter setting.

12.4.5 Completing the Tool Pattern Table with Data from Program

To all the tools of the tool management table there can be assigned a figure number of the of the tool pattern table. The tool pattern table can be completed from program too.
The instruction

G10 L77 P6

written in separate row switches on the completion of the tool pattern table with data.
Then, in separate rows

N P Q R S T
N P Q R S T
...

the data necessary for registration follow.
Completion of the tool pattern table with data is ended by the instruction

G11.

Explanation of notation:

N: figure number of the tool (1-20, serial number of the tool pattern table)
P: room occupation to the left in the dimension ½ pot (in the direction of descending pot numbers)
Q: room occupation to the right in the dimension ½ pot (in the direction of ascending pot numbers)
R: room occupation upwards in the dimension ½ pot (in the direction of descending pot numbers)
S: room occupation downwards in the dimension ½ pot (in the direction of ascending pot

numbers)

T: geometry of the tool (=0: type A, rectangle; =1: type B, circle)

Filling the addresses R, S and T makes sense in the case of matrix-type magazines only.

Explanation of room occupation:

P3

means, that the tool occupies a room of 1,5 pots to the left.

12.5 Programming Relevancies of the Tool Management

If use of tool life management in the tool management table is intended, the tools of the same type will have to be registered at the same codes T.

Let, for example, the type number

T81501070

represent 150 mm long, 10 mm diameter and 140° point angle (140/2=70) drills.

Let us presume that 2 pieces of them have been registered at the data numbers 2 and 9 in the tool management table.

The tool is of a normal size and the tool life management is carried out for number

The tool life of the tools with the data number 2 and 9 is 200 and 180, respectively.

The tool with the data number 2 has been used 131 times, but the tool with the data number 9 has not been used yet.

The state of the tool management table is as follows:

Data number	Type number (T)	Tool info	Figure number	Tool life status	Tool life counter	Tool life
...						
2	81501070	SENCV	0	Used	131	200
...						
9	81501070	SENCV	0	Not yet used	0	180
...						

On the given machine, let the magazine (magazine 1) be of 24 pots, and let there be a spindle magazine (magazine 10) too. The drill with the data number 2 is in the pot 8 of the magazine 1, the drill with the data number 9 is in the pot 4 of the magazine 1. The spindle is empty:

Serial number	Magazine number	Pot number	Data number
...			
4	1	4	9
...			
8	1	8	2
...	...	...	...
24	1	24	...
25	10	1	0

Due to the instruction

```
T81501070 M6,
```

the control takes out from the pot 8 of the magazine 1the tool with the data number 2, which tool is used but not expired yet.

12.5.1 Compensation Call in the Case of Tool Life Management

From the part program, a tool is referred to by its type number.
If several tools with the same type number are registered in the tool management table, the length and diameter compensation of the tools with the same type number could be different.
Because of this, the compensation register of the tool just changed cannot be referred directly, at the addresses H and D.
A compensation number can be specified in the parameter N2923 No. of Offset Code. When this number will be written at the addresses H and D, the control, knowing the data number of the tool being in the spindle, will determine the number of the compensation cell related to the tool from the tool management table.

Having used the example of the above sub-chapter, let the length and radius compensation cells H12 and D23 relate to the tool with the data number 2, and the cells H13 and D 14 relate to the tool with the data number 9.

Data number	Type number (T)	Tool info	Figure number	Tool life status	H	D
...						
2	81501070	SENCV	0	Used	12	23
...						
9	81501070	SENCV	0	Not yet used	13	14
...						

Let the value of the parameter N2923 No. of Offset Code be 99.
An example of the indirect compensation call is as follows:

```
T81501070 M6    (it changes the tool with the data number 2)
G43 Z10 H99     (it calls the value of the length
                compensation cell 12)
...
G42 X10 Y0 D99 (it calls the value of the radius
                compensation cell 23)
...
```

12.5.2 Tool Call Referring to the Magazine and Pot Numbers

It could be necessary to call from the part program a particular tool being in a given pot of a given magazine.
A code M can be specified by the builder of the machine tool in the parameter N2924 M Code of Particular T.
Programming the code M together with the address T in one block, the code will change interpretation of the address T for the control: the control will handle the code T as direct reference to a ***tool position (magazine+pot)***:

Mm Tjjjjkkkk

where:

m: the number given in the parameter N2924 M Code of Particular T;
iiii: the number of the magazine being referred to;
kkkk: the pot number of the magazine iiii.

The leading zeros can be eliminated. Accordingly, the 8 digit that can be given at the address T is interpreted by the control as 2 tetrads: the tetrad of upper place value represents the magazine number, and the tetrad of lower place value represents the pot number.

The meaning of the code T100001 is:
- the magazine number: 10 (spindle magazine)
- the pot number: 1 (pot 1, the spindle magazine is one-pot one)

The meaning of the code T100014 is:
- the magazine number: 1 (tool magazine)
- the pot number: 14 (pot 14)

Having used the example of the above sub-chapters, let us assume, that the tool life of the tool with the data number 2 has expired after manufacturing 200 workpieces (provided that the tool T81501070 has been called once in the program):

Data number	Type number (T)	Tool info	Figure number	Tool life status	Tool life counter	Tool life
...						
2	81501070	SENCV	0	Expired	200	200
...						

9	81501070	SENCV	0	Not yet used	0	180
...						

It is intended to replace the tool of expired tool life by a new tool. Because of the machine construction, the tool can only be changed using the spindle. Therefore, this tool has to be changed by the reference magazine+pot.
Let the value of the parameter N2924 M Code of Particular T be 62.
When the tool should be pulled out of the magazine, the tool with the data number 2 is in the pot 12.

Serial number	Magazine number	Pot number	Data number
...			
12	1	12	2
...			
21	1	21	9
...	...	...	...
24	1	24	...
25	10	1	0

The reference is as follows:

```
M62 T10012 (magazine 1, pot 12)
```

12.5.3 Reading out the Data of Tools being in the Spindle and Stand-by Magazines

If it is needed to read out the tool management data of a tool using the macro variables #8401, #8402, ... , it is necessary to specify in the macro variable

#8400 (10, 11, 20, 21, ... writable, readable)

the spindle magazine or the stand-by magazine in which that tool exists. Only the magazine numbers given in brackets can be defined. If there are several spindle magazines or stand-by magazines are on the machine, please ask the builder of the machine tool for information.
If there is only one spindle magazine on the machine and there is no stand-by magazine, the value has not to be given to the macro variable #8400; the macro variables #8401, ... relates always to the tool being in the spindle.

The readable data are as follows:

Macro variable	Description
#8401	Data number (seiel number in the tool management table)
#8402	Type number of the tool (code T)

Macro variable	Description
#8403	Value indicated by the tool life counter
#8404	Maximum tool life of the tool
#8405	Value of the warning tool life
#8406	Tool life status
#8407	Customize bit-type data
#8408	Tool info
#8409	H: number of the length compensation cell (in the milling machine channel)
#8410	D: number of the radius compensation cell (in the milling machine channel)
#8411	S: spindle speed
#8412	F: feed rate
#8413	G: number of the geometry compensation cell (in the lathe channel)
#8414	W: number of the wear compensation cell (in the lathe channel)
#8431	Customize data 1
#8432	Customize data 2
#8433	Customize data 3
#8434	Customize data 4
#8435	Customize data 5
#8436	Customize data 6
#8437	Customize data 7
#8438	Customize data 8
#8439	Customize data 9
#8440	Customize data 10
#8441	Customize data 11
#8442	Customize data 12
#8443	Customize data 13
#8444	Customize data 14

Macro variable	Description
#8445	Customize data 15
#8446	Customize data 16
#8447	Customize data 17
#8448	Customize data 18
#8449	Customize data 19
#8450	Customize data 20

The number of the customize data can be given in the parameter N2903 No. of Custom Columns. The first customize data is always a bit-type data.
The macro variables above are readable only.
If the spindle magazine or the stand-by magazine selected in the macro variable #8400 is empty (there is no tool in it), the values of the macro variables will be:

#8401=0 (data number)
#8402= #8403= ...= #8450= #0 (empty)

Compensations (H and D) or technological parameters (F and S) assigned to the tool can be called using macro variables, too.
If, for example, a subprogram call is assigned for the tool change code (M6), the following can be written into the subprogram:

```
...
M6         (Tool change)
#8400=10   (the spindle magazine 1)
H#8409 D#8410 S#8411 F#8412
...
```

13 Miscellaneous and Auxiliary Functions

13.1 Miscellaneous Functions (Codes M)

Having written a numerical value of maximum 8 digits behind the address **M**
Mnnnnnnnn,
the NC transfers the code to the PLC.
The leading zeros can be eliminated.
Eight several codes M can be transferred by the control to the PLC at the same time, i.e. ***maximum 8 codes M can be written in one block***.
The ***execute sequence*** of the functions M written in one block is determined ***in the PLC program*** by the builder of the machine tool.
If motion instruction and miscellaneous function (code M) are programmed in the same block, the miscellaneous function will be executed during or after execution of the motion instruction.
All the codes M, even those that are executed by the control, will be transferred by the control to the PLC.
The way of execution is determined by the builder of the machine tool.

The program control codes M:

M0: programmed stop

This code is executed by the PLC. At the end of the block in which the M0 is specified, the control

arrives at the condition Stop;
stops the spindles;
switches off the coolant.

All the modal functions remain unchanged. Due to the start, the control restarts the spindles, switches back the coolant and continues the program.

M1: conditional stop

This code is executed by the PLC. Its effect is the same as the effect of the code M0. It will only be executed when the button CONDITIONAL STOP is activated. If the appropriate button is not activated, this code will be ineffective.

M2, M30: end of program

This code is executed by the PLC. It means the end of the main program. The machine functions are reset by the PLC program; generally, it stops the rotation of the spindles and switches off the coolant.
Each of the instructions M2 or M30 executed increases the value of the workpiece counters by one, unless other code M is assigned in the parameter N2305 Part Count M for stepping the the counter.

M96: enabling the interruption macro

This code is transferred to the PLC but it is executed by the control. It enables the interruption signal coming from the PLC, due to which the interruption macro will be called.

M97: disabling the interruption macro

This code is transferred to the PLC but it is executed by the control. It disables interruption signal

coming from the PLC to be valid, and running the interruption macro.

M98: calling subprogram
This code is transferred to the PLC but it is executed by the control. Due to it, calling a subprogram will be occurred.

M99: end of subprogram
This code is transferred to the PLC but it is executed by the control. Due to it, execution returns to the position of the call.

The spindle control codes M
M3, M4, M5, M19: the codes of spindle management
These codes are executed by the PLC. See the sub-chapter *Functions M Controlling the Spindle*.

The spindle management codes M that can be specified in parameter
In the parameters N0689 Spindle M Low and N0690 Spindle M High, additional ***spindle control codes M*** can be assigned in one array. See the sub-chapter *Functions M Controlling the Spindle.*
The ***spindle control codes M*** together with the codes ***M3, M4, M5, M19*** are ***exclusive ones***, giving only one such code in one block is allowed.
These codes are executed by the PLC.
The initial value and the number of the ***codes M indexing the spindles*** can be specified in the parameters N0820 Start M of Spnd. Pos. and N0821 No. of M Code for Spnd. Pos., respectively. See the sub-chapter *Programming of the Positioning the Spindles*.
These codes are executed by the control.

The speed ranges management codes M
M11, ..., M18: the codes of speed range change
These codes are executed by the PLC. See the sub-chapter *Managing the Speed Ranges*.

The tool change code M
M6: the code of the tool change
This code is executed by the PLC.

The code M groups that can be set in parameter
There can be assigned ***16 different groups of codes M*** in 16 pairs of parameters. These are executed by the PLC.
The codes of the lowest number in the group has to be written in the parameters N1341 M GR Low 1, ..., N1356 M GR Low 16; the codes of the highest number in the group has to be written in the parameters N1357 M GR High 1, ..., N1372 M GR High 16.
The ***groups of codes M*** must be specified in such a way that the codes mean ***exclusive machine statuses***.
During execution of the program, the control ***filters the codes M in such a way*** that only one of the Codes M in the group is in the given block, otherwise the control sends the error message *Conflicting M codes*.
When it gathers the codes M, the control takes the values preset in the parameters into account in the course of ***finding blocks***, too. From the codes M related to a group, the only code given last will be gathered by the control.

The values of these codes M will be displayed by the PLC program on the screen in the ***window of codes M***, too.
For example:
Let the codes M of the coolant management be the following:
M8: coolant on
M9: coolant off
The parameters are set as follows:
N1341 M GR Low 1=8
N1357 M GR High 1=9
The codes M of braking the axis **A**:
M400: automatic braking on
M401: automatic braking off
The parameters are set as follows:
N1342 M GR Low 2=400
N1358 M GR High 2=401
In the part program, writing either M8 or M9 is allowed in one block, otherwise error will be indicated by the control during program run. This is related to the group of M400 and M401, too.
During block finding, from among the M8 and M9, the control gathers and gets it executed the only one programmed last. This is related to the group of M400 and M401, too.
From among the appropriate machine statuses, the only status code valid in the group will be displayed in the window of codes M.

The codes M carrying out synchronization of the channels
In the parameters N2201 Waiting M Codes Min and N2202 Waiting M Codes Max, there can be assigned a gruop of codes M, using which synchronization, waiting for each other can be realized. It is executed by the control.

13.2 Auxiliary Functions (A, B, C, U, V or W)

In addition to the addresses M, S and T, there can be assigned in parameter further 3 addresses at which auxiliary function can be transferred to the PLC program. All the 3 auxiliary functions can be transferred by the control at the same time.
In the parameters N1333 Aux Fu Addr1, N1334 Aux Fu Addr2, N1335 Aux Fu Addr3, from among the addresses A, B, C, U, V and W there can be selected ones at which auxiliary functions can be transferred.
For the auxiliary functions, value can be given using number of maximum 8 decimal digits.
If motion instruction and auxiliary function (code M) are programmed in the same block, the auxiliary function will be executed during or after execution of the motion instruction.
The execute sequence is determined by the builder of the machine tool, and it is contained in the machine tool specification.
For example, at the address B, indexing of the indexing table can be realize.

13.3 Buffer Emptying Functions

The block processor in the control ***reads ahead, processes the blocks, and then buffers them***.
The executive element (the interpolator and the PLC) takes the processed blocks from the buffer, and then executes motions and functions specified in the blocks.
In certain cases, it could be necessary ***to stop reading ahead*** the blocks ***in order to synchronize***

action between the control and the PLC.
For example, if the PLC asks the NCfor one or more axes in order to move them, reading ahead will have to be suspended. Reading ahead can be continued after execution of the function only, when the PLC gave back the axes to the NC. Then, the NC can continue the machining from the axis position changed by the PLC.

Ten single codes M, eight groups of codes M, all the three auxiliary functions and the codes S and T can be assigned in parameter for buffer emptying.
The buffer emptying functions are determined and set by the builder of the machine tool.

For example, in the course of execution of the program, reading ahead the blocks is necessary to take the tool radius compensation into account (G41 and G42). If buffer emptying function is programmed during G41and G42, the control suspends calculation of the tool radius compensation, and as a result of this, the contour will damage.

14 Part Program Configuration

The structure and the format of the part program have already been shown in the introduction. In this chapter, organizing the part programs will be dealt with.

14.1 Block Number (Address N)

The blocks of the program can have ***serial number***. The block numbers can be managed as ***labels*** which can be referred to in the other parts of the program. Blocks are numbered by the instruction

Nnnnnnnnn.

Maximum 8 digits can be written at the address N. It is not obligatory to use address N. Some blocks could be numbered, others not. The block numbers need not follow each other in a consecutive order.

14.2 Conditional Block Skip (/ address)

Conditional block skip can be programmed using slash instruction

/n.

The ***value of the slash address can be n=1-8***. The numbers from 1 up to 8 are serial numbers of switches. The conditional block switch No.1 can be found on the operator's panel of the control. Mounting the other switches is optional and is determined by the builder of the machine tool.
If a conditional block skip /n is programmed at the beginning of a block,
– that block ***will be omitted*** from the execution when the n^{th} switch is ***on***,
– that block ***will be executed*** when the n^{th} switch is ***off.***

If address /only is programmed at the beginning of the block, that will be related to the switch 1:

```
/ N1200 G0 X200      (the block will be omitted if the
                     switch 1 is on)
```

It can be programmed in the following way, too:

```
/1 N1200 G0 X200     (the block will be omitted if the
                     switch 1 is on)
```

If the intention is that a switch of conditional block skip should be taken into account by the control even in the block preceding the execution of the conditional block, the parameter N1337 Execution Config will have to be set in ***#4 CBB=0***. Then, the instruction of conditional block (blocks beginning with character /) ***suppresses*** the reading ahead of block. In this case, using ***G41 and G42***, the contour ***becomes distorted***, but it is enough to ***turn the switch of conditional block skip on during execution of the previous block*** in order that it will be effective.

In order that the instruction of the character / does not suppress the reading ahead of block, the parameter N1337 Execution Config will have to be set in ***#4 CBB=1.*** Then, the instruction of conditional block (blocks beginning with the character /) ***does not suppress*** the reading ahead of block. In this case, using ***G41 and G42***, the contour ***does not become distorted***, but the switch of conditional block skip ***has to be turned on before execution of the program***, for sure effectiveness.

Some switches can also be used by the PLC program in order to control the program run.

For example, in the case of a machine equipped with workpiece feeder, the main program can be made endless using M99:

```
... (part program)
M90 (stepping the workpiece counter)
/8 M30 (the switch 8 is controlled by the workpiece counter)
M99
```

When the workpiece counter reaches the needed workpiece quantity, the PLC turns the switch 8 of conditional block skip off, the program runs to the M30, and the execution stops.
The example above is correct when the parameter N1337 Execution Config is in the state #4 CBB=0, i.e. the switches / suppress the reading ahead of block.
When the parameter N1337 Execution Config is in the state #4 CBB=1, the program will run correctly in that case only if the code M90 is set for buffer emptying.
Before changing the parameter, please ask the builder of the machine tool for information about effects!

14.3 Writing Comments into the Part Program: (comment)

If a part of the program is parenthesized between ***round brackets ( )***, the section between the brackets will not be taken into account by the block processor.
Thus, notes (comments) can be written into the part programs.
If the intention is that a part of the part program will not be executed by the controller but that program part will not be deleted from the program, that given part will have to be put into round brackets.

For example:

```
N10 G0 X100     (positioning to X100)
N20 Z30
(N30 G1 Z60 F0.3
N40 X300)
...
```

A comment is written in the block N10. The block N30N40 is put into brackets, so these two blocks will be taken into account by the control.

14.4 Main Program and Subprogram

Two kinds of program are distinguished: main program and subprogram. Macro is a subprogram to which arguments can be transferred.
In the course of machining a part, repetitive actions could be encountered, which can be described by the same program part. In order to avoid writing the repetitive parts many times in the program, from these parts subprogram can be organized that can be called from the main program.
Structure of a main program and a subprogram is given in the Introduction.
The difference between them is as follows: after execution of a main program, machining is completed and the control waits for restart; but after execution of a subprogram, the execution returns to the calling program and continues machining from there.
In terms of programming technique, the difference between the two programs lies in the way of terminating the program. The end of the main program is indicated with the codes M02 or M30 (it is not obligatory to use them), whereas the subprogram must be terminated with the code M99.

14.4.1 Identification of Programs in Memory. The Program Number (O)

In the memory, programs are placed in folders defined by the user and having different names. Programs in the folders are identified by their file name. The control will consider a file as a part program, i.e. a file can be run as a part program if its extension (the fraction after the dot '.') is as follows:

file name.txt
file name.prg
file name.nct

or

file name.nc

The file name of a program can be composed of alphabetical and numerical characters. Subprograms are stored in separate file, they together with the main program have not to be in one file.

Program number

The program number

Onnnn.ext

or

Onnnnnnnn.ext

is a special file name beginning with the letter O obligatorily and followed by 4 or 8 decimal digits. For their extension (.ext), the notes above are valid.

Onnnn: ***Letter O followed by 4 digits*** together with the leading zeros.

O1 is an invalid file name,
O0001 is a valid file name;

or

Onnnnnnnn: ***Letter O followed by 8 digits*** together with the leading zeros.

O01234 is an invalid file name,
O00001234 is a valid file name.

14.4.2 Calling a Subprogram (M98)

Subprograms can be called in two ways, by program number or by file name.

Calling a subprogram by program number

The instruction line

M98 P....

generates a subprogram call. The program number of the called program is written at the address P. At the address P, the leading zeros can be eliminated and it is not allowed to write extension after the number. Due to the instruction, execution of the program will continue at the subprogram the number of which defined at the address P:

calling program		**subprogram**	**note**
O0010			execution of the program O0010
.....			
.....			
M98 P11	--->	O0011	calling the subprogram O0011

```
                                    ......          execution   of   the
                                    ......          subprogram O0011
                                    ......
next block  <---                    M99             return to the calling
                                                    program
......                                              continuing    the
......                                              program O0010
```

In the case of subprograms called at the address P and by program number, the following limitations concerning their location in the folder system and their file names are valid:

– The subprograms have to be in the same folder where there is the program calling them.

– For file name of the subprograms called by program number, the limitations described in the previous subchapter are valid.

– ***The extension of subprograms and the extension of the program calling them have to be the same***:

For example, if the file name of the main program is

Foprogram.**prg**,

the extension of the subprogram called from the 'Foprogram.prg' will have to be .prg, too:

in the case of O1234.**prg** the call is executed;

in the case of O1234.**nct** error message is generated.

Calling a subprogram by file name

The instruction line

M98 <alprogram.nct>

calls the subprogram named 'alprogram.nct' being together with the program in the same folder. The ***file name*** has to be written in between the symbols **<** ***less than*** and **>** ***greater than***
In this case the extensions of the main program and the subprogram do not need to be the same. ***The relative path of the file can also be given between the symbols < and >.*** The path has to be always given from the folder of the calling program.

If the called subprogram 'alprogram.prg' is in a subfolder named 'alprogramok' being ***a level lower*** from the folder of the program 'program1.nct' which calls it:

...

alprogramok (folder)

program1.nct (file)

the subprogram call will be executed by the instruction

M98 <\alprogramok\alprogram.prg>.

If the called subprogram 'alprogram.prg' is in ***a level upper*** from the folder 'foprogramok' of the program which calls it:

...

foprogramok (folder)

alprogram.prg (file)

the subprogram call will be executed by the instruction

M98 < ..\alprogram.prg>.

Stepping backward and forward can be done through several levels.
For example:

< ..\..\..\folder1\folder2\folder3\file.txt>

However, the ***length of the text between the symbols < and >*** can be ***maximum 60 characters.***

☞ ***Note:*** *In the course of giving the path, the symbol \ (backslash) has to always be used; it should not be confused with the symbol / (per).*
For stepping backward in the folder system, 2 dots (..) has to always be used.

Calling a subprogram by number of repetitions

The instruction line

M98 P.... L....

or

M98 <path \ file name> L....

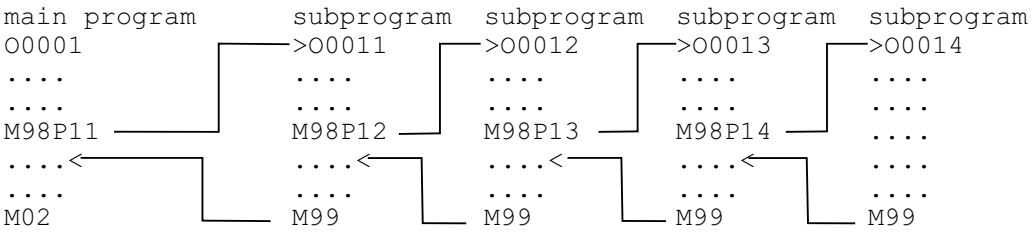
calls in succession the specified subprograms in quantities given at the address L.
The address L can be specified by maximum 8 decimal digits.
If value is not given to the L, the subprogram will be called once, i.e. L=1 is assumed by the control.
The instruction

```
M98 P11 L6
```

means that the subprogram O0011 has to be called 6 times in succession.

Multi-level subprogram call

A subprogram can be called from another subprogram, too. Subprogram calls (together with macro calls) can be nested to maximum ***16 levels***.



14.4.3 Return from a Subprogram (M99)

Return to the block following the call

Using the instruction

M99

in a subprogram means the end of that subprogram, and control will be given back to that block of the calling program which follows the call:

calling program		**subprogram**	**note**
O0010			execution of the program O0010
.....			
.....			
.....			
N101 M98 P11	--->	O0011	calling the subprogram O0011
		.....	execution of the subprogram O0011
		.....	
		.....	

```
N102 ......       <---            M99               return to the next
                                                    block of the calling
                                                    program
......                                              continuing the
......                                              program O0010
```

Return to a given block

Using the command

M99 P...

in a subprogram means the end of that subprogram, and control will be given back to that block of the calling program the number of which was specified at the address P. Maximum 8 decimal digits can be written at the address P.

```
hívó program                        alprogram       megjegyzés

O0010                                               execution of the
......                                              program O0010
......
......
N101 M98 P11      --->              O0011           calling the
                                                    subprogram O0011
                                    ......          execution of the
                                    ......          subprogram O0011
                                    ......
N250 ......       <---              M99 P250        return to the next
                                                    block of the calling
                                                    program
......                                              continuing the
......                                              program O0010
```

Return by modification of cycle counter

The instruction

M99 (P...) L...

modifies the cycle counter of the calling program. If 0 is written at the address L, the subprogram will be called once only. For example, if the subprogram O0011 is called by the instruction

```
M98 P11 L20,
```

and return from there is executed by the instruction

```
M99 L5,
```

the subprogram O0011 will be called 6 times altogether.

Maximum 8 decimal digits can be written at the address L.

14.4.4 Jump within the Main Program

Using the instruction

M99

in the main program produces an unconditional jump to the first block of the main program, and the control continues execution of the program from here. Using the instruction results in endless cycle:

```
O0123
N1... <
...
.....
.....
M99
```

Using the instruction

M99 P.....

in the main program produces an unconditional jump to that block of the main program the number of which is specified at the address P, and the control continues execution of the program from here. Using the instruction results in endless cycle:

```
O0011
....
....
N128....<
....
....
M99 P128
```

```
O0011
....
M99 P225
....
....
N225  <
....
```

The program can be recovered from the endless cycle either by reset, or by programming the block containing the instruction M99 with conditional block skip

```
/ M99;
```

depending on the position of the conditional block skip switch, jump will happen or not.

14.5 Functions M of Channel Synchronization

In the course of multi-channel operation, it could be necessary that the running of a program in a channel has to wait at a given point until a program or programs running in one or more other channels come to execution of certain operations. This is the synchronization among channels. This synchronization can be realized by the use of so-called waiting codes M.

The codes M for waiting are ***program organizing codes M***; ***they are processed by the control*** and they are not transferred to the PLC. ***They are buffer emptying codes M***, i.e. preprocessing the blocks is being intermitted until synchronization is completed in all the channels, and it will only be continued after synchronization.

Maximum 100 codes M for waiting can be assigned in parameter. The initial value of the group can be given in the parameter N2201 Waiting M Codes Min, and the final value of the can be given in the parameter N2202 Waiting M Codes Max.

If, for example,

N2201 Waiting M Codes Min=500 and

N2202 Waiting M Codes Max=599,

then the codes M M500, M501, M502, ... , M599 M can be used for waiting.
The instruction

Mm Ppppppppp

will program the waiting among two or more channels. The waiting has to be programmed in separate line.

m: one of the codes M for waiting specified in parameter,

ppppppppp: number of those channels among which the waiting has to be done. Since there can be maximum 8 channels in the system, the address can have maximum 8 digits.

If, for example, the waiting has to be done between the channels 1 and 2, the instruction

```
M501 P12
```

will have to be written into both programs, to the appropriate point.
The PLC can ignore the waiting by function M or by push-button, using the flag CP_NOWT. It can be useful in the case, when the program should be run in one of the channels only and the codes M for waiting should not be glossed.

An example:
Let the minimum and maximum values of the codes M be 500 and 599, respectively; and let there be 3 channels:

The program of the channel 1	**The program of the channel 2**	**The program of the channel 3**
... machining	... machining	... machining
N60 M501 P12		
waits for the channel 2	N100 M501 P12	
... machining	...machining	N110 M502 P123
		waits for the channels 1 and 2
	M502 P123	
	waits for the channels 1	
N130 M502 P123	and 3	
... machining		
	... machining	... machining

Explanation:

First, the channel 1 runs to the code M501 and waits until the channel 2 runs to it, too. After synchronization, both channels continue its own program. Meanwhile, the channel 3 operates continuously.

It is the channel 3 that runs to the code M502 first of all, then the channel 2 does this, and then the channel 1, last of all. The channel 3 has to wait until the channels 1 and 2 run to the code. After reaching this point, machining in all three channels can start.

15 Tool Compensation

In order that overhang and radius etc. values related to different tools should not be taken into account in the part program in the course of specifying the coordinates, tool characteristics are gathered in a so-called offset table. Whenever a tool has to be called in the part program, it has to be specified where the characteristics of that given tool can be found in the offset table. According to this, the control directs the tool along the programmed path, taking the referred offsets into account.

15.1 The Offset Memory. Referring to Tool Compensation (H and D)

Referring is made
to the tool length compensation at the address **H**,
to the tool radius compensation at the address **D**,
The number following the address is the compensation number, and it indicates the compensation value to be called. The range of the addresses H and D is 0-999. The leading zeros can be eliminated.
In the case of using the ***tool life management***, it is not possible to refer to the compensation register of the tool being changed, directly at the address H and D.
In the ***parameter*** N2923 No. of Offset Code ***there can be given a compensation number***. If this number is written at the addresses H and D in the part program, the control, with knowledge of the data number of the tool being in the spindle, will determine the number of the compensation group related to the tool. See the subchapter Compensation Call in the Case of Tool Life Management.

Distribution of the compensation memory among the channels

The ***quantity of the tool compensation groups*** to be accessible in a given channel can be given ***for each channel*** in the parameter N1400 No. of Tool Offsets. Length and radius compensations also pertain to each group.

In the whole system, ***the sum of the milling machine channel compensation groups*** has not to be more than ***999***. In each channel, referring to the compensation group begins from 1 and proceeds up to the preset parameter value, let it be executed either from program at addresses H or D, or from the offset table.

The number of ***common compensation group callable from each milling channel*** can be given in the parameter N1401 No. of Common Tool Offsets M.

In each channel, referring to a compensation group from 1 up to the No. of Common Tool Offsets T , indicates the common values, let it be executed either from program at addresses H or D, or from the offset table.

An example:

Let there be 3 channels. Let there be 30 compensations in the channel 1 (No. of Tool Offsets L1=30), 40 compensations in the channel 2 and 60 compensations in the channel 3. Let the quantity of the common compensations be 10 (No. of Common Tool Offsets M=10).

<table>
<tr><td></td><td>Channel 1</td><td>Channel 2</td><td>Channel 3</td></tr>
<tr><td>N001</td><td colspan="3" rowspan="3">From N001 up to N010, all the three channels read the common compensations (H1-H10, D1-D10)</td></tr>
<tr><td>...</td></tr>
<tr><td>N010</td></tr>
<tr><td>...</td><td rowspan="2">From N011 up to N030, it reads its own compensations (H11-H30, D11-D30)</td><td rowspan="4">From N011 up to N040, it reads its own compensations (H11-H40, D11-D40)</td><td rowspan="6">From N011 up to N060, it reads its own compensations (H11-H60, D11-D60)</td></tr>
<tr><td>N030</td></tr>
<tr><td>...</td><td></td></tr>
<tr><td>N040</td><td></td></tr>
<tr><td>...</td><td></td><td></td></tr>
<tr><td>N060</td><td></td><td></td></tr>
</table>

The quantity of compensation groups used in such a way: 10+(30−10)+(40−10)+(60−10)=110.

Distribution of the compensation table within the milling machine channel

In addition to milling machine compensations, lathe compensations can also be used in the milling machine channel, if the machine design allows turning operations, too.

If, in the parameter ***N1416 Comp. Config on Mills,*** the value of #2 TOL is

#2 TOL=0, ***only milling*** operations are executed on the machine, no lathe compensations are needed,

#2 TOL=1, turning operations are also executed on the machine, lathe compensations are also needed.

Distribution of the compensation memory within a channel, when using **only milling machine compensations**

In this case, the value of the #2 TOL of the parameter N1416 Comp. Config on Mills has to be set to 0.

	Length compensation L: H code		**Radius compensation R and rounding r: D code**			
Compensation number	**L geometric**	**L wear**	**R geometric**	**R wear**	**r geometric**	**r wear**
1	350.000	-0.130	5.000	-	5.000	-0.010
2	830.000	-0.102	10.000	0.012	6.000	-0.006
3	400.000	.	30.000	-	0.000	0.000
.	.	.	.	0.008		
.	.	.	.	-		
				0.160		
				.		
				.		

The tool offset table contains the tool length (L), the tool radius (R) and corner rounding (r).
There is no compensation number 0 in the table, ***offset values on it are always zeros***, i.e.

H0 always means length compensation of 0, and
D0 always means radius compensation of 0.

Geometry value: length/radius of the tool measured. It is a signed number.

Wear value: amount of wears resulting in the course of machining. It is a signed number. In the case of manual data input, the absolute value of the input data is limited to the value given at parameter N1415 Max. Amount of Wear Comp.

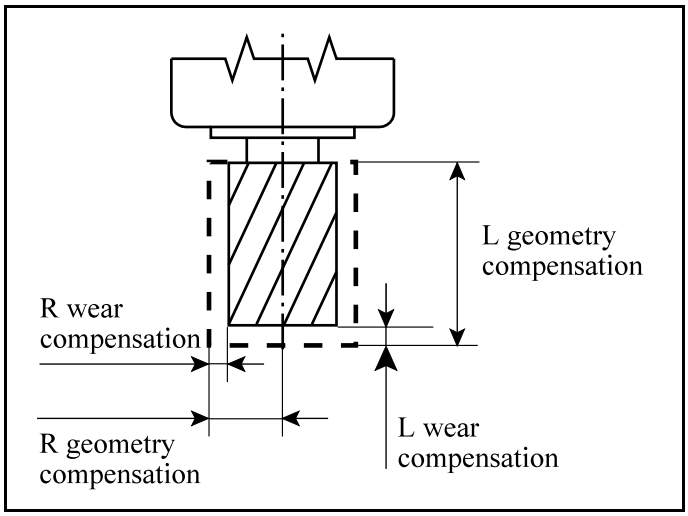


Fig. 15.1-1

Corner radius: r

In the case, if

r=0: the tool is a square end mill and its radius is R.

If

r<R: the tool end is rounded by a radius r, and its radius is R.

If

r=R: the tool is a ball end mill and its radius is R.

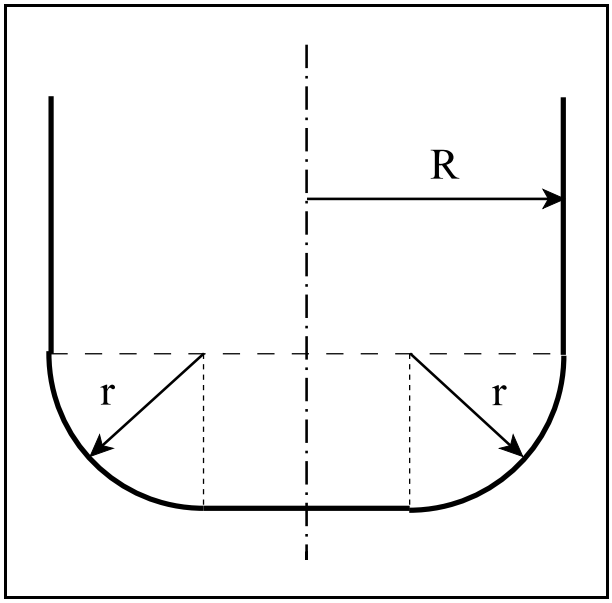


Fig. 15.1-2

Distribution of the compensation memory within a channel, when using **lathe compensations, too**

In this case, the value of the #2 TOL of the parameter N1416 Comp. Config on Mills has to be set to 1.

In the merged compensation table, the columns L, R, r of the milling machine compensation table are supplemented by the columns X, Y Q for the lathe tool compensation. The length compensation of the lathe tool in the Z direction is in the column L/Z, while the radius of the rounding is in the column R.

Number	H code		D code				H code		H code		D
	L/Z		R		r		X		Y		Q
	g	k	g	k	g	k	g	k	g	k	
1											
2											
...											

☞ ***Attention!***

The correction number 00 is not in the table, the correction values on it are always zero.

The compensation in X, Y, Z direction and the radius compensation (R) are made up of two parts: the geometric and the wear value.

Geometry value: length/radius of the tool measured. It is a signed number.

Wear value: amount of wears resulting in the course of machining. It is a signed number. In the case of manual data input, the absolute value of the input data is limited to the value given in the parameter N1415 Max. Amount of Wear Comp.

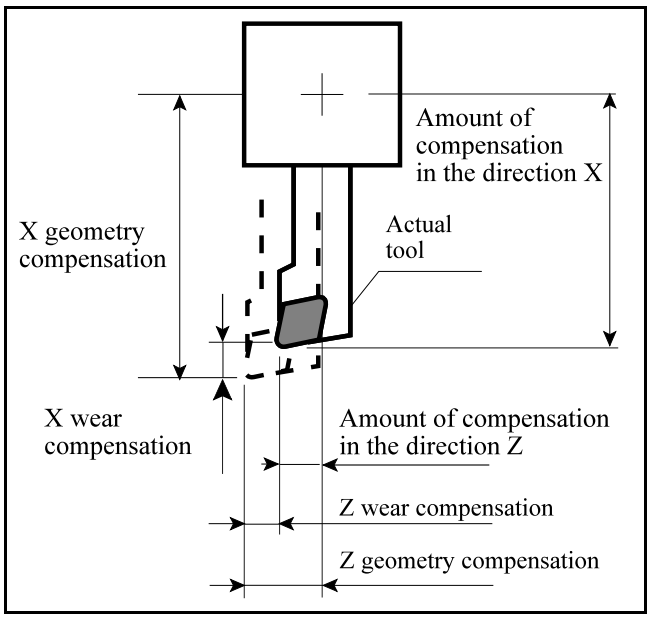


Fig. 15.4-3

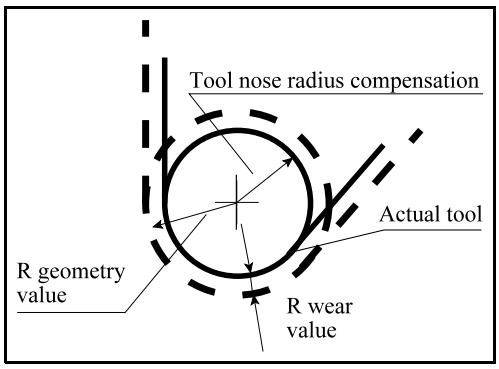


Fig. 15.4-4

Tool nose position: For milling tasks, the value of the ***code of tool nose position (Q)*** is 0. This code is to be used for turning, when tool radius compensation is applied. The code of tool nose position is always ***called*** by the ***address D***, i.e. by the code of radius compensation. The range of its values:0 ... 9.

The code of tool nose position refers to the direction of position where the imaginary tip of the tool is , viewed from the center point of the tool nose circle. It has to always be given by taking the first and second axes of the selected plane into account.

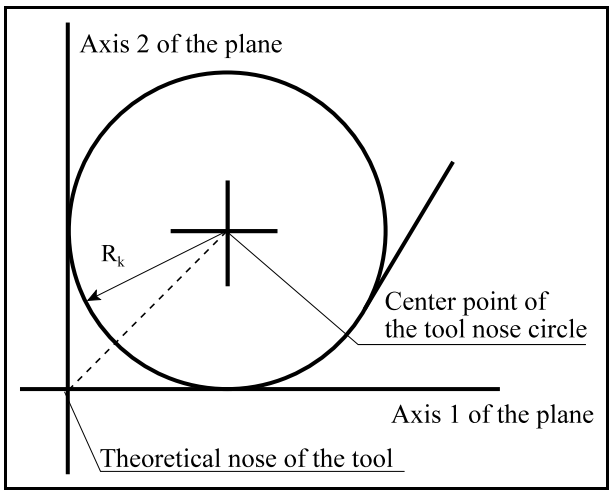


Fig. 15.1-5

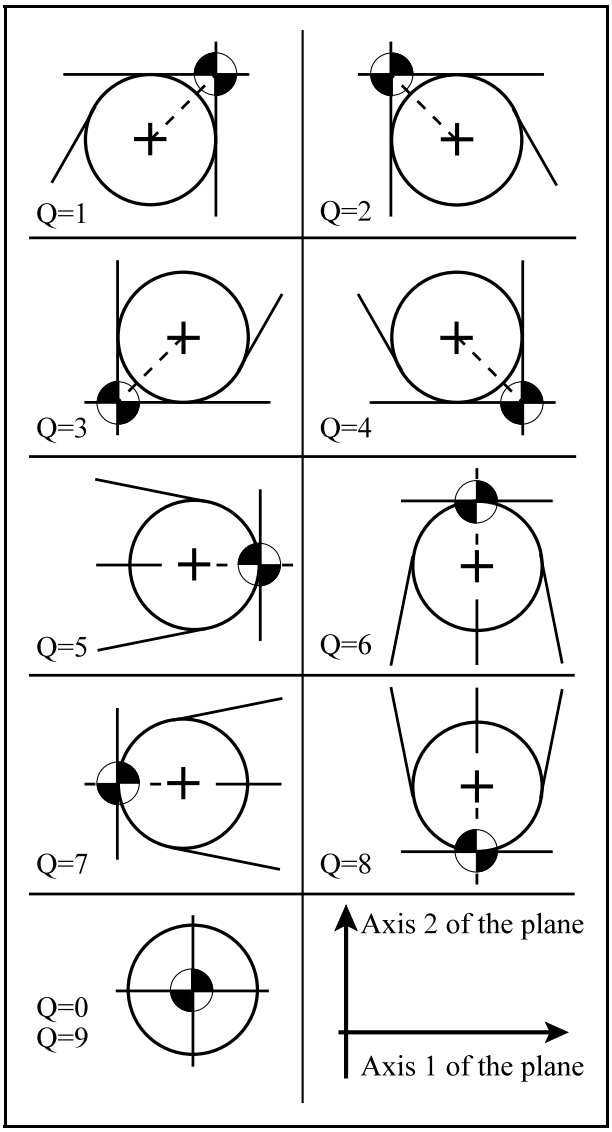


Fig. 15.1-6

If, in the program, an offset value is referred to at the address H or D, ***the sum of geometry and wear values***, as compensation will always be taken into account by the control.

The addresses H and D are modal ones, i.e. the same offset value will be taken into account by the control until it receives another instruction T or D.
It means that if the offset value is taken into account by a instruction H or D, its modification in the offset table either by manual rewriting or by programming G10, will not produce effect on the value taken into account already; recall of the address H or D will update only.
The offset values in the compensation memory will be retained after power-off.

15.2 Modifying the Tool Compensation Values from the Program (G10)

Modifying the compensation values when using only milling machine compensations
The value of #2 TOL of the parameter N1416 Comp. Config on Mills is 0.

The instruction

G10 L P R Q

can be used for modifying the tool compensation values from the program. The instruction G10 is a one-shot instruction. The meaning of t he addresses and their values is as follows:
The compensation value to be modified is specified at the address L.

L=10 means that the setting is related to the geometry value of the length compensation (code H);

L=11 means that the setting is related to the wear value of the length compensation (code H);

L=12 means that the setting is related to the geometry value of the radius compensation (code D);

L=13 means that the setting is related to the wear value of the radius compensation (code D),

L=110 means that it is related to the geometry value of the corner radius r (D code) of the tool,

L=111 means that it is related to the wear value of the corner radius r (D code) of the tool.

The compensation group to be modified can be specified at the address P:

P: the number of the compensation group.

The compensation value can be specified at the address R:

R: the value of compensation.

In the absolute data setting instruction state **G90,** the value written at the address R will be put in the appropriate offset register.

In the incremental data setting instruction state **G91** or in the case of using the operator I, the data written at the address R will be added to the content of the appropriate offset register.

☞ ***Note***: In the case of programmed modification of the ***tool radius compensation***, the value specified at the ***address R*** must be interpreted as a ***radius*** in each case, regardless of the bit state #0 DIA of the parameter N1402 Tool Meas.

Modifying the compensation values when lathe compensations are also used
The value of #2 TOL of the parameter N1416 Comp. Config on Mills is 1.

The instruction

G10 L P X Y Z R Q

can be used for modifying the tool compensation values from the program. The instruction G10 is a one-shot instruction. The meaning of the addresses and their values is as follows:

L=200: specifying the wear value

L=201: specifying the geometric value

The compensation group to be modified can be specified at the address P:

P: the number of the compensation group

The value of the legth and radius compensations can be specified at the address X, Y, Z, R:

X: the value of the lathe length compensation on the X axis

Y: the value of the lathe length compensation on the Y axis

Z: the value of the milling tool length compensation/turning tool length compensation on the Z axis

R: the value of the milling tool radius compensation/turning tool nose radius compensation

Q: code of the tool position (0...9)

The instruction

G10 L P R

can be used for specifying the ***radius of rounding*** of the milling tool at the address.

The meaning of the addresses and their values is as follows:

L=**202**: specifying the wear value

L=**203**: specifying the geometric value

The compensation group to be modified can be specified at the address P:

P: the number of the compensation group

The radius of rounding of the milling tool can be specified at the address R:

R: the radius of rounding of the milling tool

In **G90** absolute data setting instruction state, the value written at the address X, Y, Z and R will be put in the appropriate compensation register.

In **G91** incremental data setting instruction state or in the case of using the operator I, the data written at the address X, Y, Z and R will be added to the content of the appropriate compensation register.

☞ ***Note:*** *In the case of programmed modification of the* ***tool radius compensation,*** *the value specified* ***at the address R*** *must be interpreted as a* ***radius*** *in each case, regardless of state of the bit #0 DIA of the parameter N1402 Tool Meas.*

15.3 Calling the Mill Compensations (G43, G44, G49)

The information given here applies to the use of either the pure milling or the extended tool compensation table, regardless of state of the bit #2 TOL of the parameter N1416 Comp. Config on Mills.

Switching the calculation of length compensation on

Taking the length compensation into account can be switched on by the instructions G43 and G44 .

The instruction ***G43*** shifts the programmer coordinate system ***in positive direction*** along the axis assigned for taking the length compensation into account, by the compensation values given at the address H:

G43: + compensation

The instruction ***G44*** shifts the programmer coordinate system ***in negative direction*** along the axis assigned for taking the length compensation into account, by the compensation values given at the address H:

G44: – compensation

The axis, at which the ***length compensation*** specified at the address H will be taken into account by the control, can be given in the following three ways, according to the bits #0 ZAX and #1 PLN of the ***parameter N1416 Comp. Config on Mills***:

PLN	ZAX	Taking the tool length compensation into account in the blocks G43 and G44
0	0	at all the axes given in the block
0	1	always at the axis Z
1	0	in the case of G17: at the axis Z in the case of G18: at the axis Y in the case of G19: at the axis X

Compensation specified at the address H is ***always*** taken into account by the control ***at the axis Z***

The ***state of the parameter*** N1416 Comp. Config on Mills***: #0 ZAX=1, #1 PLN=0***.
When length compensation is taken into account, the ***main axis*** marked with Z base Axis and specified by the number 3 in the parameter N0103 Axis to Plane is considered to be axis Z.
The instruction

G43 Z H or
G44 Z H

switches the value of ***the length compensation H on, at the axis Z***.
It will be switched on at the axis Z even in the case, when coordinate address Z is not written in the instructions G43 or G44, or another axis address is also programmed in the block.
For example:

```
G43 H2 (the length compensation H2 is taken into account by
     transformation, there is no motion),
```

or

```
G43 X Y Z H3 (positioning along all the three axes, taking
     the length compensation H3 into account along the axis
     Z)
```

The G43 and G44 are modal, i.e. referring ***to a new address H*** in the state G43, G44 enters ***a new compensation*** at the axis Z:

```
G43 H2 (length compensation H2 on the axis Z is entered)
H3 (length compensation H3 on the axis Z is entered)
```

Programming ***H0 cancels*** the value ***of the compensation entered at the axis Z,*** but it leaves the state G43, G44 unchanged:

```
G43 H2 (length compensation H2 on the axis Z is entered)
H0 (length compensation of 0 on the axis Z is entered)
H3 (length compensation H3 on the axis Z is entered)
```

Compensation specified at the address H is taken into account by the control ***at the axis perpendicular to the plane assigned***

The ***state of the parameter*** N1416 Comp. Config on Mills***: #0 ZAX=0, #1 PLN=1***.
When length compensation is taken into account, the ***main axes*** marked with X, Y, Z base Axis and specified by the number 1, 2, 3 in the parameter N0103 Axis to Plane are considered to be axes ***X, Y, Z***.
The instruction

G43 X Y Z H or
G44 X Y Z H

switches the value of ***the length compensation H on, at the axis perpendicular to the plane***

assigned:

in the case of ***G17***: ***at the axis Z***
in the case of ***G18***: ***at the axis Y***
in the case of ***G19***: ***at the axis X***

It will be switched on at the axis perpendicular to the plane assigned even in the case, when coordinate address for the axis is not written in the instructions G43 or G44, or another axis address is also programmed in the block.

For example:

```
G17 G43 H2 (the length compensation H2 is taken into account
     at the axis Z by transformation, there is no motion),
```

or

```
G19 G43 X Y Z H3 (positioning along all the three axes,
     taking the length compensation H3 into account along
     the axis X)
```

The G43 and G44 are modal, i.e. referring ***to a new address H*** in the state G43, G44 enters ***a new compensation*** at the axis perpendicular to the plane assigned:

```
G18 G43 H2 (length compensation H2 on the axis Y is entered)
H3 (length compensation H3 on the axis Y is entered)
```

Programming ***H0 cancels*** the value ***of the compensation entered at the axis perpendicular to the plane assigned,*** but leaves the state G43, G44 unchanged:

```
G17 G43 H2 (length compensation H2 on the axis Z is entered)
H0 (length compensation of 0 on the axis Z is entered)
H3 (length compensation H3 on the axis Z is entered)
```

Referring to the address H after plane change enters the compensation at the new axis, but it leaves the previous one unchanged***:***

```
G17 G43 H2 (length compensation H2 on the axis Z is entered)
G19 H3 (length compensation H2 on the axis Z is entered,
     length compensation H3 on the axis X is entered)
```

Compensation specified at the address H is taken into account by the control ***at the axis defined in the block***

The ***state of the parameter*** N1416 Comp. Config on Mills***: #0 ZAX=0, #1 PLN=0***.

The instruction

G43 q H or
G44 q H

switches ***the value of the length compensation H*** on***, at the axes defined in the block G43 or G44***.

The meaning of the address q: the tool length compensation is effective at the axis q (q: X, Y, Z). The assigned axis can be either a main axis or a parallel axis, too.

For example:

G43 Z H2 (the length compensation H2 is taken into account at the axis Z);
G43 W H3 (the length compensation H3 is taken into account at the axis W parallel with the axis Z).

If ***several axes*** are assigned in a block, the tool length compensation will be taken into account ***at all the axes assigned***!

For example:

```
G44 X120 Z250 H27 (H27 is entered at X and at Z, too)
```

Tool length compensation can be called ***for several axes.*** For example***,*** if turning is to be executed in the plane YZ, the tool overhang in the directions Y and Z can be taken into account

in the following way:

```
G43 Z250 H15 (length compensation H15 is taken into account
     at the axis Z),
Y310 H16 (length compensation H16 is taken into account at
     the axis Y)
```

The effect of ***G43 and G44*** is ***modal*** until the control receives another instruction from this group. If the compensation value is changed by referring to a ***new address H and the same axis addres***, the old value will be cancelled, and the new value will be validated:

```
G43 Z100 H1  (it moves to the machine position Z=110)
...
Z100 H2 (it moves to the machine position Z=120)
```

The state of G43 is inherited by the block Z100 H2, therefore the new compensation H2 will be taken into account along the axis Z.

The compensation H0 together with programming an axis cancels the compensation at that given axis. Continuing the example above:

```
G43 Z100 H2 (it moves to the machine position Z=120)
...
Z100 H0 (it moves to the machine position Z100)
```

If compensation H is programmed without specifying an axis address, this will change the modal value H only. Continuing the example above:

```
G43 Z100 H2 (it moves to the machine position Z=120)
H0 (axis address is not specified)
Z100 (it does not cancel the compensation at the axis Z, it
     stays at the machine position Z=120)
```

G43 and G44 in absolute and incremental programming

In the case of ***absolute data specification*** the instruction

```
G43 G0 G17 G90 Z50 H5
```

moves the tip of the tool to the point of coordinates Z50, taking the compensation H5 into account.

In the case of ***incremental data specification*** the instruction

```
G43 G0 G17 G91 Z10 H1
```

or

```
G44 G0 G17 G91 Z10 H1
```

shifts the starting point by the compensation value, and then the incremental displacement will be counted from here, from the starting point compensated.

Accordingly, due to the instruction

```
G43 G0 G17 G91 Z0 H2
```

the axis Z does not execute motion; only the position will get the value adequate to the new compensation.

Switching the length compensation calculation off

The instruction

G49

will switch the tool length compensation off on all the axes either by motion if axis address is also programmed in the block, or by transformation, if motion is not programmed in the block. At switch on, at reset or after the program end, the bits #5 G43 és #6 G44 of the parameter N1300 DefaultG1 determine which code (G43, G44, G49) is valid. If both bits are 0, then code G49 is valid, i.e. compensation is not entered.

The example below illustrates a simple drilling operation with taking the tool length compensation; the length of the drilling tool is H1=400.

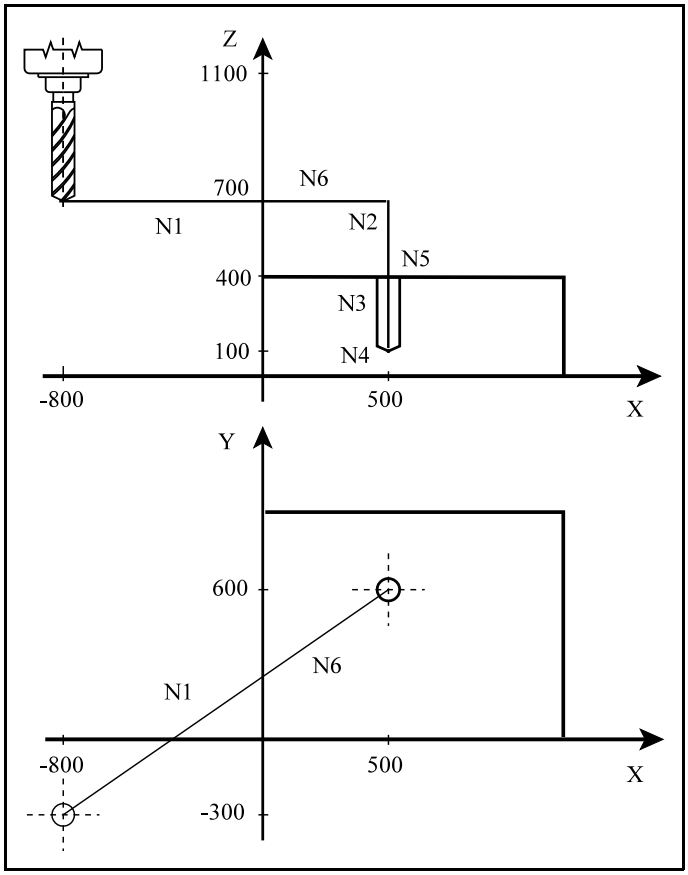


Fig. 15.3-1

```
N1 G90 G0 X500 Y600 (positioning in the plane X, Y)
N2 G43 G17 Z410 H1  (moving to the coordinate Z410 with length
                    compensation H1)
N3 G1 Z100 F180     (drilling to the coordinate Z100 at feed
                    rate F180)
N4 G4 P2            (dwelling for 2 seconds)
N5 G0 Z1100 H0      (pulling the tool out by switching the
                    length compensation off, the tip of the tool
                    is in the coordinate Z700)
N6 X-800 Y-300      (returning with rapid traverse in the plane
                    X, Y)
```

15.4 Calling the lathe compensations (G43.7)

The information given here applies to the use of the extended tool compensation table, i.e., when the bit #2 TOL of the parameter N1416 Comp. Config on Mills is 1.

In the milling machine channel, the instruction

G43.7 Hh

calls the sum of the geometry and wear compensations Xg+Xw, Yg+Yw and Zg+Zw specified at the address H to all three (X, Y, Z) axes.

Compensation is called ***without taking*** the bits PLN and ZAX of the parameter N1416 Comp. Config on Mills ***into account***.
In the course of calling the compensation, the control ***disregards*** all the bits of the parameter N1414 Comp. Config on Lathes. Thus, logically, it does not manage the lathe compensation table 2 either.
The G43.7 is a member of the group 8 of G codes.

Cancelling the compensation is executed by the instruction

G49.

The instructions G43, G44, G43.1, G43.4, G43.5 change back to the use of the milling machine compensation.

An example:

```
...                        (Milling)
...
G49 H0 D0                  (Cancelling milling machine
                           compensations)
G10.9 X1                   (Diameter programming at X)
T5 M6                      (Calling lathe tool)
G54 G0 X200 Y0 Z100
G43.7 H5                   (Calling H5 lathe compensation)
G18 G95 G0 X101 Z1 S3000
G96 S400 P1 M3             (Constant cutting speed for X)
G92 S200
G77.7 X90 Z-30 R-3 F0.5    (Turning cycle)
X80
X70
G49 G0 X200 Z100 H0 D0     (Cancelling the compensations)
...
G10.9 X0                   (Radius programming at X)
G43                        (Use of the milling machine
                           compensations)
...                        (Milling)
```

15.5 Tool radius compensation (G40, G41, G42)

In order that an optional contour can be milled around, and the contour points corresponding to the drawing can be given in the program independently of the size of the tool used, the center point of the tool has to be guided by the control parallel with the programmed contour in a distance of the tool radius from it. The distance of the center point of the tool from the programmed contour during guidance is determined by the control according to value of the tool radius compensation entered on the called compensation number D.

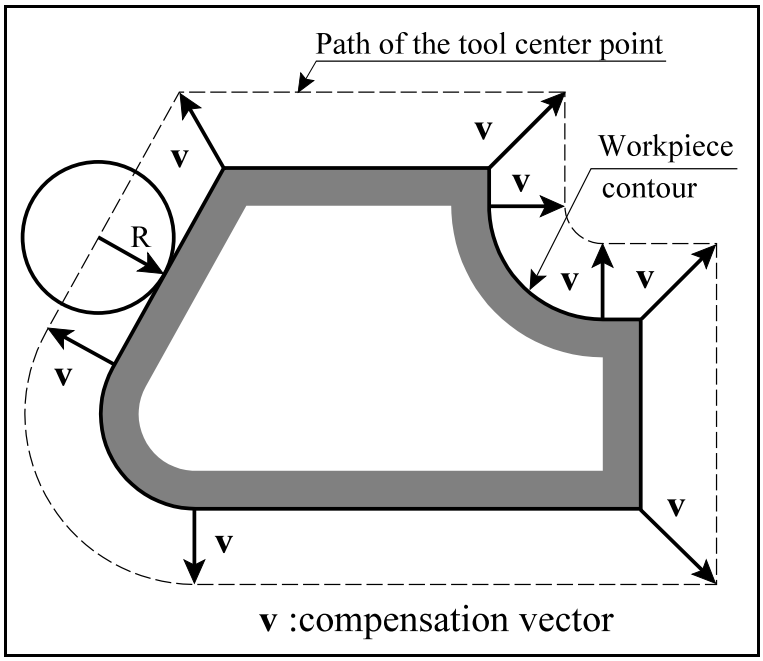


Fig. 15.5-1

The ***compensation vector*** is a ***two-dimensional vector*** recalculated by the control in each block, and the programmed displacements are modified by the control with the compensation vectors effective at the beginning and at the end of the block. The length and the direction of the compensation vectors obtained depends on ***the compensation value called at the address D*** and the geometry of the transition between the two blocks.

The control computes the compensation vectors ***in the plane selected by the instructions G17, G18 and G19***.This is the plane of tool radius compensation. Motions out of this plane are not affected by the tool radius compensation. For example: if, in the state G17, the plane X, Z is selected, the compensation vectors will be computed in the plane X, Z. In this case, the motion in the direction Z is not affected by the tool radius compensation.
It is not permitted to change compensation plane while tool radius compensation is being computed. Should it is tried, the control will send an error message.
In the case of defining the compensation plane not along the axes being in the main plane, secondary axes have to be defined in the parameter field as parallel axes. For example, if U is defined as parallel axis and tool radius compensation should be used in the plane Z, U, the plane will have to be assigned by specification G18 U__ Z__ .

Computing the tool radius compensation can be switched on and off from the program:

G40: **cancelling** calculation of the tool radius compensation

G41: calculation of the tool radius compensation **to the left according to the motion direction**

G42: calculation of the tool radius compensation **to the right according to the motion direction**

The instructions ***G40, G41, G42*** are ***modal*** ones. After start, at the program end and due to reset the control comes to the state G40.

The instruction G41 or G42 starts the compensation calculation. The programmed contour is tracked by the tool from the left in the state G41, and from the right in the state G42, according to the motion direction. ***The applied values of tool radius compensation must be given at the address D. Compensation calculation is carried out for the interpolation motions G00, G01, G02, G03***.

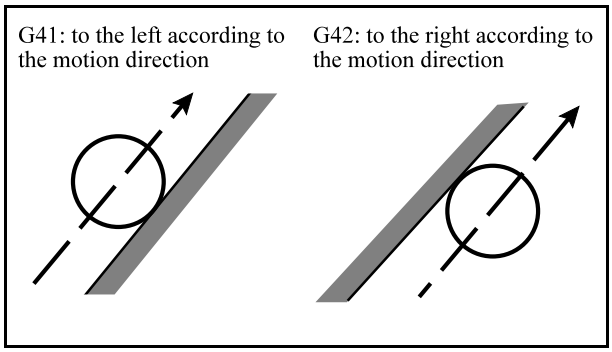


Fig. 15.5-2

The findings mentioned above are valid in the case of positive tool radius compensation. However, ***the value of tool radius compensation can be negative, too***. It has practical meaning, for example, in the case when the same subprogram is to be used for contouring a female workpiece and then a male workpiece fitting together. Another way of doing this is that G41 is used for machining the female workpiece and G42 is used for machining the male workpiece. But this changeover will not have to be edited into the program if the female one is machined with positive tool radius compensation and the male one is machined with negative tool radius compensation, for example. In this case, the path of the tool center point will be reversed by the programmed G41 or G42:

	Tool radius compensation: **positive**	Tool radius compensation: **negative**
G41	from the left	from the right
G42	from the right	from the left

☞ ***Note***: For the sake of simplicity, hereafter, positive tool radius compensation is always assumed in descriptions and figures.

The instructions ***G40*** or ***D00*** cancel the compensation calculation. The difference between the two instructions is that the instruction D00 cancels the length of the compensation vector only, and leaves the state G41 or G42 unchanged. If, after that, the address D differing from zero is referred to, the compensation vector will be calculated using the new tool radius, depending on the state G41 or G42.
If, however, the instruction G40 is used, any reference to the address D will be ineffective until G41 or G42 is programmed.
There are specific rules for starting or cancelling the tool radius compensation; they will be detailed in the following.

Tool radius compensation instructions are executed by the control in automatic or manual data input mode only. In manual mode, it is not effective in case of single blocks. The reason for this is as follows. In order that the control can calculate the compensation vector in the endpoint of a block, it is necessary for the control to read in the next block containing the motion in the selected plane. The compensation vector depends on the transition between the two blocks. It is apparent that preliminary processing of several blocks is required for calculation of the compensation vector.

Before dealing with details of the compensation calculation, an auxiliary data is to be introduced. It is the angle α between the tangent lines drawn to the meeting point of two curve segments, i.e. to blocks. The direction of the angle α depends on whether the contour is compassed from the left or from the right.

The control selects the strategy of deflecting at the intersection points according to the angle α. If α>180°, i.e. the tool works inside, the control will compute an intersection point between the two segments. If α<180°, i.e. the tool moves outside, the control will insert additional straight segments for moving.

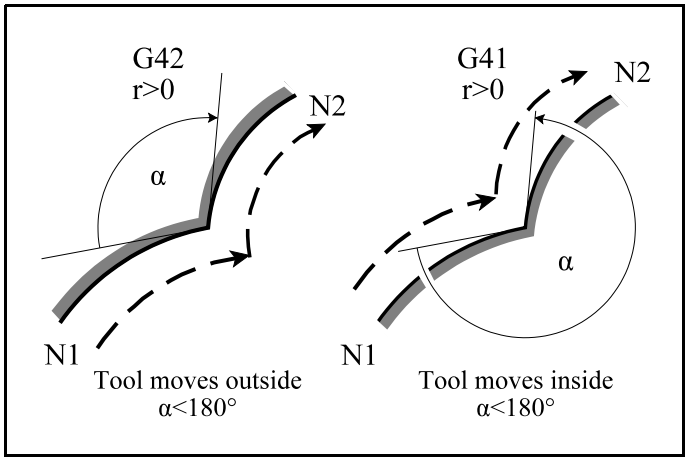


Fig. 15.5-3

Hereafter, this manual deals with cases of calculation of tool radius compensation mainly from the point of view of milling operations. For turning operations, cases of calculation of tool radius compensation are dealt with in the book Control for Lathes Programmer's Manual.

15.5.1 Start up of the tool radius compensation. Moving to the contour

Due to the instruction G41 or G42, the control enters in the mode of tool radius compensation from the state G40. It takes the value of compensation from the compensation cell specified at the address D.
The control comes to the state ***G41*** or ***G42*** in the block of ***positioning G0*** or in the block of ***linear positioning G1*** only.
If start up of compensation calculation is attempted in circular block (G2, G3), the control will send an error message.
The ***strategy of moving to the contour*** will be chosen by the control in the only case of change over ***from the state G40 to the state G41 or G42***. In other words, if compensation is cancelled by D0 and then it is restarted by Dn (n is a number different from 0), not the strategy of moving to the contour will be selected by the control.
The basic cases of starting up the compensation according to the angle α and the possible transitions (linear - linear and linear - circular) are illustrated below. The figures are drawn for the case of G42 and ***positive tool radius compensation*** is assumed.
☞ *Note*: Here and hereafter, meanings of the marks in the figures are the following:

r: value of tool radius compensation;
L: linear segment;
C: circle arc;
S: in the block by block mode, the point of stop;
dash line: the path of the center point of the tool;
continuous line: the programmed path.

The basic cases of starting up the tool radius compensation:

G40	G40
G42 G1 X_ Y_ D_	G42 G1 X_ Y_ D_
X_ Y_	G2 X_ Y_ R_

Positioning to an inner corner: 180°<α<360°

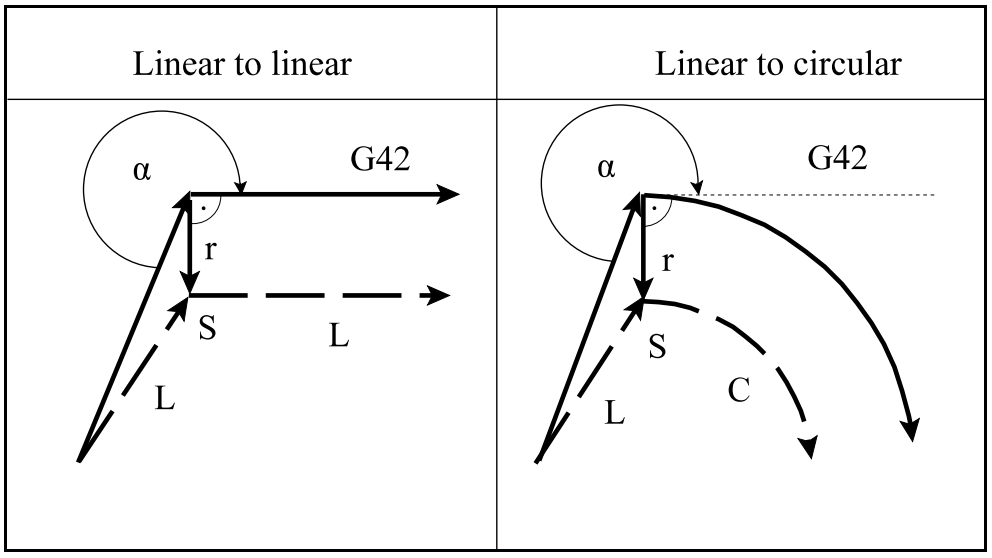


Fig. 15.5.1-1

Positioning to an outer corner at an obtuse angle: 90° ≤ α ≤ 180°

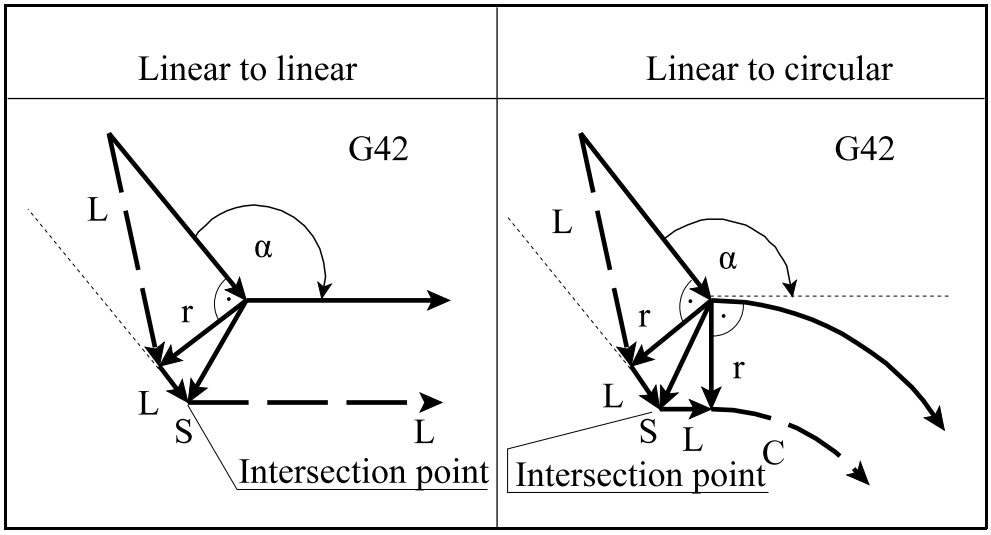


Fig. 15.5.1-2

Positioning to an outer corner at an acute angle: 0° ≤ α < 90°

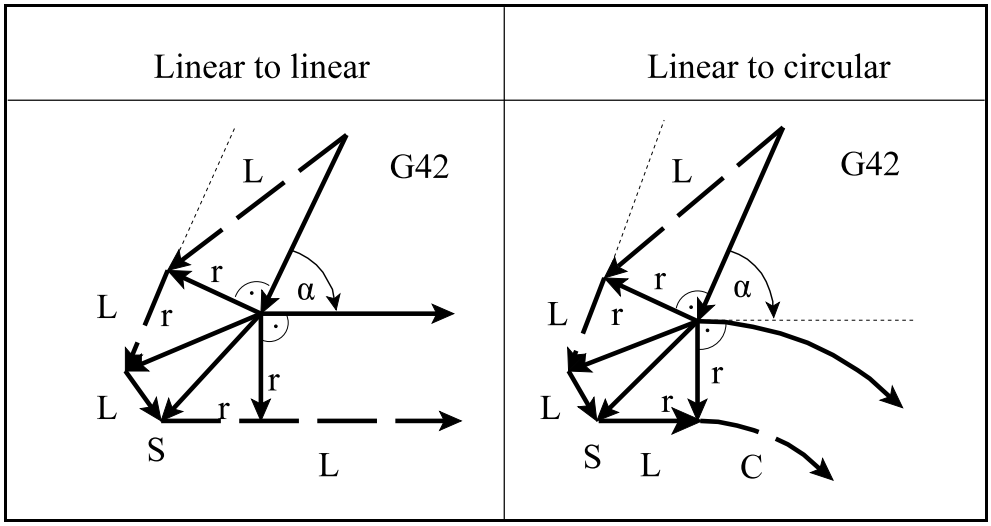


Fig. 15.5.1-3

An example:
A circle with the radius R60 is to be machined internally. Positioning to the circle with the radius R60 is executed along a circle arc with the radius R50.
The block N80 starts up the tool radius compensation with positioning to the point of X10 Y50 in such a way that a perpendicular with a length of tool radius (r) is let fall to the starting point of the circle arc with the radius R50 in the block N90:

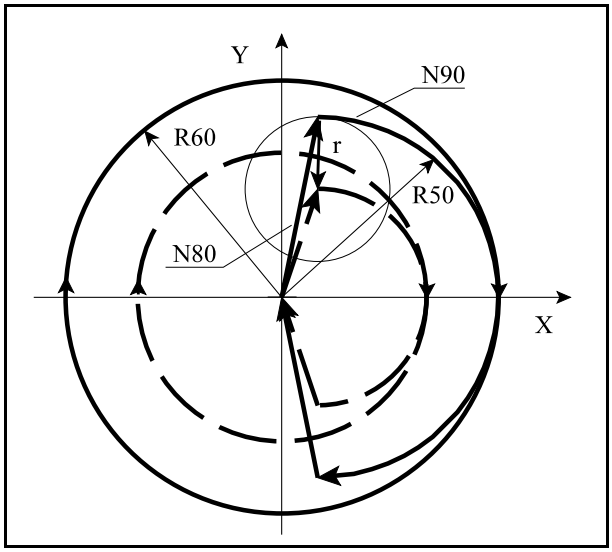


Fig. 15.5.1-4

```
G54 G17
...
N50 G0 X0 Y0
N60 G43 Z1 H1
N70 G1 Z-2 F1000
N80 G42 G0 X10 Y50 D1
N90 G2 X60 Y0 R50
N100 G2 I-60
N110 G2 X10 Y-50 R50
N120 G40 G0 X0 Y0
...
```

A square with side length of 60 mm is to be milled around. The control defines the endpoint of the block N50 in such a way that it lets fall a perpendicular with a length of tool radius (r) to the starting point of the block N60 and moves there.

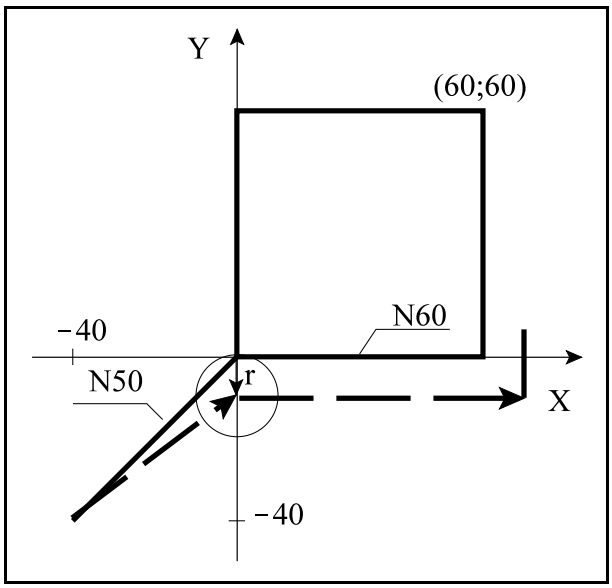


Fig. 15.5.1-5

```
N10 G54 G17 G49 M3 S500
N20 G0 X-40 Y-40
N30 G43 Z2 H1
N40 G1 Z-5 F500
N50 G42 G0 X0 Y0 D1
N60 G1 X60
N70 Y60
N80 X0
N90 Y0
N100 G40 G0 X-40 Y-40
N110 M30
```

Starting up the tool radius compensation without programming of motion

If ***tool radius compensation is started up*** (G41, G42) in a block in which there are ***none of the addresses of the axes belonging to the selected plane*** (as in the example below, neither the address X nor the address Z is given in the block N50), motion in the block N50 will not be executed, but, in the block N60, a tool-radius-long displacement marked with 1 in the figure will be executed by the control first, and then the motion marked with 2. Both motions will be

executed using rapid traverse.
In this case, intersection point is calculated between the blocks N60 and N70, and the center point of the tool will be guided in accordance with it by the control. It is not good, because the corner X0 Y0 will not be machined! Correction can be done by merging the blocks N50 and N60.

```
N10 G54 G17 G49 M3 S500
N20 G0 X-40 Y-40
N30 G43 Z2 H1
N40 G1 Z-5 F500
N50 G42 G0 D1
N60 X0 Y0
N70 G1 X60
N80 Y60
N90 X0
N100 Y0
N110 G40
N120 G0 X-40 Y-40
N130 M30
```

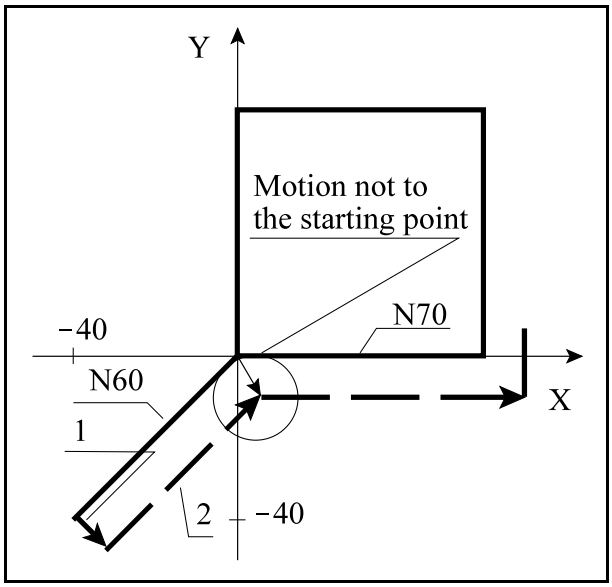


Fig. 15.5.1-6

If ***tool radius compensation is started up*** (G41, G42) in a block in which ***there is no displacement along none of the axes belonging to the selected plane*** (as in the example below, in the block N50 the address X is filled, but because of the incremental 0 belonging to it there is no motion), a tool-radius-long displacement marked with 1 in the figure will be executed by the control at the feed rate in the block N50, and then the motion marked with 2 in the block N60.

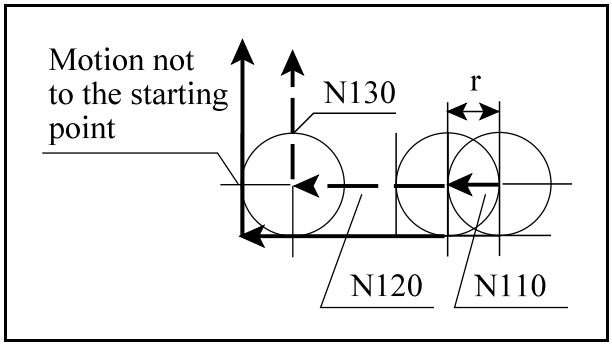


Fig. 15.5.1-7

In this case, the control guides the center point of the tool while calculating intersection point between the blocks N60 and N70. It is not good, because the corner X0 Y0 will not be machined! Correction can be done by merging the blocks N50 and N60.

```
N10 G54 G17 G49 M3 S500
N20 G0 X-40 Y-40
N30 G43 Z2 H1
N40 G1 Z-5 F500
N50 G42 G0 G91 X0 D1
N60 G90 X0 Y0
N70 G1 X60
N80 Y60
N90 X0
N100 Y0
```

```
N110 G40 G91 Y0
N120 G0 G90 X-40 Y-40
N130 M30
```

There is no displacement in the block succeeding the start up of the tool radius compensation

If, in the block succeeding the start up of the tool radius compensation, there is displacement 0 in the selected plane, as in the block N60 of the program below, the control will put the compensation vector at right angels to the endpoint of the block carrying out the start up (N50) and it will move there. Then it lets fall a perpendicular to the starting point of the block carrying out the succeeding motion (N70) too, it moves on up to this point, and then moves the tool onward, in parallel.
This can cause overcutting on the workpiece, so these cases have to be avoided! Correction can be done by cancelling the block N60.

```
N10 G54 G17 G49 M3 S500
N20 G0 X-10 Y-40
N30 G43 Z2 H1
N40 G1 Z-5 F500
N50 G42 G0 X0 Y0 D1
N60 G1 X0
N70 X60
N80 Y60
N90 X0
N100 Y0
N110 Y0
N120 G40 G0 X-40 Y-10
N130 M30
```

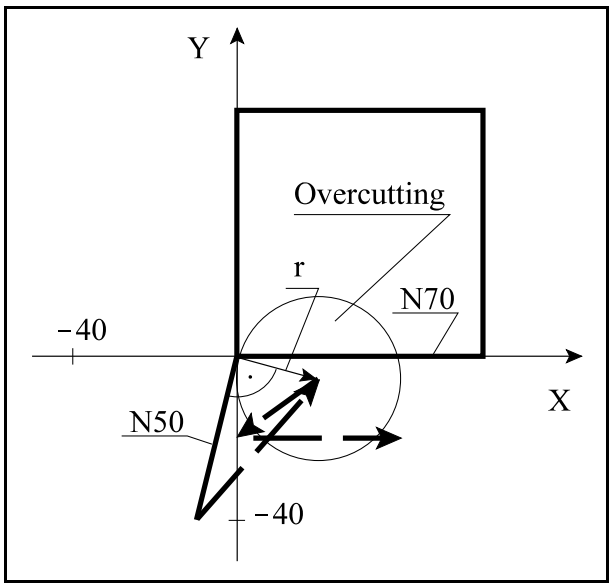


Fig. 15.5.1-8

Starting up the tool radius compensation with calculation of intersection point

If, in the block carrying out start up (G41 or G42), a value is assigned to I, J and K, but only to those that are in the selected plane (for example, to I and K in the case of G17), the control will move to the intersection point defined by the succeeding block and by I, J and K, taking the tool radius compensation into account. The value of I, J and K is always incremental, and the vector defined by them points at the endpoint of the that block in which it was programmed. It is a useful thing in the case of positioning to an inner corner, for example.

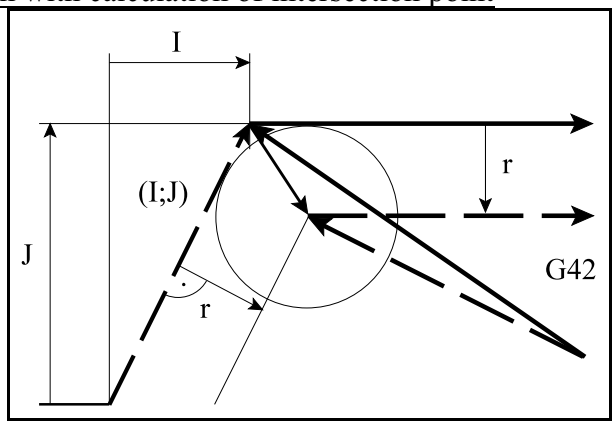


Fig. 15.5.1-9

An example:
A hexagon with side length of 100 mm is to be milled internally. The tool has to be positioned to the corner point with the coordinates of X100 Y0.The block N50 of the program below positions to the contour using calculation of intersection point, with specification of I and J. The coordinates of I and J were calculated on the basis of the displacement executed in the block N110 of the program.

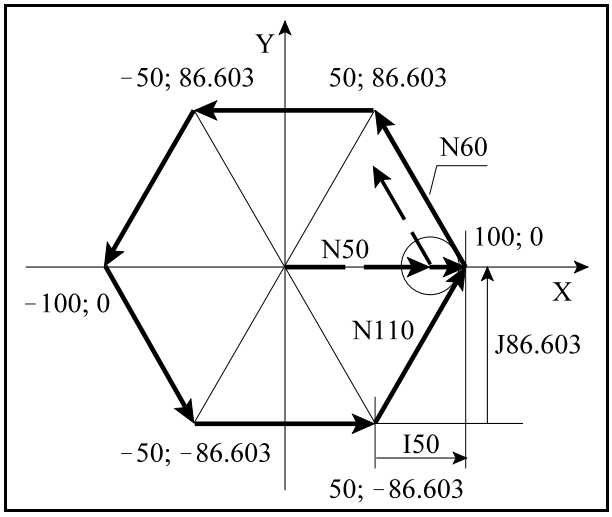


Fig. 15.5.1-10

```
N10 G54 G17 M3 S300
N20 G0 X0 Y0 Z100
N30 G43 Z5 H1
N40 G1 Z-5 F1000
N50 G41 G0 X100 Y0 I50 J86.603 D1
N60 G1 X50 Y86.603
N70 X-50
N80 X-100 Y0
N90 X-50 Y-86.603
N100 X50
N110 X100 Y0
N120 G40 G0 X0 I-50 J86.603
N130 M30
```

Cases of positioning using intersection point

In the case of specification I, J and K, the control always calculates intersection point regardless of whether inner or outer corner is to be machined.

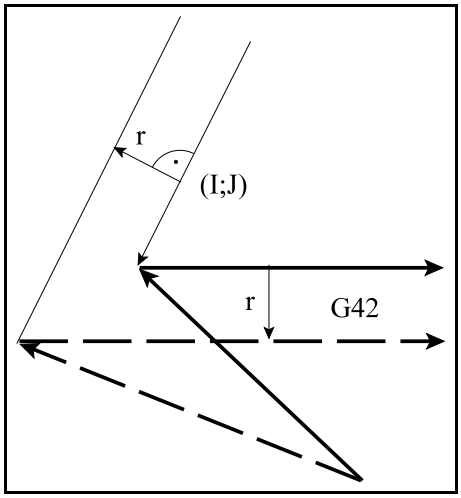


Fig. 15.5.1-11

If the control does not find intersection point, it will position to the starting point of the succeeding block at right angles.

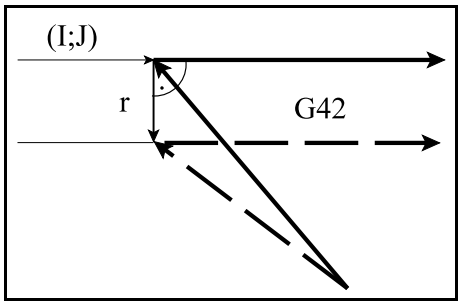


Fig. 15.5.1-12

15.5.2 Calculation of Tool Radius Compensation in Offset Mode. Movement on the Contour

In the course of calculation of tool radius compensation in offset mode, the compensation vectors will be calculated continuously between the blocks G0, G1, G2 and G3. In order that these vectors can be calculated, the blocks have to be continuously read ahead.
The control reads ahead + 2 blocks, the number of which is given in the parameter N1404 BK No. Interf. This means, that the compensation calculation will remain continuous yet if between two motion blocks belonging to the selected plane there is an other block numbered in parameter, for example function, dwell, motion outside of the plane etc.
Continuance may be interrupted by those codes G and functions which enforce buffer emptying, i.e. suspend reading ahead of the blocks.

The basic cases of calculation of tool radius compensation in offset mode:

Calculation of intersection point in the case of inner corners: 180°<α<360°

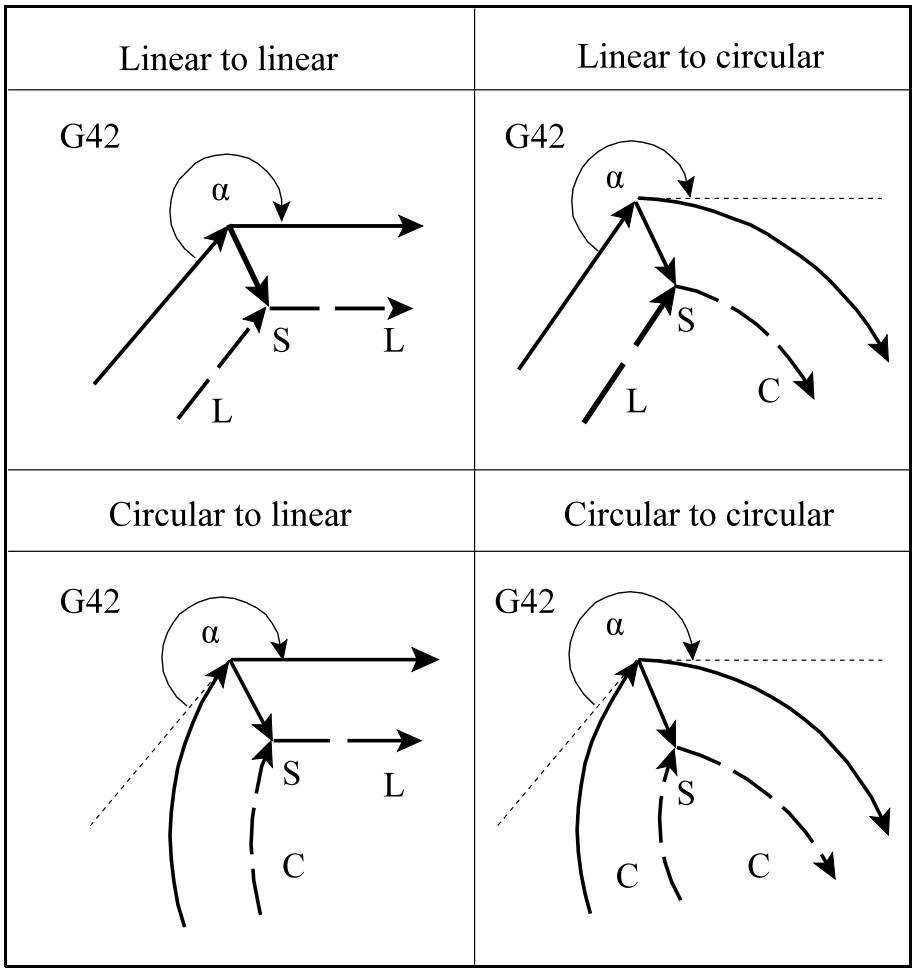


Fig. 15.5.2-1

Going around outer corners having obtuse angle: $90° \leq \alpha \leq 180°$

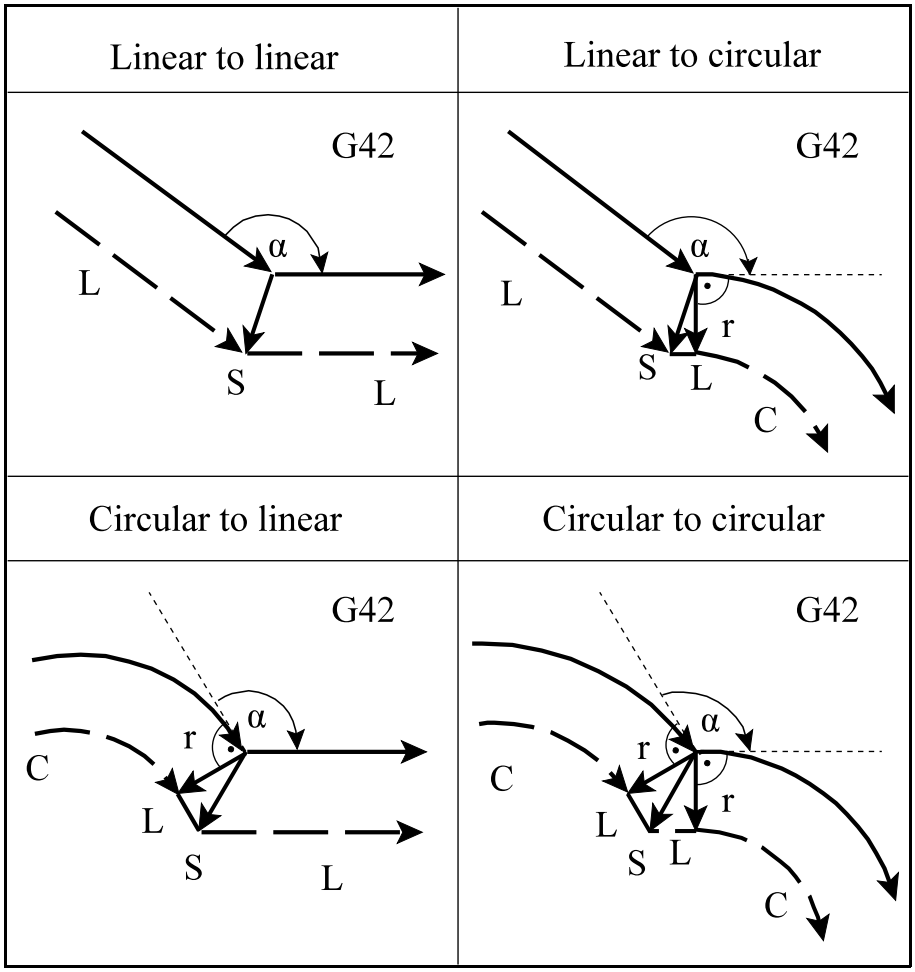


Fig. 15.5.2-2

Going around outer corners having acute angle: 0°≤α<90°

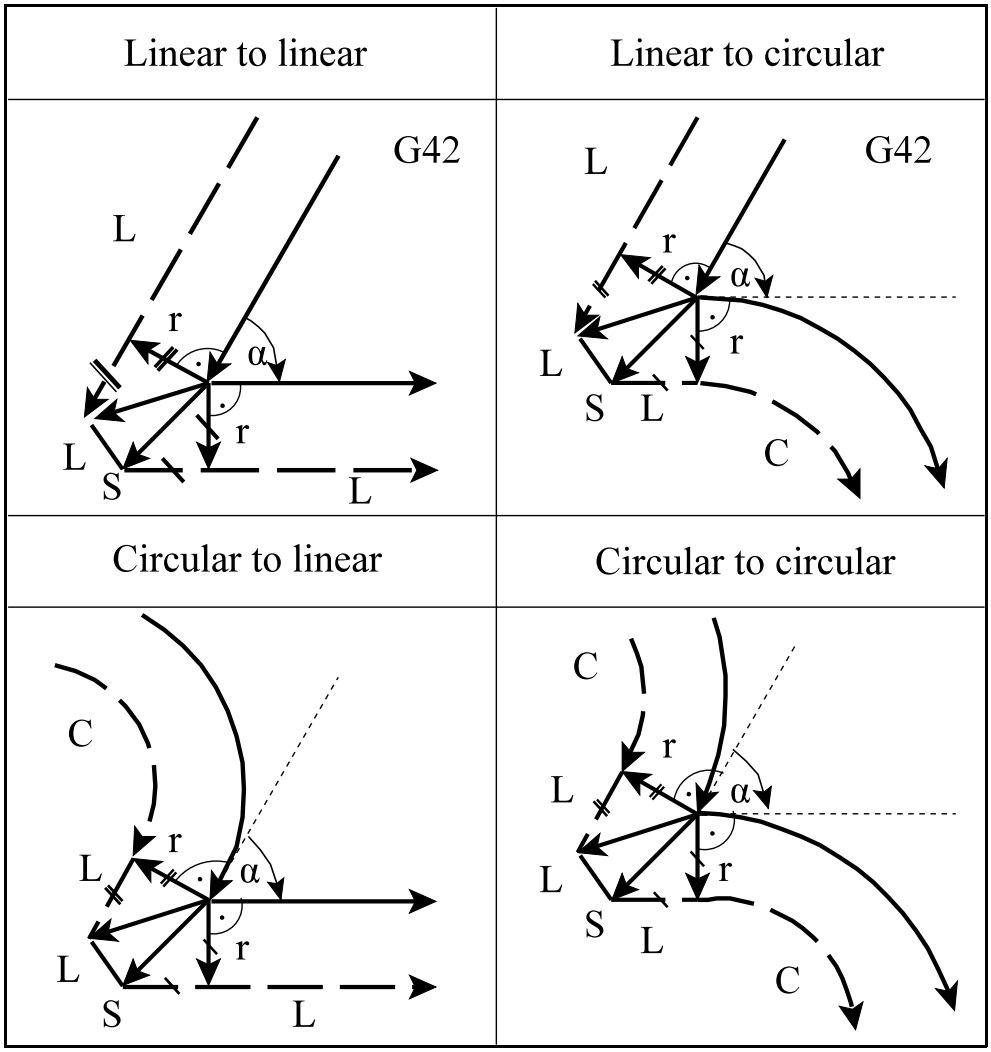


Fig. 15.5.2-3

There is no intersection point in case of inner corners

It can happen, that there will not be intersection point at certain values of tool radius. In this case, the control will stop during execution of the previous block and will send the error message '2047 No intersection G41, G42'.

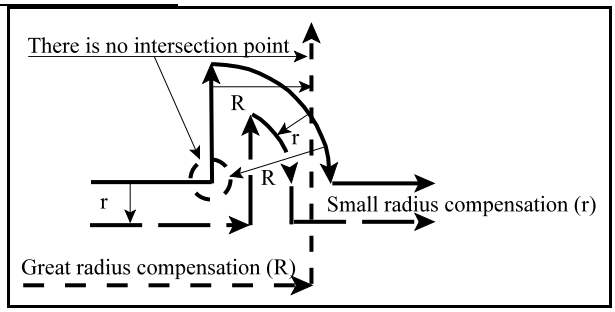


Fig. 15.5.2-4

Zero displacement in the selected plane takes place

If, when G41 or G42 is effective, zero displacement in the selected plane is programmed in one of the blocks or zero displacement takes place as in the block N120 of the example below, the following will happen. The control lets fall perpendicular vectors to the endpoint of the previous block (N110) and to the starting point of the succeeding block (N130) the length of which is equal to the tool radius compensation, and then it connects these two vectors by linear interpolation. In these cases, increased attention is required because unintended overcutting or, in case of circle, distortion can be caused.

For example:

```
... G91 G17 G42 ...
N110 G1 X40 Y50
N120 X0
N130 X90
N140 X50 Y-20
...
```

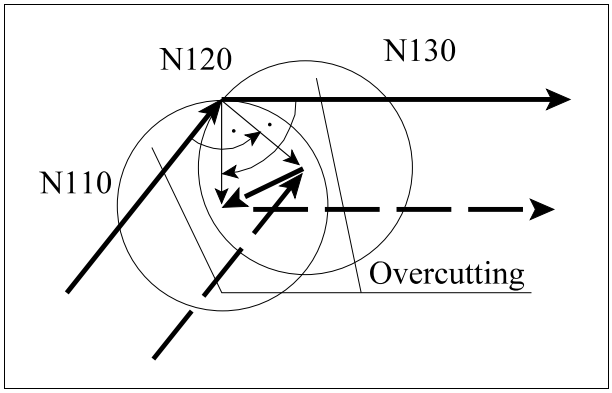


Fig. 15.5.2-5

Tool radius compensation for a spiral or a circle with variable radius

If tool nose radius compensation is used for a spiral or a circle with variable radius, the control calculates compensation vector(s) in the starting point of the circle for an imaginary circle, the radius of which is equal to the starting point radius of the programmed circle (in the figure R1=50) and the center point of which coincides with the center point programmed (X0, Z0). The control calculates compensation vector(s) in the endpoint of the circle for an imaginary circle, the radius of which is equal to the endpoint radius of the programmed circle (R2=20) and the center point of which coincides with the center point of the circle programmed.

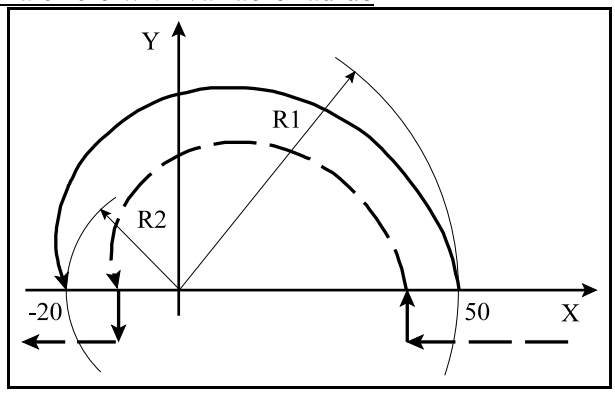


Fig. 15.5.2-6

An example:

```
G0 G17 G41 ...
G1 X50
G3 X-20 I-50 L1
G1 X-30
...
```

A new tool radius compensation is called at address D when G41or G42 is effective

When calculation of tool radius compensation is active (G41 or G42), a new radius compensation value can be called at address D. When the radius value changes sign, it means change in direction on the contour what is dealt with in the subchapter 'Directional change during calculation of tool radius compensation'. If the radius value does not change sign, the process will be the following.

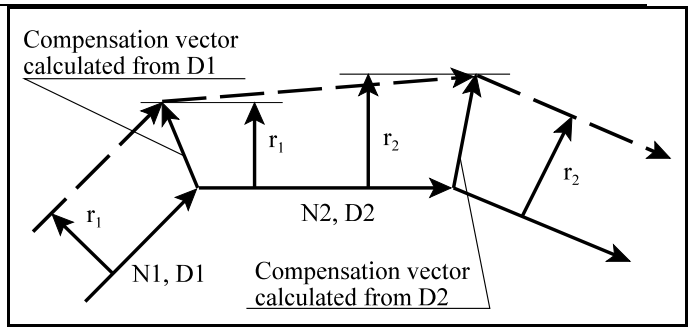


Fig. 15.5.2-7

The compensation vector (in the figure D2) calculated with ***the new compensation value*** will be calculated to ***the endpoint of that block*** (N2) in which the new address D2 is programmed. Since in the starting point of this block the compensation vector was calculated with the previous radius value (D1), the path of the tool center point will not be parallel with the programmed path. The new value of tool radius compensation can also be called at address D in circular block, but in this case the tool center point will move along a circle arc with variable radius.

The specific case of the abovementioned is when ***compensation cancelled by D0*** and started by D1 while calculation of tool radius compensation is active. The example below illustrates how the tool path will differ from each other if compensation is ***started by G41 or G42 and cancelled by G40***, or, if ***starting and cancelling of the compensation*** is carried out ***by calling address D***:

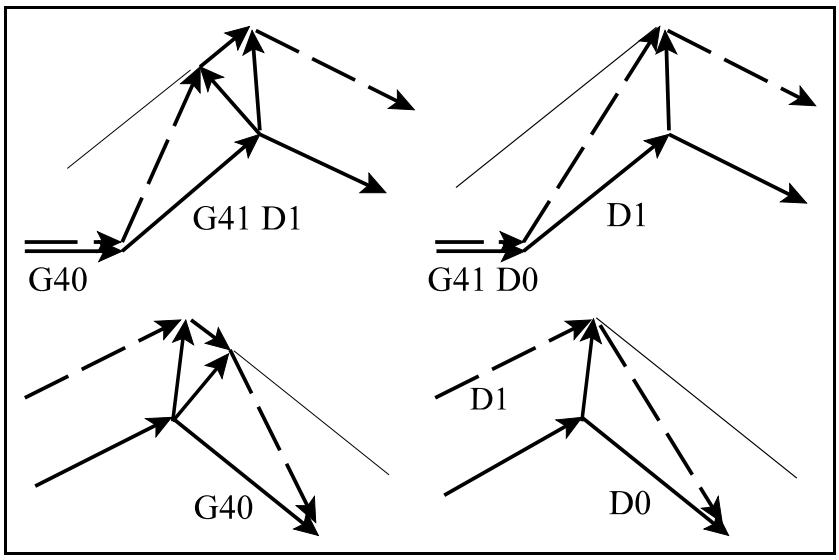


Fig. 15.5.2-8

Omitting the corner movements

When moving around acute-angled or obtuse-angled corners, two or more compensation vectors can be produced. If their endpoints almost coincide with each other, there barely will be motion between the two point.
In the case, when the distance between the two vectors on both axes is less than the value set in the parameter N1405 DELTV, the vector shown in the figure will be omitted and the path of the tool will be changed as it is illustrated in the figure.

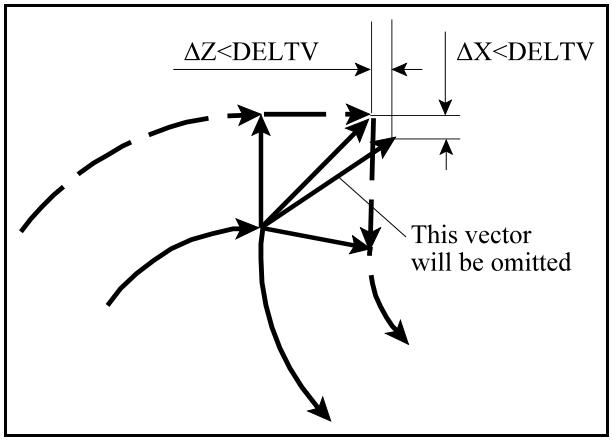


Fig. 15.5.2-9

☞ *Note*: If the value of the parameter DELTV is unduly great while moving around outer acute-angled corners, the corner can be damaged by the tool!

In the selected plane, there is no motion instruction in several consecutive blocks

In order that the control manages tool radius compensation in a proper way, for example, it can be able to calculate intersection point between the endpoint of the block read in and the starting point of the succeeding contour block, ***the blocks have to be read ahead and preprocessed***. The preprocessed blocks get into the block buffer..
In practice, it could be necessary to program a block not containing motion or a block containing motion performed not in the selected plane, in between two block of planar motion. These, for example, can be the following:

functions: M, S, T
dwell: G4 P
motion performed outside of the selected plane: (G17) G1 Z
subprogram call: M98 P, etc.

This means, that if there are any other blocks between two motion blocks belonging to the selected plane, for example, functions, dwell motion outside of the plane etc., the compensation calculation will remain continuous yet until the block buffer will become full.
When the ***buffer becomes full***, the control will send the message

'2090 Unable to continue radius compensation. Buffer full'

at the beginning of the last motion block belonging to the selected plane.

Buffer emptying function is programmed when G41 or G42 is effective

Continuance of compensation calculation, i.e. ***reading ahead*** of the blocks ***will be interrupted*** by those codes G and functions (for example certain functions M etc.) which enforce ***buffer emptying***.
When in the course of reading ahead the control reads in such a code, it suspends reading ahead of further blocks and waits until the block buffer becomes empty, i.e. until all the blocks in the buffer are executed. It results in ***suspending calculation of tool radius compensation*** by the control. Then, the control executes the function, and after that it begins reading in and buffering the succeeding block.
The codes G suspending calculation of tool nose radius compensation are the following:

G22, G23,

G54-G59, G54.1, ...,
G52, G92,
G53,
G28, G30
G43

The code H of calling the length compensation also suspends compensation calculation:

Hnn

The functions M suspending calculation of tool radius compensation are the following:

M0, M1, M2, M30

In addition to the functions M above, functions M and groups of codes M executing buffer emptying can be assigned in parameters, too. Functions S, T and auxiliary functions (A, B, C, U, V, W) besides functions M can also be assigned for buffer emptying.

The functions executing buffer emptying are determined by the the builder of the machine tool, and they are contained in the manual of the machine!

In the case, when G41 or G42 is effective, and the abovementioned codes G, H, or functions are programmed between two motion block, the control cancels the compensation vector at the endpoint of the previous block, executes the instruction and then restores the vector at the endpoint of the succeeding motion block.

For example:

```
...G91 G17 G41...
N110 G1 X80 Y-50
N120 G92 X0 Y0
N130 X80 Y50
...
```

In the example above, the control emptyes the buffer. The situation is similar to the cases of the other buffer emptying codes, too.

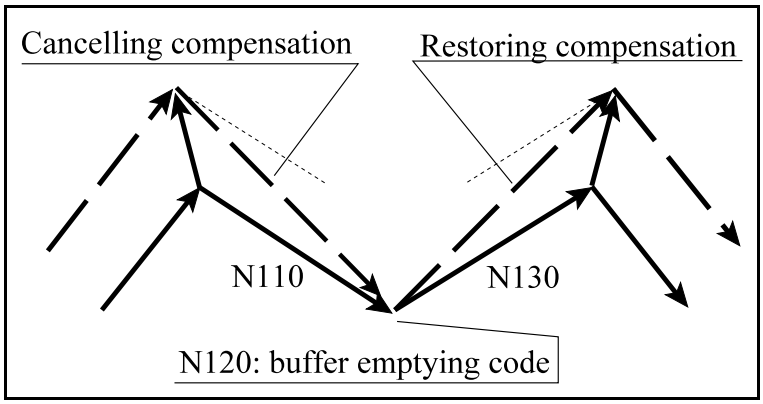


Fig. 15.5.2-10

Conditional block skip is programmed when G41 or G42 is effective

At the bit state ***#4 CBB=0*** of the parameter N1337 Execution Config, the conditional block instruction (blocks beginning with slash /) ***suppresses*** reading the block ahead. In this case, when the ***G41, G42*** is effective, the contour ***will be distorted***, however, for effectiveness, it is enough ***to turn*** the conditional block switch ***during execution of the previous block***.

At the bit state ***#4 CBB=1*** of the parameter N1337 Execution Config, the conditional block instruction (blocks beginning with slash /) ***does not suppress*** reading the block ahead. In this case, when the ***G41, G42*** is effective, the contour ***will not be distorted***, however, for sure effectiveness, the conditional block switch ***has to be set before execution of the program***.

For the effect of the parameter setting, please see the subchapter Conditional block skip.

An example:

```
...G91 G17 G41...
N110 G1 X80 Y-50
/ N120 S2500
N130 X80 Y50
...
```

In the example above, the block N120 is conditional.

If CBB=0, the control will cancel compensation at the end of the block N110 and will recover it in the block N130.

If CBB=1, the control will not suspend the reading ahead, the calculation of compensation will be continuous.

In the state of G41, G42 it is recommended to avoid programming a conditional block.

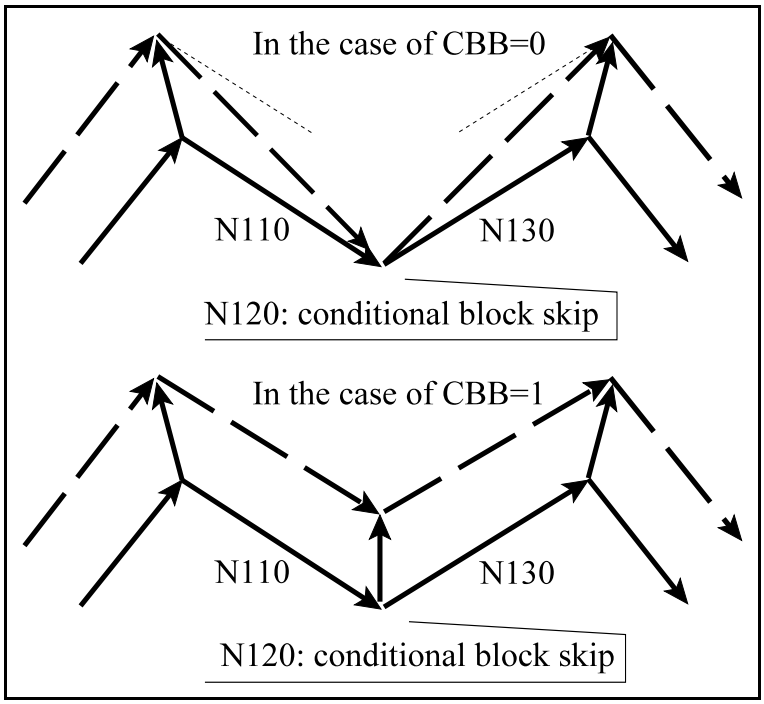


Fig. 15.5.2-11

15.5.3 Cancelling the Tool Radius Compensation. Leaving the Contour

The instruction G40 cancels calculation of tool radius compensation. The instruction G40 can be issued by linear interpolation only. If G40 is programmed in circular block, the control will send the error message '2043 G40 in circle interpolation'.

The basic cases of cancelling the tool radius compensation:

(G42)	(G42)
G1 X_ Y_	G2 X_ Y_ R_
G40 X_ Y_	G40 G1 X_ Y_

Leaving an inner corner: 180°<α<360°

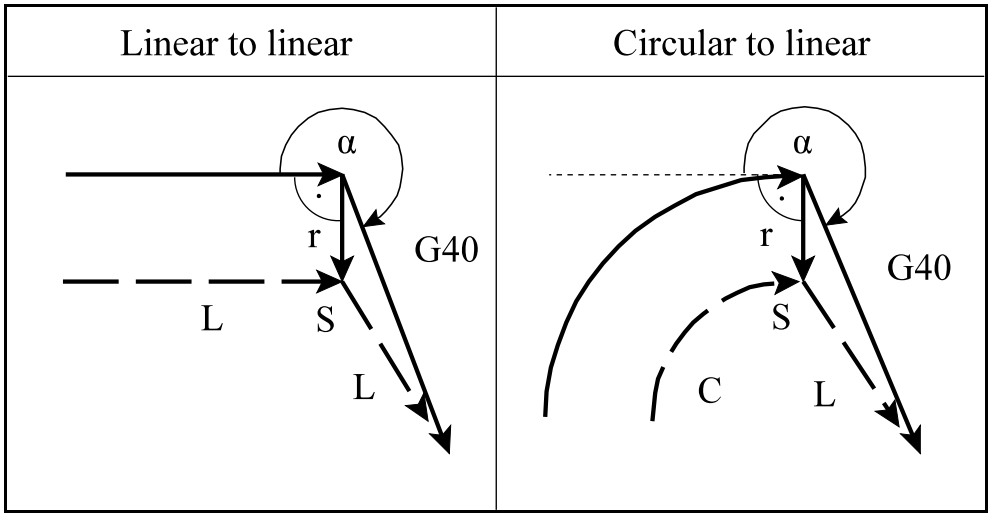


Fig. 15.5.3-1

Leaving an outer corner at an obtuse angle: 90°≤α≤180°

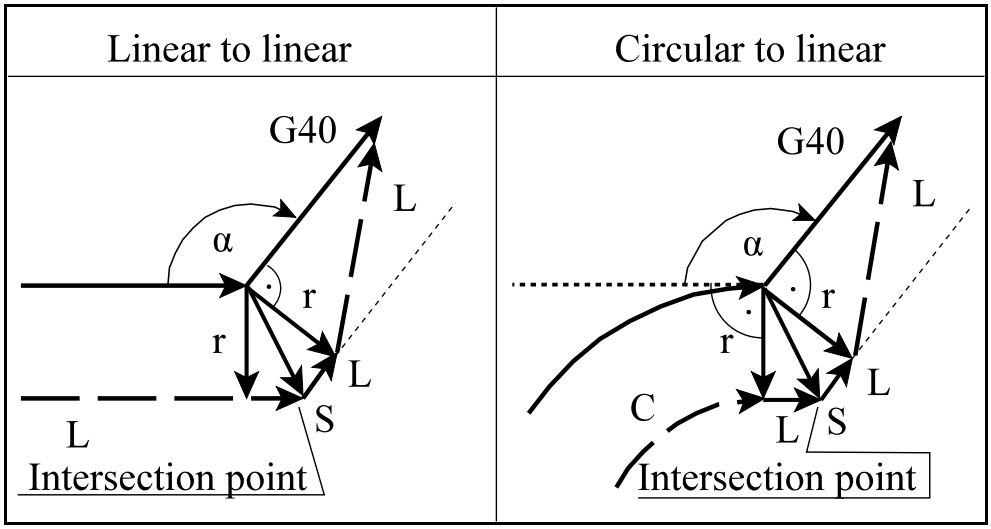


Fig. 15.5.3-2

Leaving an outer corner at an acute angle: $0° \leq \alpha < 90°$

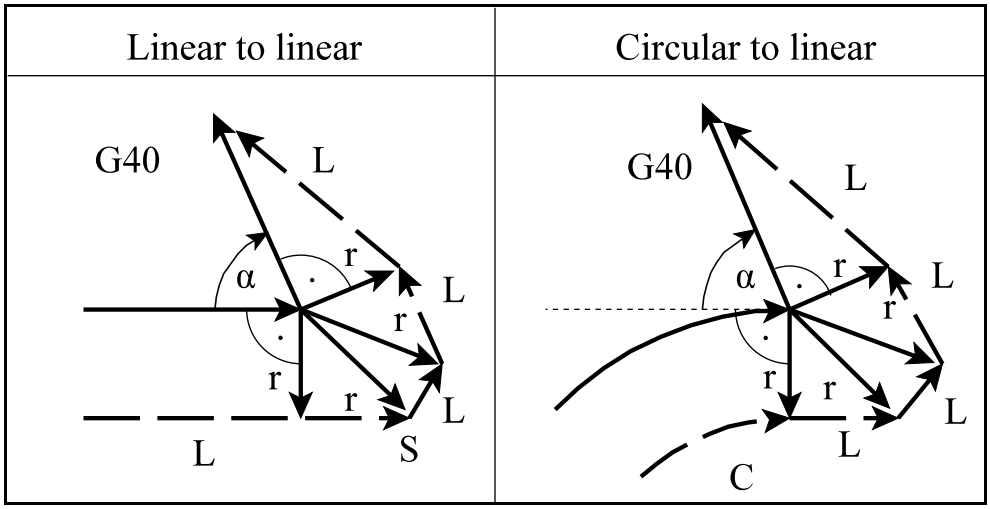


Fig. 15.5.3-3

An example:
A circle with the radius R60 is to be machined internally. Leaving the circle with the radius R60 is executed along a circle arc with the radius R50 (N110) to the endpoint of which a perpendicular with a length of tool radius (r) is let fall by the control.
The block N120 cancels the tool radius compensation by positioning to the point X0 Y0:

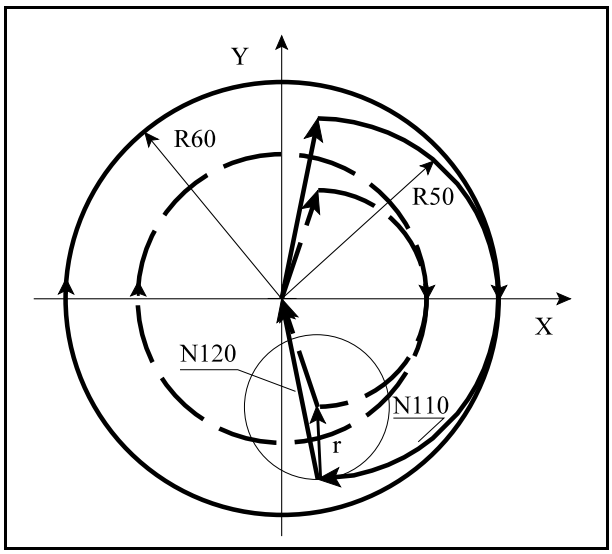


Fig. 15.5.3-4

```
G54 G17
...
N50 G0 X0 Y0
N60 G43 Z1 H1
N70 G1 Z-2 F1000
N80 G42 G0 X10 Y50 D1
N90 G2 X60 Y0 R50
N100 I-60
N110 X10 Y-50 R50
N120 G40 G0 X0 Y0
...
```

A square with side length of 60 mm is to be milled around. The tool radius compensation is cancelled by the block N110. The control defines the endpoint of the block N90 in such a way that it lets fall a perpendicular with a length of tool radius (r) to the endpoint of the block and moves from there to the point X-40 Y-40 in the block N100.

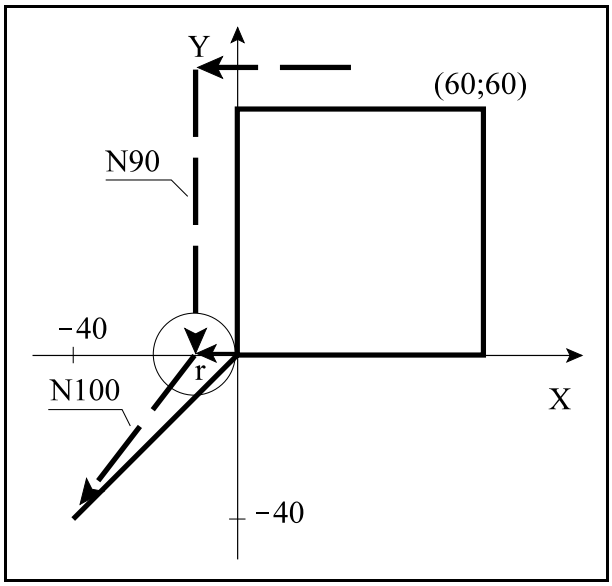


Fig. 15.5.3-5

```
N10 G54 G17 G49 M3 S500
N20 G0 X-40 Y-40
N30 G43 Z2 H1
N40 G1 Z-5 F500
N50 G42 G0 X0 Y0 D1
N60 G1 X60
N70 Y60
N80 X0
N90 Y0
N100 G40 G0 X-40 Y-40
N110 M30
```

Cancelling the tool radius compensation without programming of motion:

If ***tool nose radius compensation is cancelled*** (G40) in a block in which there are ***none of the addresses of the axes belonging to the selected plane*** (as in the example below, neither the address X nor the address Z is given in the block N110), motion in the block N110 will not be executed. First, in the block N100, the control moves the tool being offset by tool radius up to the endpoint in parallel (the line segment 1 in the figure), and then, moving the distance of tool radius (r) it executes the motion 2, and ***overcuts the workpiece!***
Correction can be done by merging the blocks N110 and N120.

```
N10 G54 G17 G49 M3 S500
N20 G0 X-40 Y-40
N30 G43 Z2 H1
N40 G1 Z-5 F500
N50 G42 G0 D1
N60 X0 Y0
N70 G1 X60
N80 Y60
N90 X0
N100 Y0
N110 G40
N120 G0 X-40 Y-40
N130 M30
```

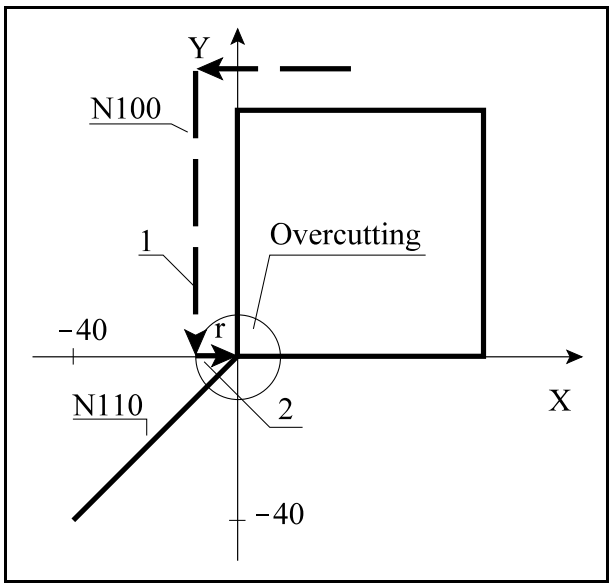


Fig. 15.5.3-6

If ***tool radius compensation is cancelled*** (G40) in a block in which ***there is no displacement along none of the axes belonging to the selected plane*** (as in the example below, in the block N110 the address Y is filled, but because of the incremental 0 belonging to it, there is no motion), in the block N100, the control will move the tool being offset by tool radius up to the endpoint in parallel (the line segment 1 in the figure), and then, moving the distance of tool radius (r) it will execute the motion 2, and ***overcuts the workpiece!***
Correction can be done by merging the blocks N110 and N120.

```
N10 G54 G17 G49 M3 S500
N20 G0 X-40 Y-40
N30 G43 Z2 H1
N40 G1 Z-5 F500
N50 G42 G0 G91 X0 D1
N60 G90 X0 Y0
N70 G1 X60
N80 Y60
N90 X0
N100 Y0
N110 G40 G91 Y0
N120 G0 G90 X-40 Y-40
N130 M30
```

Cancelling tool radius compensation with calculation of intersection point

If, in the block carrying out cancelling (G40), a value is assigned to I, J and K, but only to those that are in the selected plane (for example, to I and K in the case of G18), the control will move to the intersection point defined by the previous block and by I, J and K. The value of I, J and K is always incremental, and the vector defined by them points from the endpoint of the previous block.
It is a useful thing in the case of leaving an inner corner, for example.

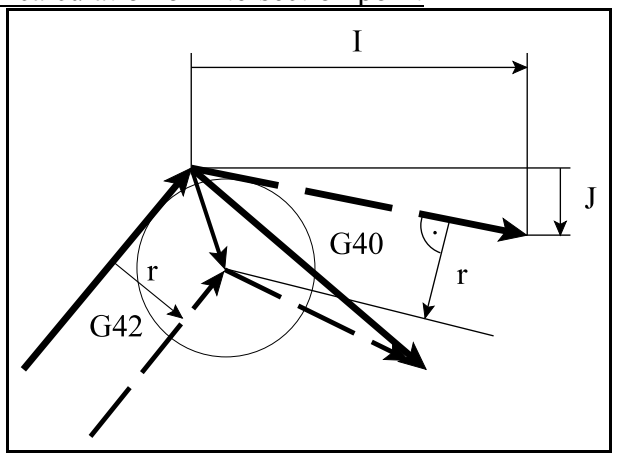


Fig. 15.5.3-7

An example:
A hexagon with side length of 100 mm is to be milled internally. The tool has to be left from the corner point with the coordinates of X100 Y0.The block N120 of the program below leaves from the contour using calculation of intersection point, with specification of I and J. The coordinates of I and J were calculated on the basis of the displacement executed in the block N60 of the program.

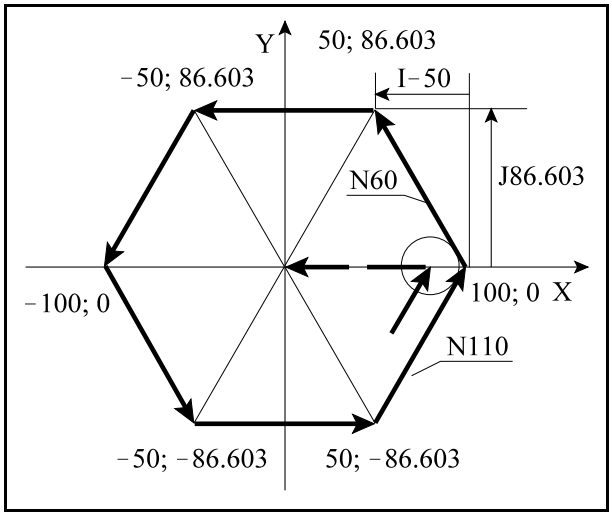


Fig. 15.5.3-8

```
N10 G54 G17 M3 S300
N20 G0 X0 Y0 Z100
N30 G43 Z5 H1
N40 G1 Z-5 F1000
N50 G41 G0 X100 Y0 I50 J86.603 D1
N60 G1 X50 Y86.603
N70 X-50
N80 X-100 Y0
N90 X-50 Y-86.603
N100 X50
N110 X100 Y0
N120 G40 G0 X0 I-50 J86.603
N130 M30
```

Cases of leaving using intersection point

In the case of specification I, J and K, the control always calculates intersection point regardless of whether inner or outer corner is to be machined.

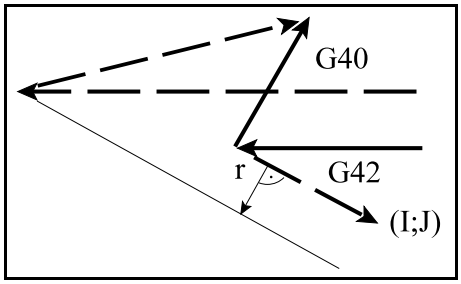


Fig. 15.5.3-9

If the control does not find intersection point, it will position to the endpoint of the previous block at right angles.

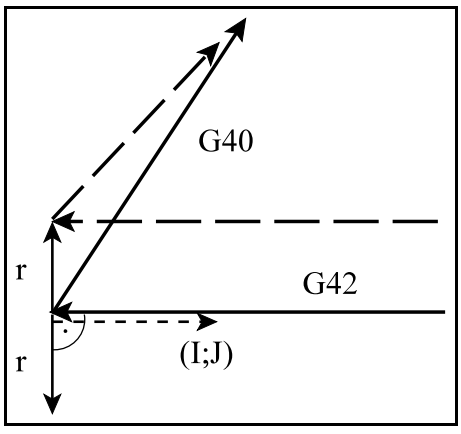


Fig. 15.5.3-10

15.5.4 Reversal in Calculation of Tool Radius Compensation

The following table shows the direction of calculation of tool nose radius compensation, i.e. the direction of tracking the contour:

	Tool radius compensation: **positive**	Tool radius compensation: **negative**
G41	from the left	from the right
G42	from the right	from the left

The direction of tracking the contour can also be reversed when the calculation of tool radius compensation is started up. It can be carried out by programming G41 or G42, or by calling tool radius compensation with opposite sign, at the address D. When the direction of tracking the contour is being changed, the control will not check whether the position is ‘outside’ or ‘inside’, but it will always calculate intersection point first. In the examples illustrated below, positive tool radius and change over from G42 to G41 are assumed:

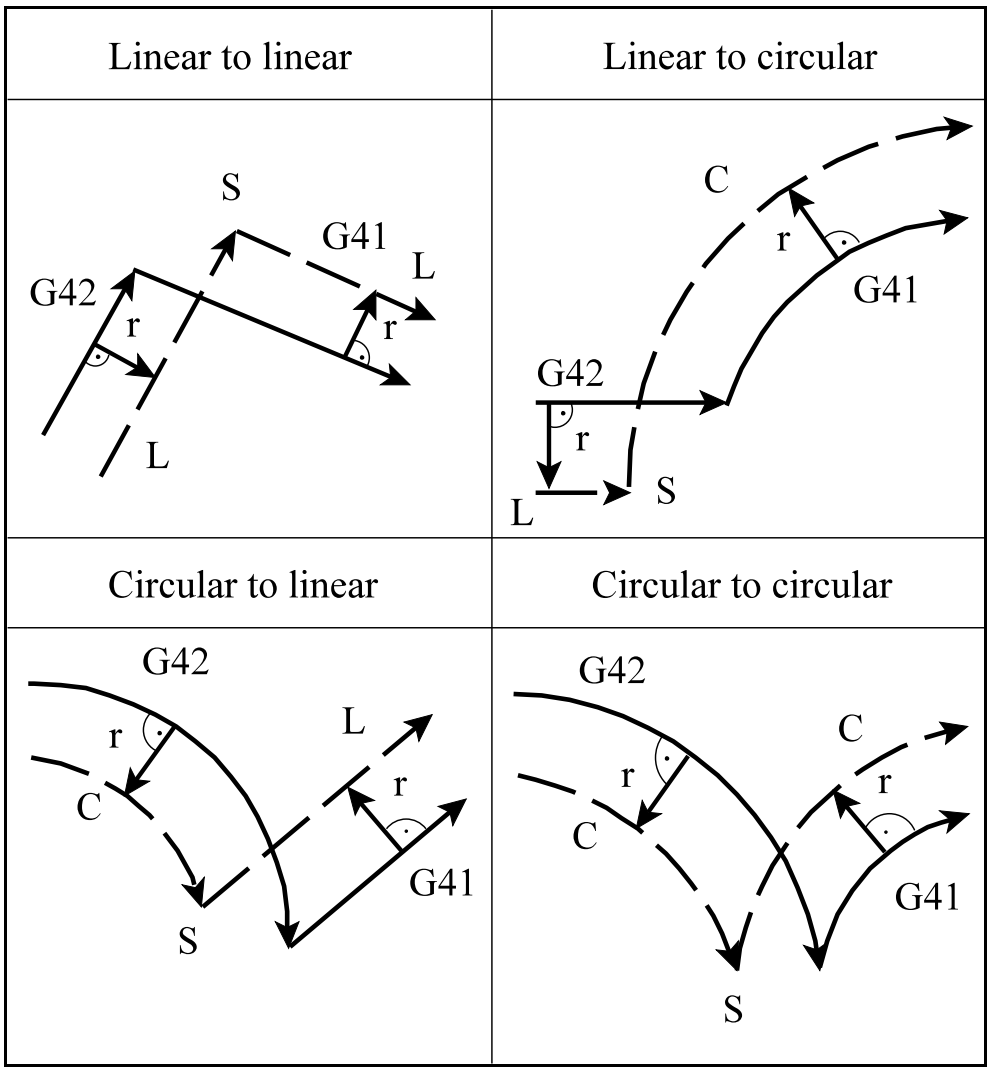


Fig. 15.5.4-1

An example:
Positioning to a circle with radius of R50 is executed in the blocks N30 and N40, along a straight line from the left, using G41. Then, the circle is tracked in the block N50, from the right, using G42. Leaving is also executed along a straight line from the left, using G41.

```
...
N10 G17 G0 X100 Y0
N20 M3 S500
N30 G41 X80 D1
N40 G1 X50
N50 G42 G3 I-50
N60 G41 G1 X80
N70 G40 X100
...
```

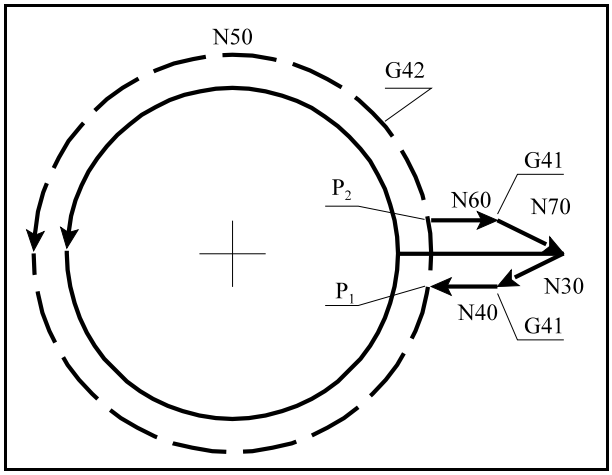


Fig. 15.5.4-2

Programming reversal along full circle

Programming full circle with reversal G41-G42, several cases can occur, when the path covered by the tool will be longer than the length of the full circle.
The example above illustrates this case.
The center point of the tool covers a full circle from the point P1 up to the point P1, and then a circle arc from the point P1 up to the point P2.

There is no intersection point in the case of linear - linear transition

If, in the case of linear - linear transition, no intersection point is resulted, the path of the tool will be as it is illustrated in the figure. According to the figure, reversal will occur from G42 to G41.
In this case, in the endpoint of the block preceding the reversal, the endpoint will be the vector with the length of tool radius which is put perpendicularly to the starting point of the succeeding straight line.

The upper figure shows the case when the direction of the path does not change, the lower figure illustrates the case of the path reversal of 180°.

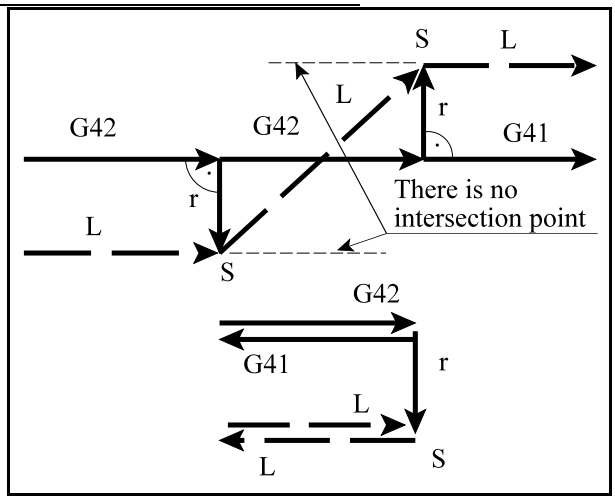


Fig. 15.5.4-3

There is no intersection point in the case of linear - circular transition

If, in the case of linear - circular transition, no intersection point is resulted, the path of the tool will be as it is illustrated in the figure. According to the figure, reversal will occur from G41 to G42.
In this case, in the endpoint of the straight line preceding the reversal, the endpoint will be the vector with the length of tool radius which is put perpendicularly to the starting point of the succeeding straight line.

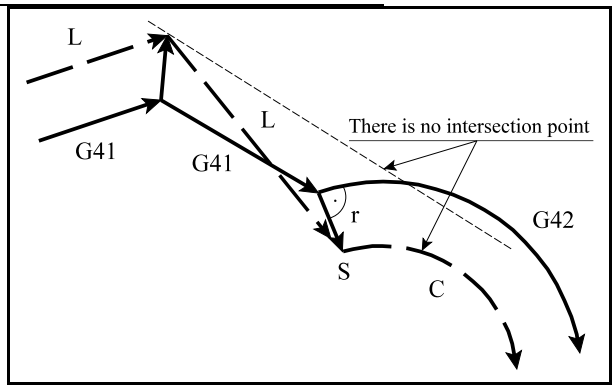


Fig. 15.5.4-4

There is no intersection point in the case of circular - linear or circular - circular transition

If, in the case of circular - linear or circular - circular transition, no intersection point is resulted, the control will interconnect the endpoint of the compensation vector resulted in the starting point of the first circular block and the endpoint of the compensation vector put perpendicularly to the starting point of the second block using an uncorrected programmed circle arc with radius R. In this case, the center point of the interconnecting circle arc will not coincide with the center point of the programmed circle arc. If reversal cannot be executed even with this change over of center point of circle, the control will send an error message.

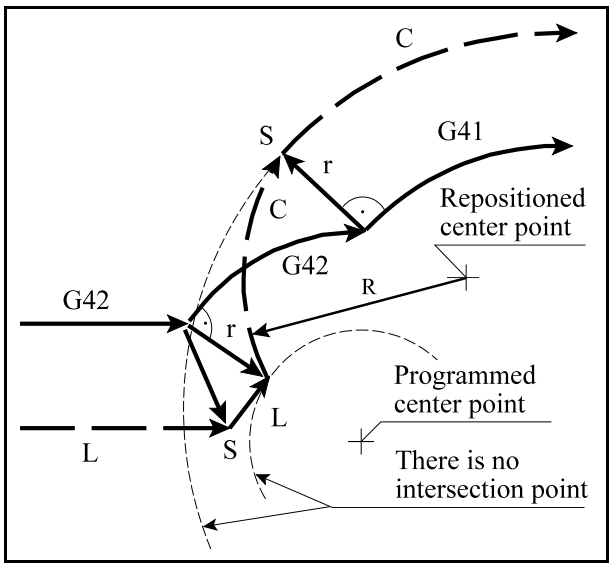


Fig. 15.5.4-5

15.5.5 Programming Vector Preservation (G38)

Due to the instruction

G38 v,

in started up status of calculation of planar tool nose radius compensation, the control ***preserves the last compensation vector between the preceding block and the block of G38, and validates it at the end of the block of G38***, irrespective of transition between the block of G38 and the succeeding block.
The code is a one-shot code, i.e. it is not a modal one. If it is necessary to preserve the vector in several consecutive blocks, the code G38 has to be programmed again.

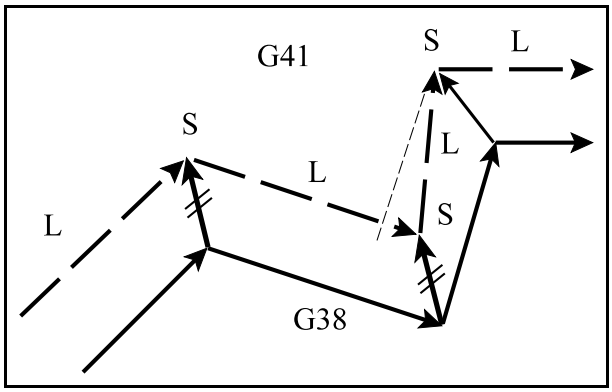


Fig. 15.5.5-1

An example:
Grooving is to be programmed without cancelling the contour tracking:

```
...G17 G42 G91...
N110 G1 X40
N120 G38 X50
N130 G38 Y70
N140 G38 Y-70
N150 X60
...
```

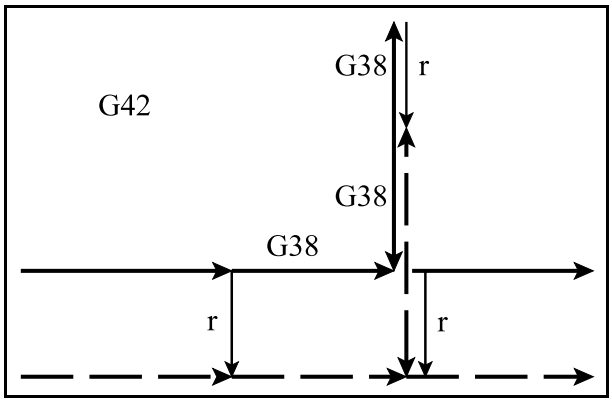


Fig. 15.5.5-2

15.5.6 Programming Corner Arc (G39)

When programming a block

G39 (I J K)

in the started up state of calculation of planar tool nose radius compensation, it can be reached, that in the case of going around an outer corner the control will not calculate intersection point automatically or will not insert straight line segments for going around; instead, the center point of the tool will move along a circle arc with radius equal to the tool radius.

The control ***inserts a circle with radius equal to the tool radius***; the direction of the circle is G02 in the state G41, and G03 in the state G42.

The starting point of the circle is defined by the vector perpendicular to the endpoint of the preceding block and length of which is equal to the tool radius, while the endpoint of the circle is defined by the vector perpendicular to the starting point of the succeeding block and length of which is equal to the tool radius. ***G39 must be programmed in separate block***:

```
...G17 G91 G41...
N110 G1 X100
N120 G39
N130 G3 X80 Y-80 I80
...
```

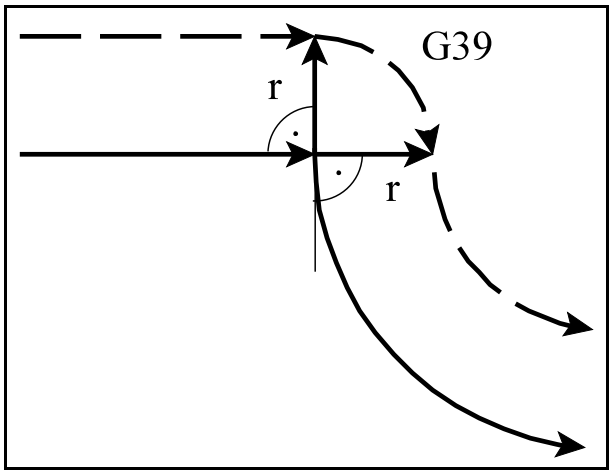


Fig. 15.5.6-1

If I, J or K is programmed in the block of G39 in accordance with the selected plane, the endpoint of the circle arc will be defined by the vector which is perpendicular in the endpoint of the preceding block to the vector defined by I, J or K and length of which is equal to tool nose radius:

```
...G17 G91 G41...
N110 G1 X100
N120 G39 I50 J-60
N130 G40 X110 Y30
...
```

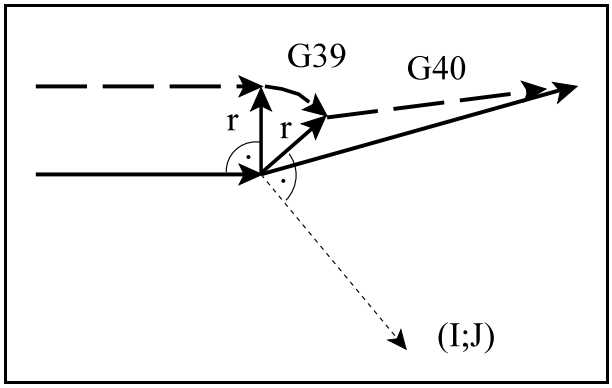


Fig. 15.5.6-2

Instructions on mirroring or rotation set previously are valid for the vector defined by I, J or K. Direction, certainly, is not affected by scaling instruction.

In the block of G39, no motion instruction of any kind can be programmed.

An example:
A triangular slot is to be milled with fillet at the corners.
In order that the contour will be closed, J is used for giving G39 in the block N110. Data of I and J are specified from displacement of the block N60.

```
...
N20 G0 G17 G40 G54
N30 X-30 Y150 Z5 M3 S300
N40 G42 X0 Y86.603 D1
N50 Z-2
N60 G1 X-50 Y0 F1000
N70 G39
N80 X50
N90 G39
N100 X0 Y86.603
N110 G39 I-50 J-86.603
N120 Z5
N130 X-40
N140 G40 Y120
...
```

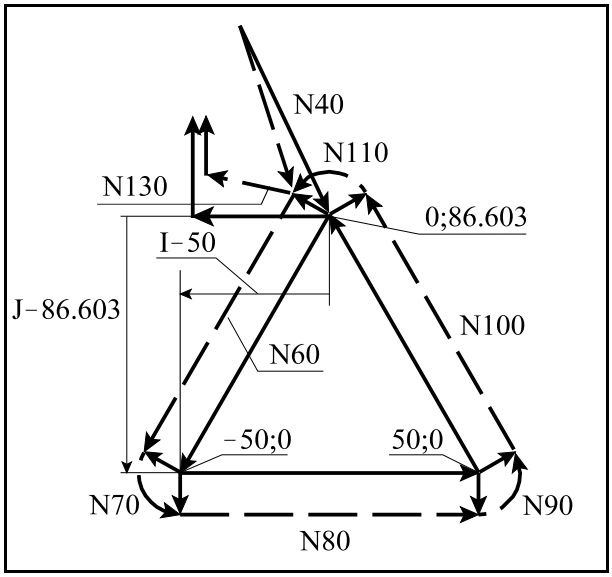


Fig. 15.5.6-3

15.5.7 Troubles in Tracking the Contour. Interference Check

During execution of contour tracking, it occurs in many cases that the tool path will be opposite of the programmed path. In this case, ***the tool can cut into the workpiece but that is not the intention of the programmer***. This phenomenon is called ***trouble in tracking the contour*** or ***interference***.
In the case illustrated in the figure, after calculation of the intersection points, during execution of the block N2, a tool path (dash line) opposite to the programmed path (continuous line) will be generated. The tool will cut into the workpiece.

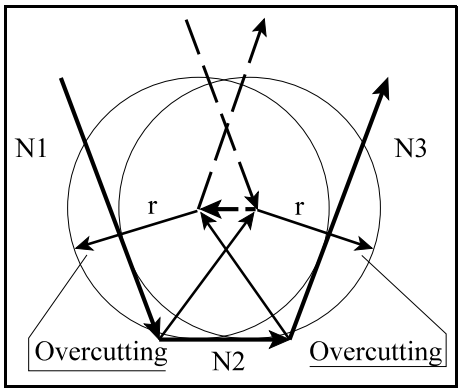


Fig. 15.5.7-1

In order to avoid such cases, the control executes interference check at the bit position #0 IEN=1 of the parameter N1403 Interference. In this case, ***the control checks whether the angle φ between the programmed displacement and the displacement corrected by the radius compensation satisfies the condition -90°≤φ≤+90°***.
In other words, the controls checks whether the compensated displacement vector has a component opposite to the programmed displacement vector.

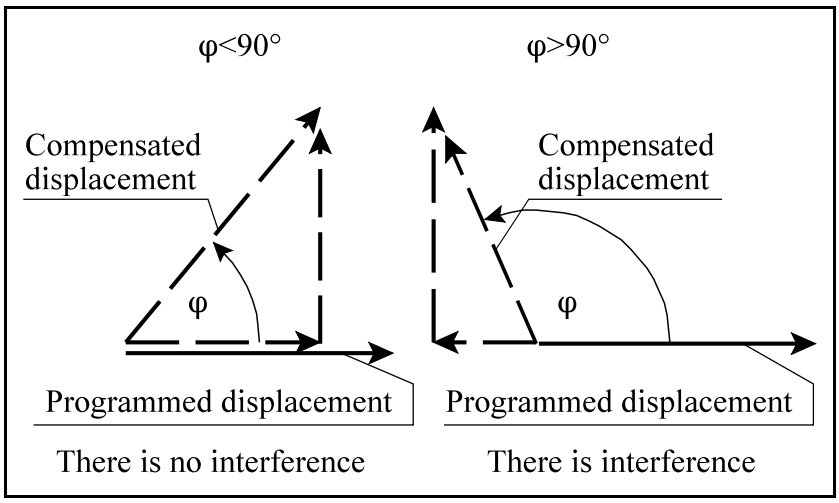


Fig. 15.5.7-2

If the control detects interference, ***it will either indicate the error or try to correct the error***, according to the bit position #1 AAL of the parameter N1403 Interference.
If ***AAL=1***, the control ***will always send the error message '2049 Interference alarm'*** at the end of the block preceding the block which causes the interference.
If ***AAL=0***, the control ***will try to correct the error automatically***, and it will send error message in the only case, if the automatic correction gives no result.

For monitoring interference and automatic correction, ***the control normally reads 3 blocks in ahead. If the value of the parameter N1404 BK No. Interf is greater than 0, the control will read in blocks quantity of which is a parameter value 3+*** and will execute interference check.

Automatic correction of interference

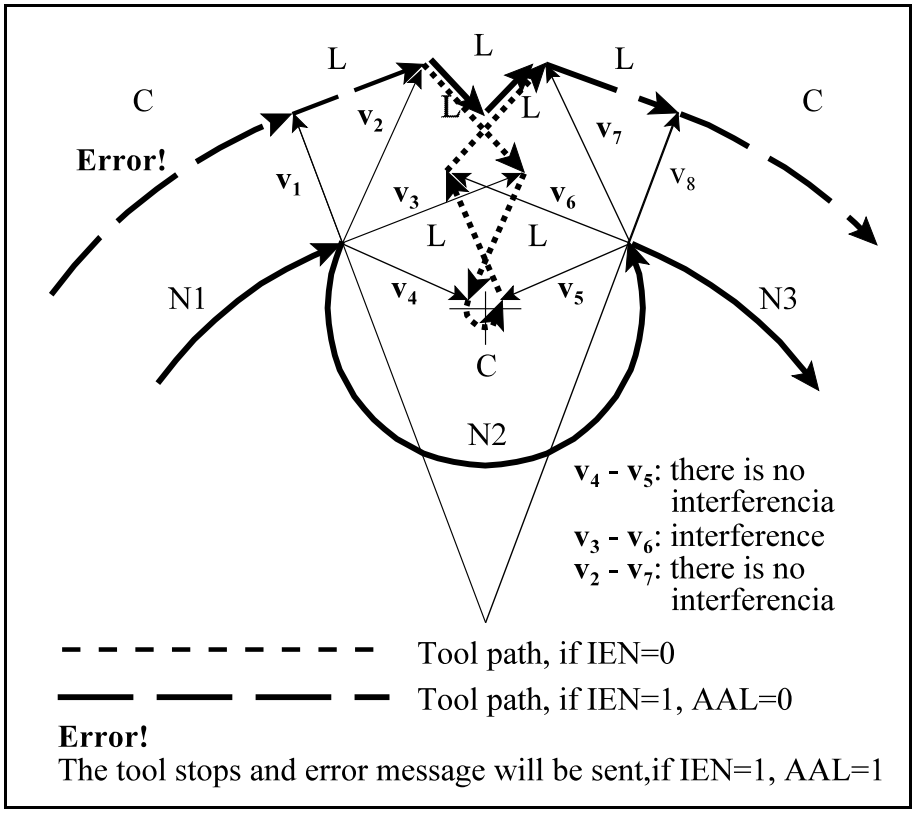


Fig. 15.5.7-3

If AAL=0, the control will not indicate error, but it will try to correct the contour automatically in order to avoid overcuttings. The course of the correction is as follows:
The tool nose radius compensation (G41) is started up in the blocks N1, N2 and N3.
The calculated compensation vectors between the blocks N1 and N2: $\mathbf{v}_1$, $\mathbf{v}_2$, $\mathbf{v}_3$ és $\mathbf{v}_4$.
The calculated compensation vectors between the blocks N2 and N3: $\mathbf{v}_5$, $\mathbf{v}_6$, $\mathbf{v}_7$ és $\mathbf{v}_8$.
If there is interference between the vectors $\mathbf{v}_4$ and $\mathbf{v}_5$ (a displacement opposite to the displacement of N2 is generated), the control will calculate intersection point between the straight line defined by the vectors $\mathbf{v}_3$ and $\mathbf{v}_4$ and the straight line defined by the vectors $\mathbf{v}_5$ and $\mathbf{v}_6$, and it will skip the circle arc N2.
If there is interference between the vectors $\mathbf{v}_3$ and $\mathbf{v}_6$ (displacement opposite to the displacement of N2 is generated), the control will calculate intersection point between the straight line defined by the vectors $\mathbf{v}_2$ and $\mathbf{v}_3$ and the straight line defined by the vectors $\mathbf{v}_6$ and $\mathbf{v}_7$, and it will skip motions between them (the figure above shows this case).
If there is interference between the vectors $\mathbf{v}_2$ and $\mathbf{v}_7$ (displacement opposite to the displacement of N2 is generated), the control will calculate intersection point between the straight line defined by the vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ and the straight line defined by the vectors $\mathbf{v}_7$ and $\mathbf{v}_8$, and it will skip motions between them.
If there is interference between the vectors $\mathbf{v}_1$ and $\mathbf{v}_8$, the control will try to calculate an intersection point between the blocks N1 and N3.
It is evident from the example above, that execution of the block N1 will start only after the interference check for the block N2 is carried out by the control. For this, however, the control had to read the block N3 into the buffer too, and to calculate the compensation vectors at the

transition N2 - N3.

If, in the block N2, in the case of AAL=0, the control cannot eliminate interference by calculation of an intersection point between the compassing segments, it will try to calculate an intersection point between the blocks N1 and N3. If there is an intersection point, the control will go on, otherwise it will send the error message '2049 Interference alarm'.

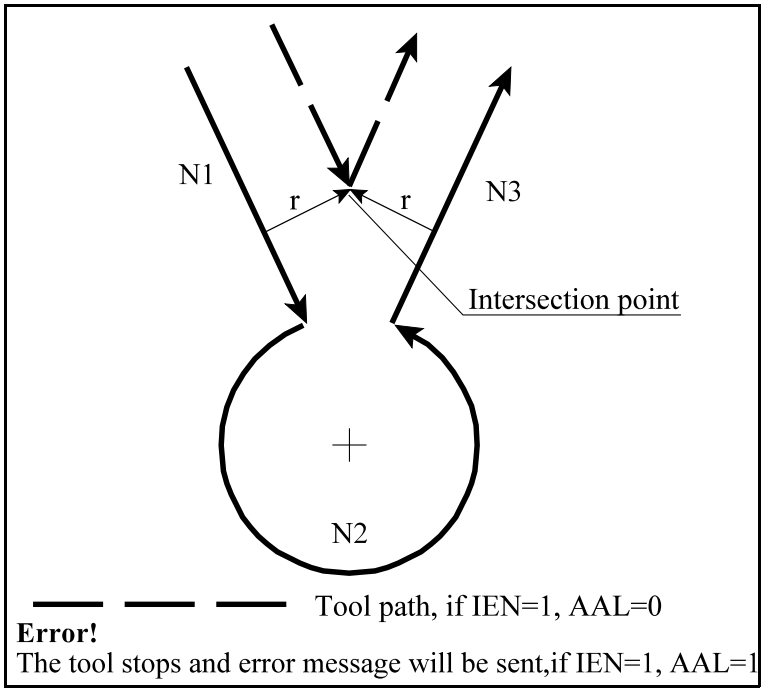


Fig. 15.5.7-4

Monitoring the interference several blocks in advance

In the examples above, 3 blocks are checked by the control. Namely, the transition between N1 and N2, and the transition between N2 and N3. It is adequate to the parameter value N1404 BK No. Interf=0.

The maximum value of the parameter N1404 BK No. Interf can be 8. In this case, the control executes the checks above

for the blocks (displacements) between N1 and N2, and, N2 and N3;

for the blocks (displacements) between N1 and N2, and, N3 and N4;

...

for the blocks (displacements) between N1 and N2, and, N [BK No. Interf+2] and N [BK No. Interf+3]. Then, depending on the position of the parameter AAL, in the block N1 the control will indicate error, or it will try carry out correction.

It can be used in the case when the tool enters into a hollow and it has to be checked whether there is enough room within for the tool with its diameter.

If the hollow illustrated in the figure is to be checked by the control, the value N1404 BK No. Interf=7 will have to be set.

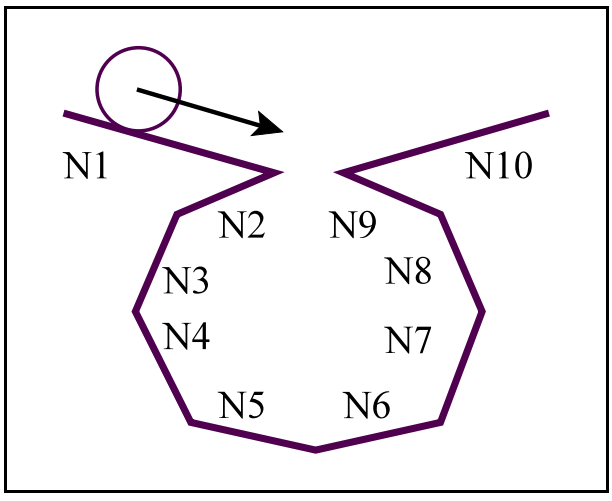


Fig. 15.5.7-5

Typical interferences

There are described below some typical cases when the control detects interference.

Producing a ***step with height smaller than the radius of the tool***.
In the case of IEN=0: the control cuts into the workpiece.
In the case of IEN=1:
if AAL=0, the control will avoid cutting into the workpiece by calculating an intersection point between the blocks N1 and N3;
if AAL=1, the control will send the error message '2049 Interference alarm', because it would cut into the workpiece.

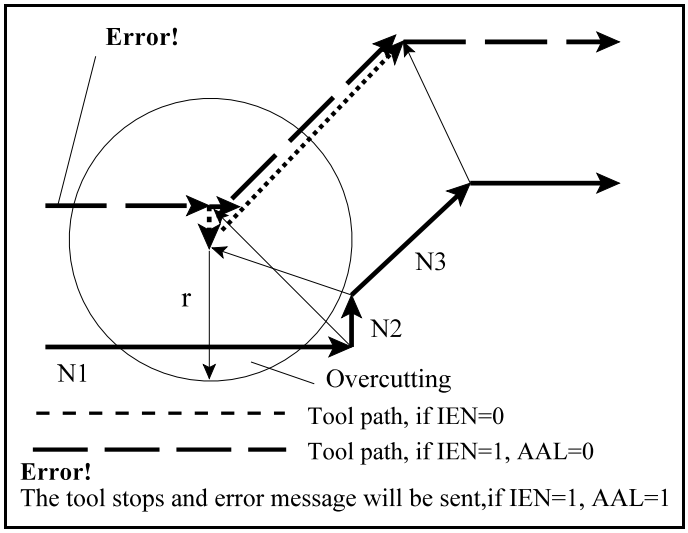


Fig. 15.5.7-6

Producing an ***inner corner with a radius smaller than the radius of the tool***.
In the case of AAL=1, the control will send the error message '2049 Interference alarm'.
In the case of AAL=0, the control, skipping the circle and calculating an intersection point between the two straight lines, will correct the error.

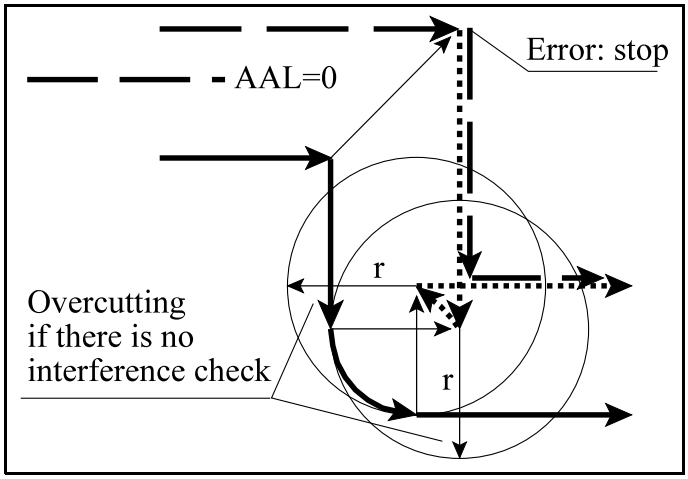


Fig. 15.5.7-7

Producing a ***step with height smaller than the radius of the tool, along a circle arc.***
If AAL=0, the control will calculate an intersection point between the straight line L1, and the straight line interconnecting the vectors $\mathbf{v_1}$ és a $\mathbf{v_3}$, in order to avoid the cutting in.
If AAL=1, the control will send the error message '2049 Interference alarm' and it will stop in the previous block.

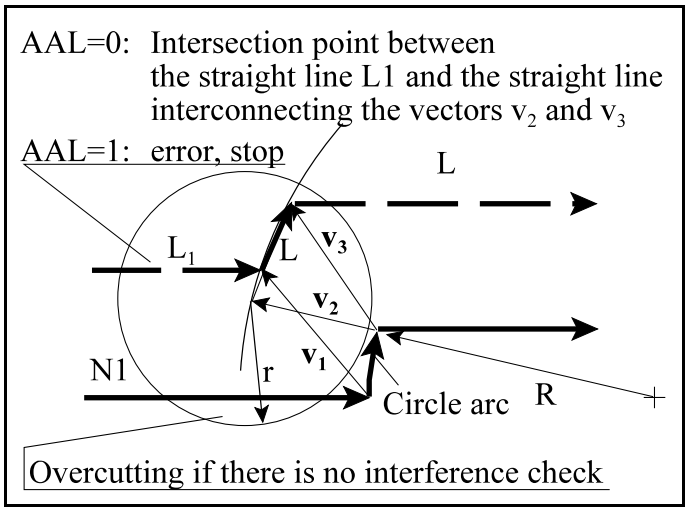


Fig. 15.5.7-8

In the case of the path shown in the figure, in the block N2 the displacement along the compensated path is opposite to the programmed path.
The control cannot correct if the value of the parameter N1404 BK No. Interf is 0, because there is no intersection point between the blocks N1 and N3, so it will indicate error in both cases of AAL=0 and AAL=1.
If the value of the parameter ***N1404 BK No. Interf is greater than 0,*** the control will continue reading the blocks, therefore it ***will correct*** interference by creating an intersection point between the blocks N1 and N4, skipping the blocks N2 and N3.

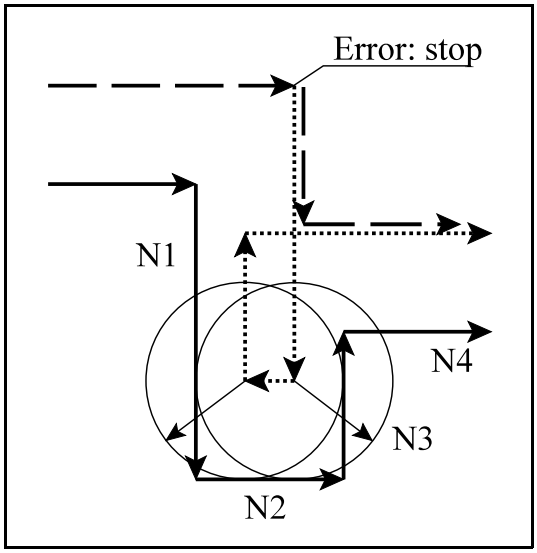


Fig. 15.5.7-9

In the case of the path shown in the figure, in the block N2 the displacement along the compensated path is opposite to the programmed path.
In the case of AAL=1, the control will indicate error at the starting point of the block N1.
In the case of AAL=0, the control will calculate an intersection point between the blocks N1 and N3, and moves up to this intersection point. If it proceeded from this intersection point, a motion opposite to the block N3 would be created along the corrected path, therefore the control will indicate error if the value of the parameter N1404 BK No. Interf is 0.

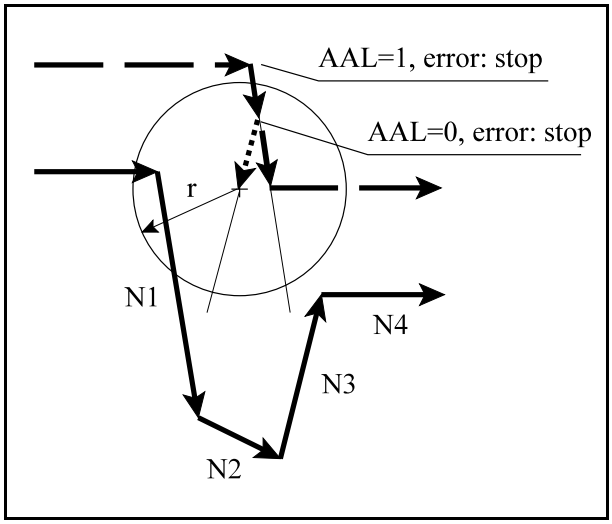


Fig. 15.5.7-10

If the value of the parameter ***N1404 BK No. Interf is greater than 0,*** the control will continue reading the blocks, therefore it ***will correct interference*** by creating an intersection point between the blocks N1 and N4, skipping the blocks N2 and N3.

Indicates interference error but does not cut into the workpiece

There are cases, when the interference check indicates error, but the control would not cut into the workpiece.

If producing a recess with depth smaller than the radius compensation is to be produced, maybe there will not be cutting into the workpiece in reality, as it is shown in the figure, but the control will send the error message '2049 Interference alarm' in the case of AAL=1, because in the block N3 the direction of the displacement along the corrected path is opposite to the programmed one.

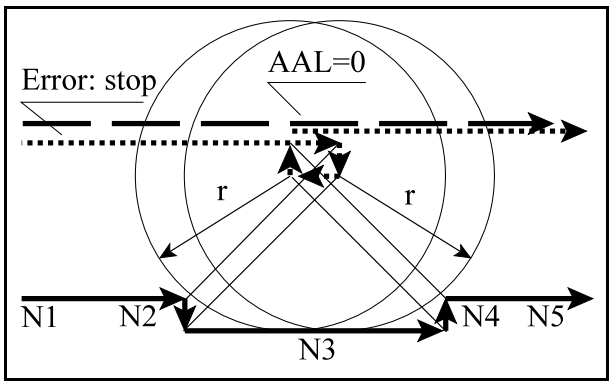


Fig. 15.5.7-11

In the case of AAL=0, the control continues machining, skipping the blocks N2, N3 and N4 and interconnecting the blocks N1 and N5, as it is illustrated in the figure.

In the example shown in the figure, the control also indicate interference, because in the block N3 the displacement along the corrected path is opposite to the programmed one.

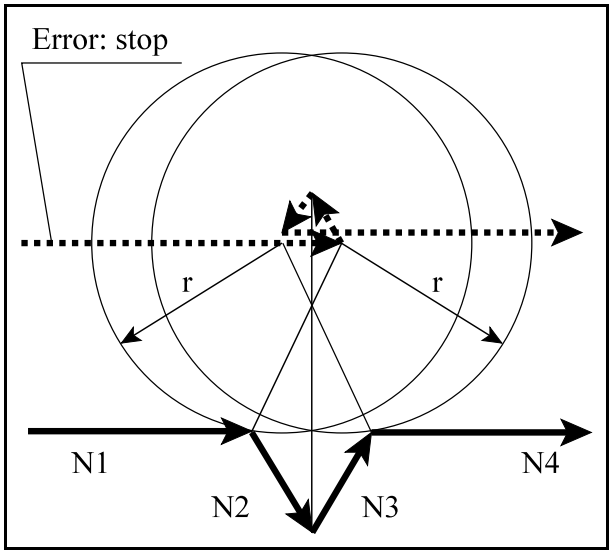


Fig. 15.5.7-12

There is cutting into the workpiece despite the monitoring interference

There are cases from the geometry of the path, when the control cuts into the workpiece, despite the monitoring interface. Some cases are shown below.

In the case illustrated in the figure, in the block N3, the displacement occurs in the direction opposite to the programmed one, and for this reason, the control will indicate interference at the starting point of block N2 and it will stop if the value of the parameter N1404 BK No. Interf is 0. Because of the geometry (G2) of the N4, the control will cut into the workpiece .

If the value of the parameter N1404 BK No. Interf is greater than 0, the control will continue reading the blocks, and for this reason, it will ***correct the interference*** by the intersection point between the blocks N1 and N4, skipping the blocks N2 and N3 in the case of AAL=0, otherwise it will indicate error at the beginning of the blockN1.

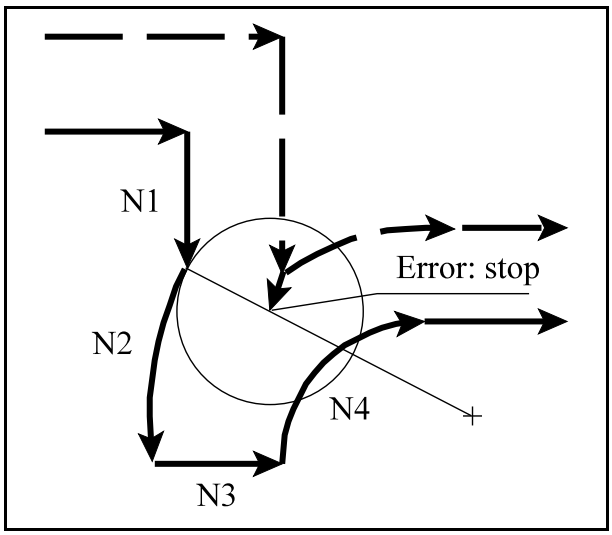


Fig. 15.5.7-13

In the case shown in the figure, no displacement occurs in the direction opposite to the programmed one in none of the blocks N1, N2 and N3, and for this reason, the control will not indicate interference if the value of the parameter N1404 BK No. Interf is 0, but even so, it will cut into the workpiece because of the geometry of the path.

If the value of the parameter N1404 BK No. Interf is greater than 0, the control will continue reading the blocks, and for this reason, it will ***correct the interference*** by the intersection point between the blocks N1 and N4, skipping the blocks N2 and N3 in the case of AAL=0, otherwise it will indicate error.

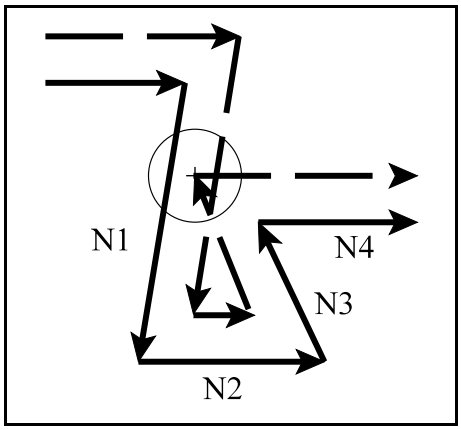


Fig. 15.5.7-14

In the case of the path illustrated in the figure, in the block N2, the displacement along the corrected path would be opposite to the programmed one.
In the case of AAL=1, the control indicates error at the starting point of the block N1.
If AAL=0, the control will calculate an intersection point between the blocks N1 and N3 and it will continue the motion up to this intersection point. If the control continued the motion from this intersection point, a motion opposite to the block N3 would occur, and for this reason the control indicates error.
However, at the end of the block, cutting into the workpiece will already occur if the value of the parameter N1404 BK No. Interf is 0.

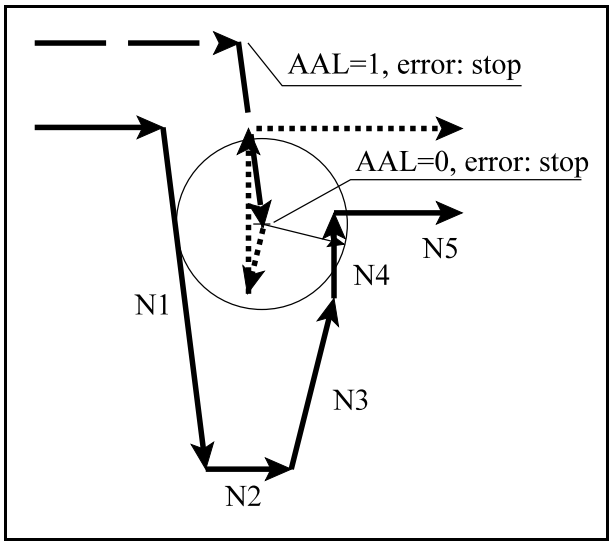


Fig. 15.5.7-15

If the value of the parameter N1404 BK No. Interf is 1, the control will indicate interference.
If the value of the parameter N1404 BK No. Interf is 2, the control will correct the interference by calculation of an intersection point between the blocks N1 and N5.

16 Special Transformations

16.1 Rotating a Shape Around a Given Point (G68, G69)

By the use of the instruction

G68 p q R

a programmed shape ***can be rotated in the plane selected by G17, G18 and G19***.

The ***coordinates of the center of rotation are*** specified at the addresses

p and ***q***.

The only values written at the coordinates p and q of the selected plane are interpreted by the control.

The entered ***coordinate data p and q*** will be interpreted by the control ***in the orthogonal coordinate system***, even if polar coordinate data specification is valid.

The coordinates p and q can be specified as ***either absolute or incremental*** data by the use of G90, G91 or the operator I. The incremental data has to be interpreted from the last programmed axis position (not from the rotated one).

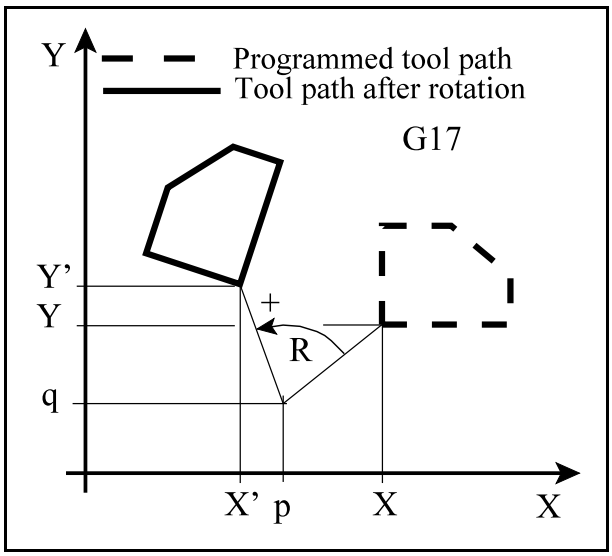


Fig. 16.1-1

If value is not assigned to one of the p and the q or both of them, the instantaneous axis position will be interpreted as the center of rotation.

The equation of rotation of a shape in the plane XY, in the case of G17 is the following:

$$X' = (X - p)\cos R - (Y - q)\sin R + p$$

$$Y' = (X - p)\sin R + (Y - q)\cos R + q$$

where: p, q: the center of rotation;

R: the angle of rotation;

X, Y : the coordinates of the programmed point;

X', Y': the coordinates of the rotated point.

It is the address

R,

at which the ***angle of rotation*** is specified. A ***positive*** or ***negative*** value entered at the address represents ***counter-clockwise*** or ***clockwise*** direction, respectively, to be interpreted in the selected plane.

The value assigned to the R can be ***either absolute or incremental***. If incremental angel of rotation is specified, the value of R will be added to the angles of rotation programmed previously.

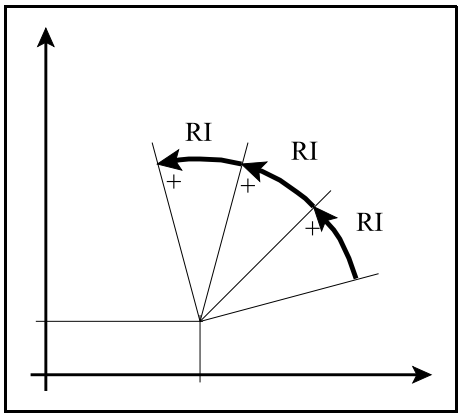


Fig. 16.1-2

The instruction

G69

cancels rotation. It deletes the coordinates of of the center of rotation and the angle of rotation, too. This instruction can also accompany other instructions, too.

An example:

```
N1 G17 G90 G0 X0 Y0
N2 G68 X90 Y60 R80
N3 G1 X60 Y20 F150
N4 G91 X140
N5 G3 Y80 R50
N6 G1 X60
N7 Y20
N8 G69 G0 X0 Y0
```

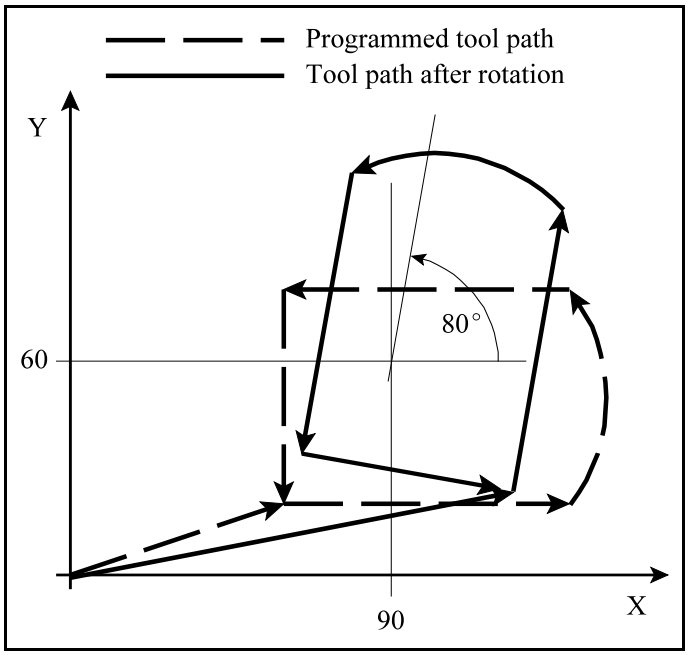


Fig. 16.1-3

16.2 Scaling a Shape in Relation to a Given Point (G50, G51)

Scaling can be activated by the code G51. Two ways are available for scaling specification: one of them is, when a common scale factor is valid for each axis; the other is, when different scale factors are applied to the axes X, Y and Z.
Both ways of specification can be cancelled by the instruction G50.

Scaling by the use of a scale factor valid for each axis (P)

Using the instruction

G51 v P,

a programmed shape can be ***reduced or magnified***.
At the coordinates

v ,

the ***position of the center point of scaling*** can be specified. The axes that can be used are the axes X, Y and Z, and the parallel axes.
The entered ***coordinate data v*** will be interpreted by the control ***in the orthogonal coordinate system***, even if polar coordinate data specification is valid.
The coordinates v of the center point of scaling can be specified as ***either absolute or incremental*** data by the use of G90, G91 or the operator I.

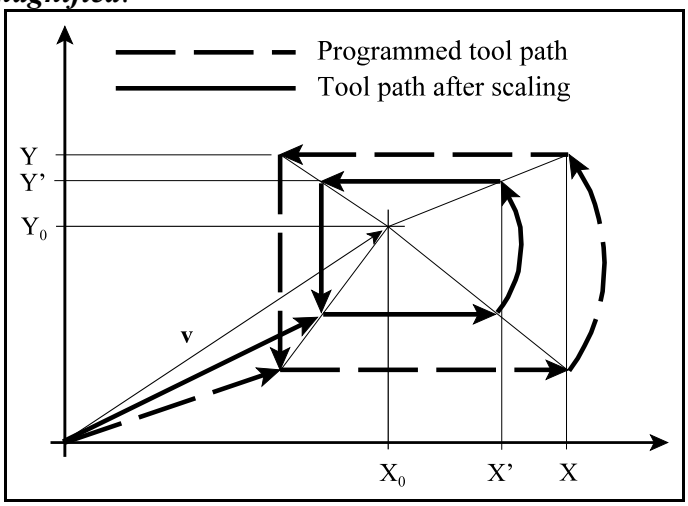


Fig. 16.2-1

If value is not assigned to one of the axis addresses or each of them, the instantaneous axis position will be interpreted as the center of scaling.
It is the address

P,

at which the ***scale factor*** is specified.
There will be reduction if P<1, and if P>1, magnification will occur.
The equation of the scaling of a shape in the space XYZ is the following:

$$X' = (X - X_0)P + X_0$$
$$Y' = (Y - Y_0)P + Y_0$$
$$Z' = (Z - Z_0)P + Z_0$$

where: X_0, Y_0, Z_0: the coordinates v of the center of scaling;
P: the magnification rate of scaling;
X, Y, Z: the coordinates of the programmed point;
X', Y', Z': the coordinates after scaling.

The instruction

G50

cancels the scaling.

☞ ***Note:***

The magnification rate of scaling (P) does not affect the value of radius compensation applied.

An example:

```
N1 G17 G90 G0 X0 Y0
N2 G51 X90 Y60 P0.6
N3 G1 X60 Y20 F150
N4 X140
N5 G3 Y80 R50
N6 G1 X60
N7 Y20
N8 G50 G0 X0 Y0
```

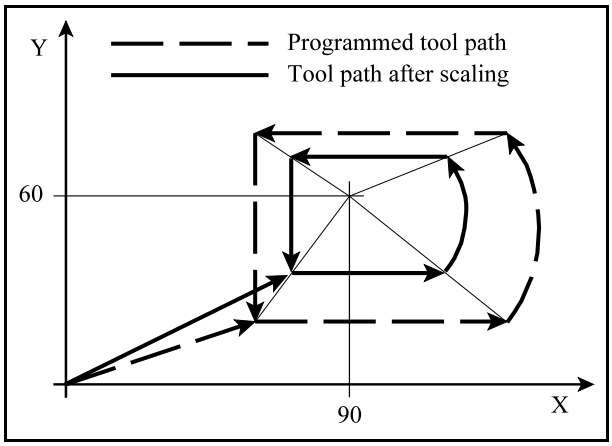


Fig. 16.2-2

Scaling by the use of a scaling rate different for each of the axes X, Y and Z (I J K)

Using the instruction

G51 X Y Z I J K

a programmed shape can be ***reduced or magnified***.

At the coordinates

X Y Z

the ***position of the center point of scaling*** can be specified. The axes that can be used are the main axes X, Y and Z.

The entered ***coordinate data v*** will be interpreted by the control ***in the orthogonal coordinate system***, even if polar coordinate data specification is valid.

The coordinates X, Y and Z of the center point of scaling can be specified as ***either absolute or incremental*** data by the use of G90, G91 or the operator I.

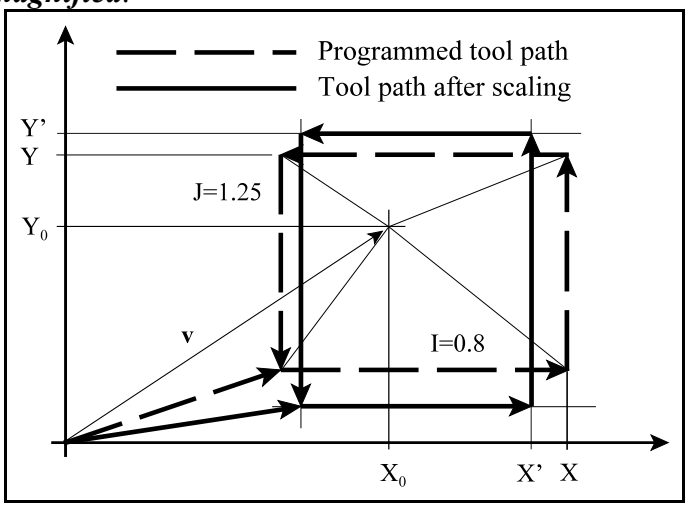


Fig. 16.2-3

If value is not assigned to one of the axis addresses or each of them, the instantaneous axis position will be interpreted as the center of scaling.

At the address

I J K

the ***magnification rate for the axes I-X, J-Y and K-Z,*** respectively can be specified.

There will be reduction if I J K<1, and if I J K>1, magnification will occur.

The equation of the scaling of a shape in the space XYZ is the following:

$$X' = (X - X_0)I + X_0$$
$$Y' = (Y - Y_0)J + Y_0$$
$$Z' = (Z - Z_0)K + Z_0$$

where: X_0, Y_0, Z_0: the coordinates v of the center of scaling;
I J K: the magnification rate of scaling;
X, Y, Z: the coordinates of the programmed point;
X', Y', Z': the coordinates after scaling.

The instruction

G50

cancels the scaling.

☞ ***Notes:***

– The magnification rates of scaling (I, J and K) do not affect the value of radius compensation applied.

– The values of I, J and K can also be negative. In this case, not only scaling, but mirroring too will be executed by the control, along the given axis. If the values of I, J and K are -1, only mirroring will be executed, along the given axis.

– If the values of I, J and K are different, the circular interpolation (G1, G3) and the arc radius R will be modified as follows::

```
G17
G51 X0 Y0 Z0 I0.5 J0.75 K1.25
G1 X100 ,R10 (R=,R*(I+J)/2)
Y100
X10
G3 X0 Y90 R10 (R=R*I)
G1 Y0
G50
```

The control will multiply the arc radius by the average of I and J, and in the block G3 the circle radius by I.

16.3 Mirroring a Shape Through One or More Straight Line (G50.1, G51.1)

By the use of the instruction

G51.1 v

a programmed shape can be ***mirrored by the control through the coordinates selected*** in v.
In the coordinates

v

there can be specified ***that axis***, ***the straight line or lines parallel with which will be used for mirroring***.
The v can be the axis address X, Y and Z, and other axis addresses.
The position of the straight line has to be determined as follows: If the straight line the mirroring is to be done is parallel with the axis X, the position will have to be specified at the axis Y; if the straight line the mirroring is to be done is parallel with the axis Y, the position will have to be specified at the axis X,; and so on.
The entered ***coordinate data v*** will be interpreted by the control ***in the orthogonal coordinate system***, even if polar coordinate data specification is valid.

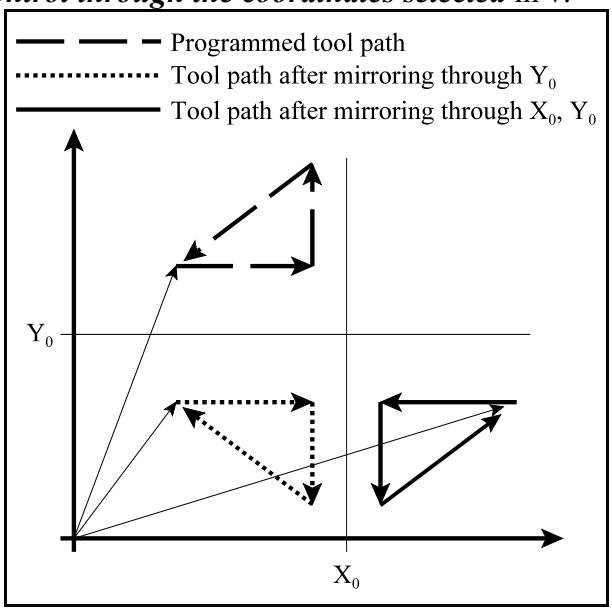


Fig. 16.3-1

The coordinates v of the axes of mirroring can be specified as ***either absolute or incremental*** data by the use of G90, G91 or the operator I.
If no value is assigned to any of the axis addresses, the control will not execute mirroring in that one.

The equation of the scaling of a shape in the space XYZ is the following:

$$X' = -(X - X_0) + X_0$$

$$Y' = -(Y - Y_0) + Y_0$$

$$Z' = -(Z - Z_0) + Z_0$$

where: X_0, Y_0, Z_0: the coordinates v of the center point of mirroring;
X, Y, Z: the coordinates of the programmed point;
X', Y', Z': the coordinates after mirroring.

Mirroring through the axes of odd number will change the direction of motion around the shape. As a result of this, the circle directions (G2-G3), the direction of tool radius compensation (G41-G42) and the angle of rotation (G68R) also will change, and this change will automatically be taken into account by the control.

The instruction

G50.1 v

cancels mirroring through coordinate axis (axes) specified in v. In the coordinates, any data can be written, the effect will be only recording the fact of cancellation.

When activating or cancelling the mirroring, neither rotation (G68) nor scaling (G51) has to be

in effect. Otherwise, the control sends the error message ‘2001 Unable to turn on or off mirroring under G51 or G68'.

An example:

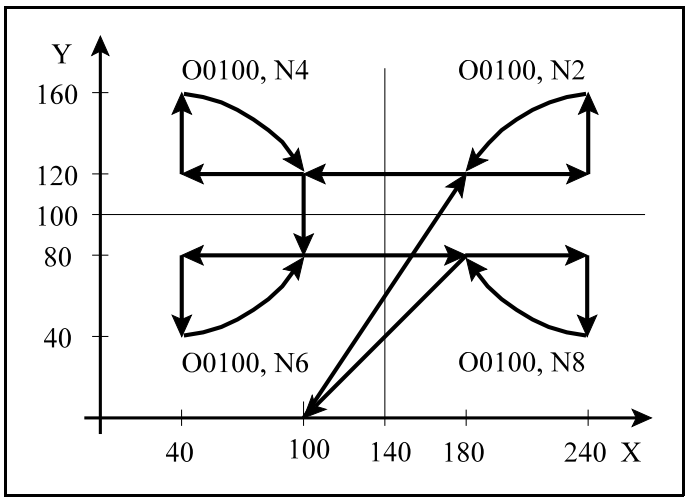


Fig. 16.3-2

Subprogram:

```
O0101
N1 G90 G0 X180 Y120 F120
N2 G1 X240
N3 Y160
N4 G3 X180 Y120 R80
N5 M99
```

Main program:

```
O0100
N1 G0 X100 Y0
N2 M98 P101     (calling the subprogram)
N3 G51.1 X140   (mirroring through the axis with the coordinate
X140)
N4 M98 P101     (calling the subprogram)
N5 G51.1 Y100   (mirroring through the axis with the coordinate
Y100 too)
N6 M98 P101     (calling the subprogram)
N7 G50.1 X0     (cancellation of mirroring through the axis with
the coordinate X140)
N8 M98 P101     (calling the subprogram)
N9 G50.1 Y0     (cancellation of mirroring through the axis with
the coordinate Y100 too)
N10 G0 X100 Y0
```

16.4 Programming Rules for Specific Transformations

The sequence of the rotation G68 and the scaling G51 is optional.
However, attention should be payed if rotation is done first and scaling follows, because ***the instruction of rotation will be valid for the coordinates of the center point of scaling***. But, if scaling is done first and rotation follows, ***the instruction of scaling will be valid for the coordinates of the center point of rotation.***
The starting and cancelling instructions of both operations have to be nested into one another; overlapping one another is not permitted:

Rotation–scaling

```
N1 G90 G17 G0 X0 Y0
N2 G68 X80 Y20 R75
N3 G51 X130 Y50 P0.5
N4 X180 Y20
N5 G1 Y80 F200
N6 X80
N7 Y20
N8 X180
N9 G50
N10 G69 G0 X0 Y0
```

Scaling–rotation

```
N1 G90 G17 G0 X0 Y0
N2 G51 X130 Y50 P0.5
N3 G68 X80 Y20 R75
N4 X180 Y20
N5 G1 Y80 F200
N6 X80
N7 Y20
N8 X180
N9 G69
N10 G50 G0 X0 Y0
```

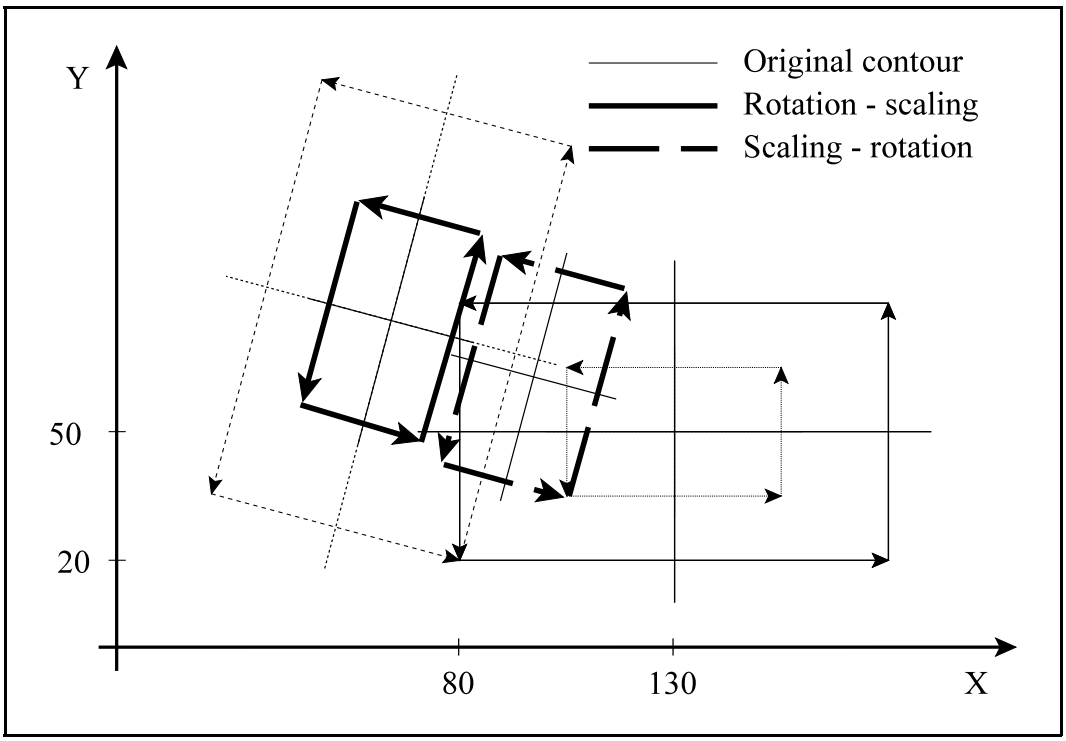


Fig. 16.4-1

It can be seen from the figure, that application sequence of several transformations cannot be disregarded.

Mirroring is quite a different matter. Starting up the mirroring is allowed in states G50 and G69 only, i.e. when there is neither scaling nor rotation states.

However, either scaling or rotation can be started up in the started up state of the mirroring.
It is valid for mirroring too, that overlapping of mirroring with either scaling or rotation is not allowed, so first the rotation and the scaling have to be cancelled in proper sequence, and the mirroring only after that.

```
G51.1 ... (starting up of the mirroring)
G51 ...   (starting up of the scaling)
G68 ...   (starting up of the rotation)
...
G69 ...   (cancelling of the rotation)
G50 ...   (cancelling of the scaling)
G50.1 ... (cancelling of the mirroring)
```

17 Automatic Geometric Calculations

17.1 Programming Chamfer and Corner Rounding

The control can insert ***chamfer*** or ***corner rounding*** automatically ***in the selected plane*** between two blocks containing linear interpolation (G01) or circular interpolation (G02, G03).

An isosceles chamfer, the side length of which is specified at the address

,C

(comma and C) is inserted by the control between the endpoint of the block containing the address ,C and the starting point of the succeeding block.

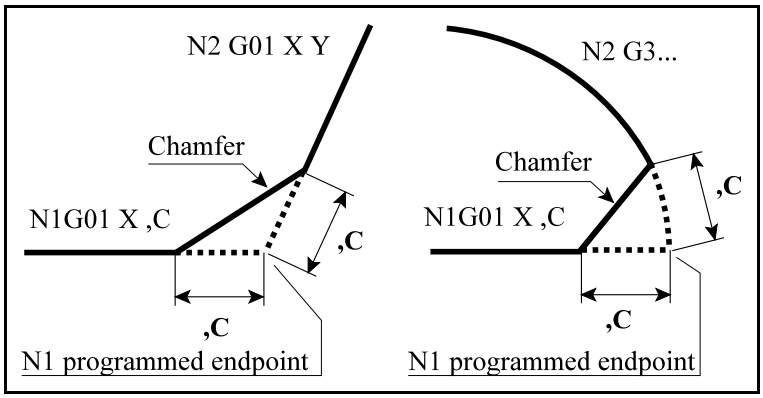


Fig. 17.1-1

For example:

```
N1 G17 G1 G91 X30 ,C10
N2 X10 Y40
```

The value specified at the address ,C shows the distance, with which the chamfer starts and ends from the supposed intersection point of the two sequence blocks. Chamfer can also be inserted between circles or circle and straight line. In this case, the value of ,C is the length of the chord drawn from the intersection point.

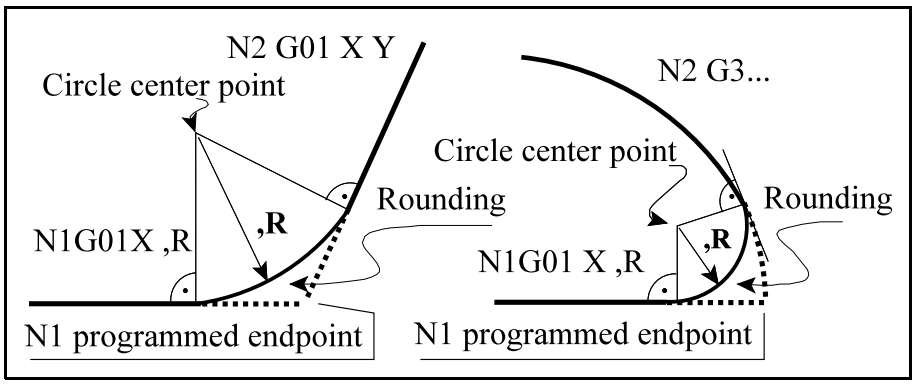


Fig. 17.1-2

A corner rounding, the radius of which is specified at the address

,R

(comma and R) is inserted by the control between the endpoint of the block containing the address ,R and the starting point of the succeeding block.

For example:

```
N1 G17 G91 G01 X30 ,R8
N2 G03 X-30 Y30 R30
```

The control inserts the circle arc with the radius of ,R between the two blocks in such a way, that

the circle osculates tangentially to the both path elements.
A instruction containing chamfer or corner rounding can also be written at the end of several succeeding blocks, as it is illustrated by the example below:

```
...
G17 G1 Y40 ,C10
X60 ,R22
G3 X20 Y80 R40 ,C10
G1 Y110
...
```

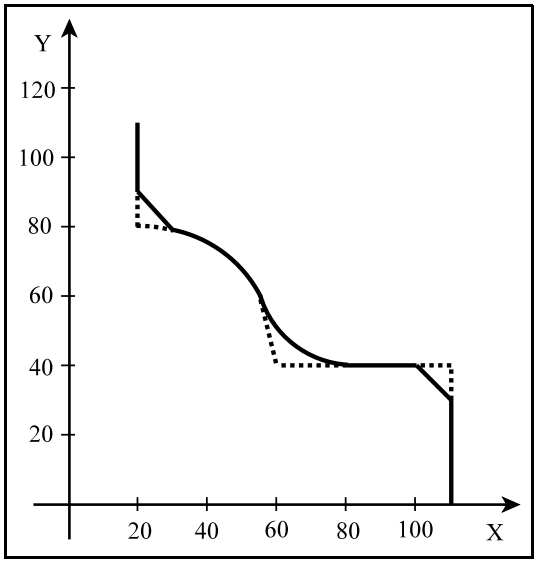


Fig. 17.1-3

☞ ***Notes**:*

– Chamfer or corner rounding can only be programmed between elements being in the selected plane (G17, G18, G19), otherwise the control sends error message '2085 Illegal ,C or ,R'.

– If the side length of the chamfer or the radius of the corner rounding is so great that it cannot be suited to the programmed blocks, the control will send the error message '2083 ,C or ,R is too high'.

– If both ,C and ,R is programmed within one block, the control will send the error message '2082 Whether chamfer or rounding'.

– In block by block mode, the control stops and gets into the stop state after execution of chamfering or corner rounding.

17.2 Specification of a Straight Line Using its Angle of Inclination

A straight line in the plane defined by the instructions G17, G18, G19 ***can be specified by one of the coordinates of the selected plane and by its angle of inclination given at the address ,A***.

$$G17\left\{\frac{G0}{G1}\right\}\left\{\frac{X_P,A}{Y_P,A}\right\}qF$$

$$G18\left\{\frac{G0}{G1}\right\}\left\{\frac{Z_P,A}{X_P,A}\right\}qF$$

$$G19\left\{\frac{G0}{G1}\right\}\left\{\frac{Y_P,A}{Z_P,A}\right\}qF$$

In the formulas above, X_p, Y_p and Z_p are the endpoint coordinates of the straight line along the

axes X, Y and Z or along the axes parallel with them; q indicates one or more arbitrary axes being out of the selected plane.
Specification at the address ,A can also be used in addition to the codes G0 and G1.
The angle ,A is measured from the first axis of the selected plane, and the positive direction is opposite to the clockwise direction.
The value of ,A can be either positive or negative, and it can also be greater than 360°or smaller than −360°.

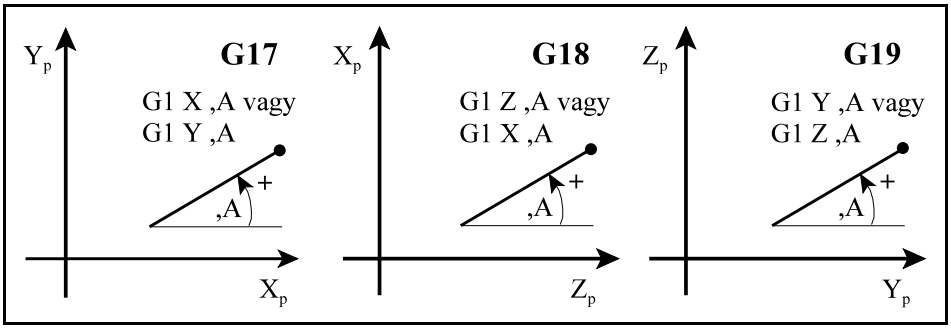


Fig. 17.2-1

For example:

```
G17 G90 G0 X57.735 Y0 ...
G1 G91...
X100 ,A30
```

(this specification is equivalent to the specification X100 Y57.735 where 57.735=100·tg30°)

```
Y100 ,A120
```

(this specification is equivalent to the specification X-57.735 Y100 where −57.735=100/tg120°)

```
X-100 ,A210
```

(this specification is equivalent to the specification X-100 Y-57.735 where −57.735=−100·tg30°)

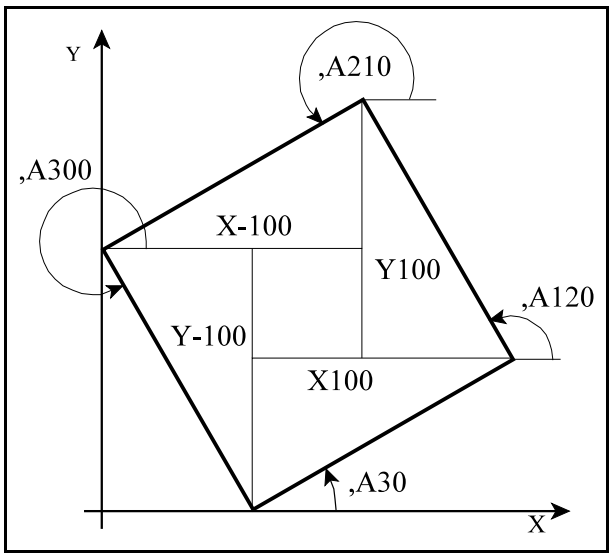


Fig. 17.2-2

```
Y-100 ,A300
```

(this specification is equivalent to the specification X57.735 Y-100 where 57.735=−100/tg120°)

☞ *Note:*

– Straight line with its angle of inclination and chamfer or corner rounding can also be specified in one block. For example:

```
X100 ,A30 ,C5
Y100 ,A120 ,R10
X-100 ,A210
```

– Specification of the angle of inclination at the address ,A can be used in drilling cycles, too. In this case, it is taken into account by the control during execution of positioning in the selected plane in the way described above. For example, the block

```
G81 G91 X100 ,A30 R-2 Z-25
```

is equivalent to the following block:

```
G81 G91 X100 Y57.735 R-2 Z-25
```

17.3 Calculations of Intersection Point in the Plane

The calculations described here are executed by the control in the only case, ***when the calculation of tool radius compensation (G41 or G42 offset mode) is on.*** If tool radius compensation is not to be taken into account in the program, even this case it has to be switched on, and compensation D00 has to be called:

with tool radius compensation:

```
G41(or G42) ...Dnn
...
Calculations of intersection
points
...
G40
```

without tool radius compensation:

```
G41(or G42) ...D00
...
Calculations of intersection
points
...
G40
```

17.3.1 Linear-Linear Intersection

If there are two succeeding blocks executing linear interpolation, and the ***second straight line*** is defined ***by specification the endpoint of the line on both axes in the selected plain and the angle of inclination of the line too***, the control will calculate the intersection point of the straight line assigned in the first block and the straight line specified in the second block.
Hereafter, the straight line specified in the second block in such a way is called ***overdetermined straight line***.

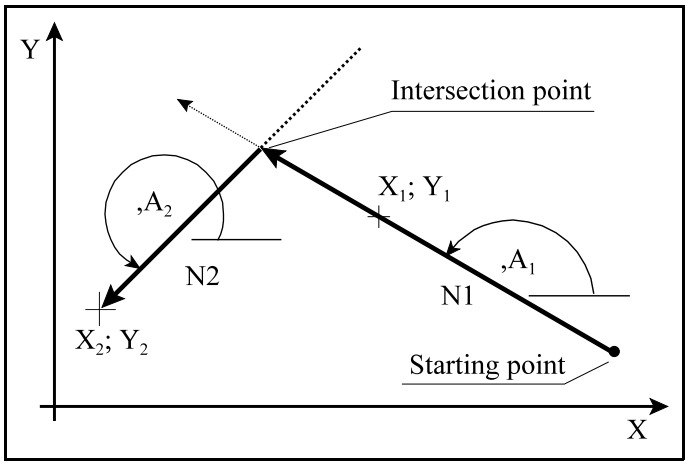


Fig. 17.3.1-1

The endpoint of the first block and the starting point of the second block will be the calculated intersection point.

G17 G41 (G42)	G18 G41 (G42)	G19 G41 (G42)
N1 G1 ,A_1 or	N1 G1 ,A_1 or	N1 G1 ,A_1 or
X_1 Y_1	X_1 Z_1	Y_1 Z_1
N2 G1G90 **X_2 Y_2 ,A_2**	N2 G1G90 **X_2 Z_2 ,A_2**	N2 G1G90 **Y_2 Z_2 ,A_2**

The intersection point is always calculated in the plane selected by the G17, G18 and G19.
The ***first block*** (N1) is specified either ***by its angle of inclination (,A) only***, and in this case the control draws a straight line from the starting point up to the intersection point at the appropriate angle of inclination; ***or***, ***an arbitrary point of the straight line*** (X_1, Y_1; X_1, Z_1; or Y_1, Z_1) different from the starting point ***is specified***, and in this case the control calculates the intersection point

using the straight line passing through these two points.
The ***coordinates*** specified ***in the second block*** (N2) are always interpreted by the control as ***absolute*** data (G90).
For example:

```
G17 G90 G41 D0...
G0 X90 Y10
N10 G1 ,A150
N20 X10 Y20 ,A225
G0 X0 Y20
...
```

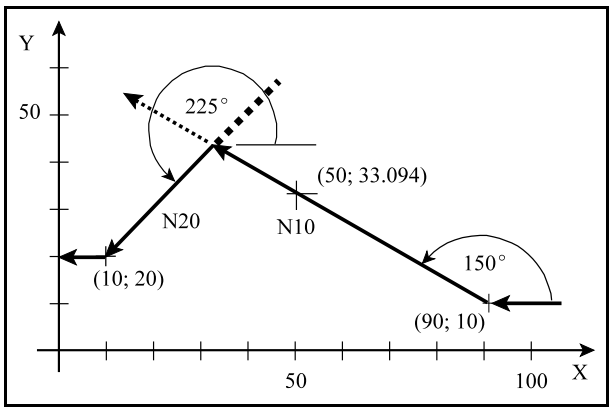


Fig. 17.3.1-2

The block N10 can also be given using the coordinates of a point of the straight line:

```
G17 G90 G41 D0...
G0 X90 Y10
N10 G1 X50 Y33.094
N20 X10 Y20 ,A225
G0 X0 Y20
...
```

It can be noted, that in this case the coordinates X and Y(X50 Y33.094) given in the block N10 are not considered by the control to be endpoint, but to be a transition point only between the starting point of the straight line and the point given.

Calculation of intersection point can also be combined with specification of chamfer or corner rounding. For example:

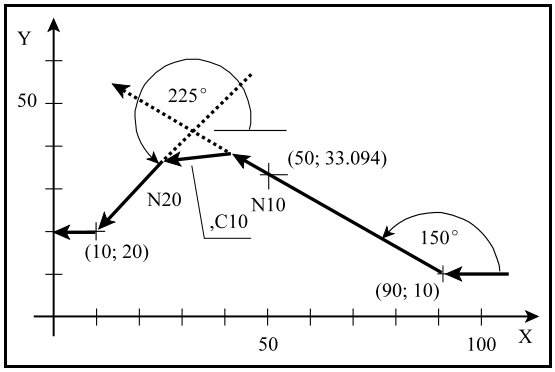


Fig. 17.3.1-3

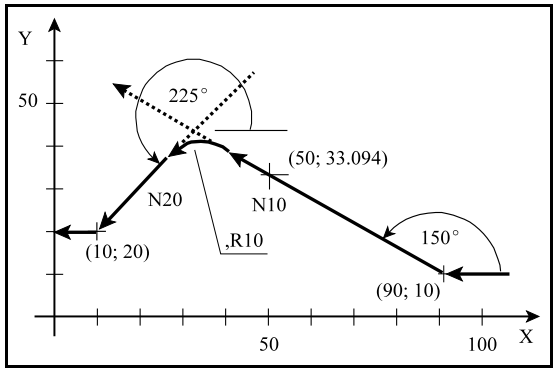


Fig. 17.3.1-4

```
G17 G90 G41 D0...
G0 X90 Y10
N10 G1 X50 Y33.094 ,C10
N20 X10 Y20 ,A225
G0 X0 Y20
...
```

```
G17 G90 G41 D0...
G0 X90 Y10
N10 G1 X50 Y33.094 ,R10
N20 X10 Y20 ,A225
G0 X0 Y20
...
```

In the examples above, the control measures the length of chamfer from the calculated intersection point and fits the corner rounding to the calculated intersection point, respectively.

17.3.2 Linear-Circular Intersection

If a circular block is given after a linear block in a way that ***the coordinates of the end point and the center point as well as the radius of the circle are specified, i.e. the circle is overdetermined***, the control will calculate intersection point between the straight line and the circle. ***The calculated intersection point is the end point of the first block and the start point of the second one.***

G17 G41 (G42)	G18 G41 (G42)	G19 G41 (G42)
N1 G1 ,A or	N1 G_1 ,A or	N1 G1 ,A or
X_1 Y_1	X_1 Z_1	Y_1 Z_1
N2 G2 (G3) G90 X_2 Y_2 I J	N2 G2 (G3) G90 X_2 Z_2 I K	N2 G2 (G3) G90 Y_2 Z_2 J K
R Q	R Q	R Q

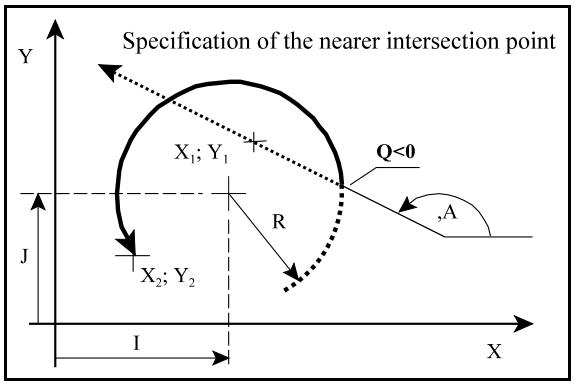


Fig. 17.3.2-1

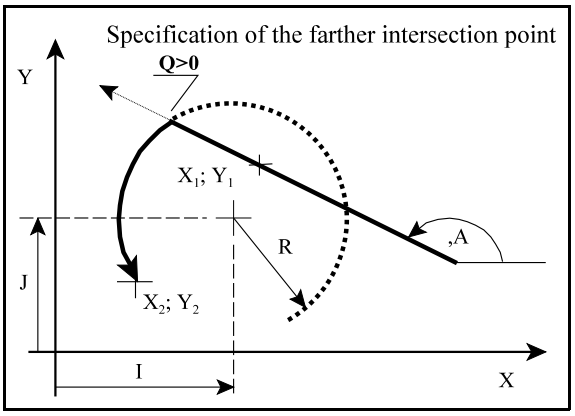


Fig. 17.3.2-2

The intersection point is always calculated by the control in the plane selected by G17, G18 and G19.

The ***first block*** (N1) is specified either ***by its angle of inclination (,A) only***, and in this case the control draws a straight line from the starting point up to the intersection point at the appropriate angle of inclination; ***or***, ***an arbitrary point of the straight line*** (X_1, Y_1; X_1, Z_1; or Y_1, Z_1) different from the starting point ***is specified***, and in this case the control calculates the intersection point using the straight line passing through these two points.

The ***coordinates*** specified ***in the second block*** (N2), thus ***the coordinates I, J and K*** defining ***the center point of the circle*** are also always interpreted by the control as ***absolute*** data (G90).

It can be specified ***at the address Q***, which one between the two resulting intersection points is to be calculated by the control. ***If the value of the address is smaller than zero (Q<0), the control will calculate the nearer intersection point in the direction of the straight line, however, if the value of the address is greater than zero (Q>0), the farther one will be calculated***. The direction of movement along the straight line is determined by the angle of inclination.

Let us see the following example:

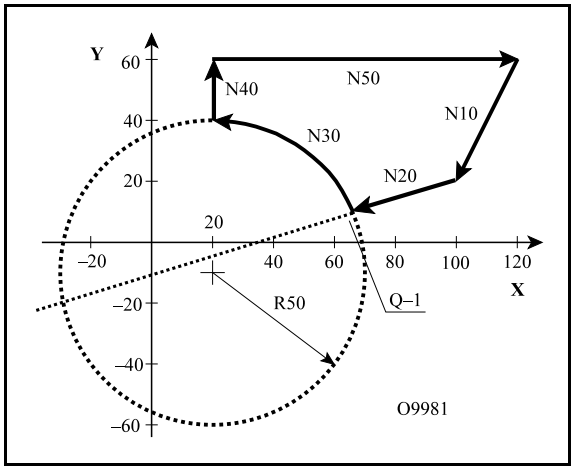


Fig. 17.3.2-3

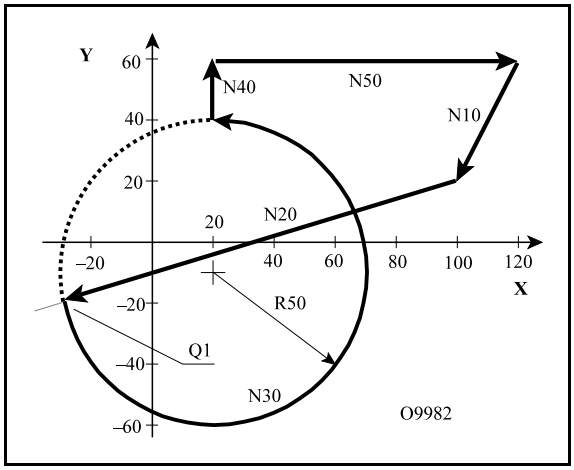


Fig. 17.3.2-4

```
O9981
N10 G17 G42 G0 X100 Y20 D0
N20 G1 X-30 Y-20
N30 G3 X20 Y40 I20J-10 R50 Q-1
N40 G40 G0 Y60
N50 X120
```

```
O9982
N10 G17 G42 G0 X100 Y20 D0
N20 G1 X-30 Y-20
N30 G3 X20 Y40 I20 J-10 R50 Q1
N40 G40 G0 Y60
N50 X120
```

The circular block N30 G3 is an overdetermined one since the coordinates of the center point of the circle (I20 J–10 in absolute value) as well as the radius of the circle (R50) are specified; the intersection point of the straight line specified in the block N20 and the circle specified in the block N30 will be calculated by the control.
In the program O9981, the control calculates the nearer intersection point in the direction of the straight line, because Q–1 is programmed in the circle block N30.
However, in the program O9982, the control calculates the farther intersection point in the direction of the straight line, because Q1 is programmed in the circle block N30.
The linear-circle intersection calculation can also be combined with specification of chamfer or corner rounding. For example:

```
N10 G17 G42 G0 X100 Y20 D0
N20 G1 X-30 Y-20 ,R15
N30 G3 X20 Y40 I20 J-10 R50 Q-1
N40 G40 G0 Y60
N50 X120
```

The control calculates the intersection point of the blocks N20 and N30, and then, due to the ,R15 given in the blockN20, it fits a corner rounds with radius of 15 mm to the intersection point.

17.3.3 Circular-Linear Intersection

If a linear block is given after a circular block in a way that ***the straight line is overdetermined***, i.e. ***the coordinate of the endpoint as well as the angle of inclination of the straight line are specified***, the control will calculate intersection point between the circle and the straight line. ***The calculated intersection point is the end point of the first block and the start point of the second one.***

G17 G41 (G42)	G18 G41 (G42)	G19 G41 (G42)
N1 G2 (G3) X_1 Y_1 I J	N1 G2 (G3) X_1 Z_1 I K	N1 G2 (G3) Y_1 Z_1 J K
or R	or R	or R
N2 G1 G90 X_2 Y_2 ,A Q	N2 G1 G90 X_2 Z_2 ,A Q	N2 G1 G90 Y_2 Z_2 ,A Q

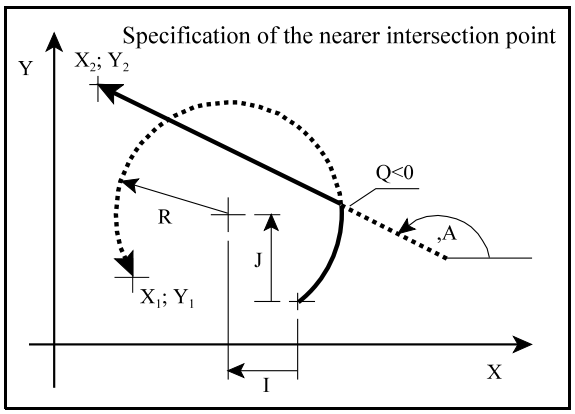


Fig. 17.3.3-1

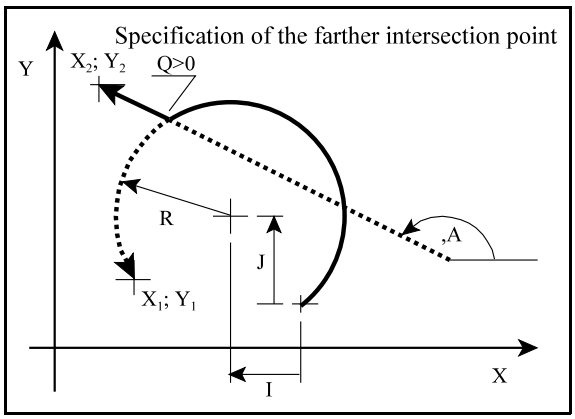


Fig. 17.3.3-2

The intersection point is always calculated by the control in the plane selected by G17, G18 and G19.

The first block (N1), i.e. the circle is specified either by ***one arbitrary pont of it*** (X_1, Y_1; X_1, Z_1; or Y_1, Z_1) as well as ***the coordinates of its center point*** (I J; I K; or J K), ***or*** instead of the coordinates of the center point, ***the radius (R) of the circle also can be specified***.

In the second block (N2), ***the straight line is overdetermined***, i.e. ***its endpoint coordinates (X_2 Y_2; X_2 Z_2; orY_2 Z_2) and its angle of inclination (,A) are specified***. The ***endpoint coordinates*** of the straight line are always interpreted by the control as ***absolute*** data (G90). It is always ***the angle of inclination of the straight vector pointing from the resulting intersection point to the given endpoint*** has to be specified at the address ,A; otherwise motions opposite to the intended ones will occur.

It can be specified ***at the address Q***, which one between the two resulting intersection points is to be calculated by the control. ***If the value of the address is smaller than zero (Q<0, for example Q–1), the control will calculate the nearer intersection point in the direction of the straight line; however, if the value of the address is greater than zero (Q>0, for example Q1), the farther one will be calculated. The direction of movement along the straight line is determined by the angle of inclination.***

Let us see the following example:

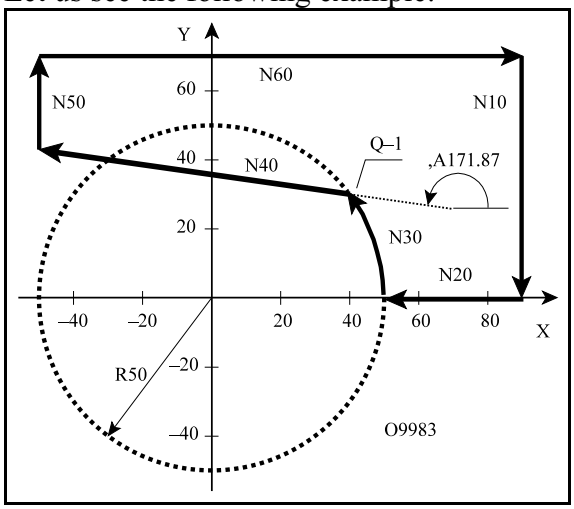


Fig. 17.3.3-3

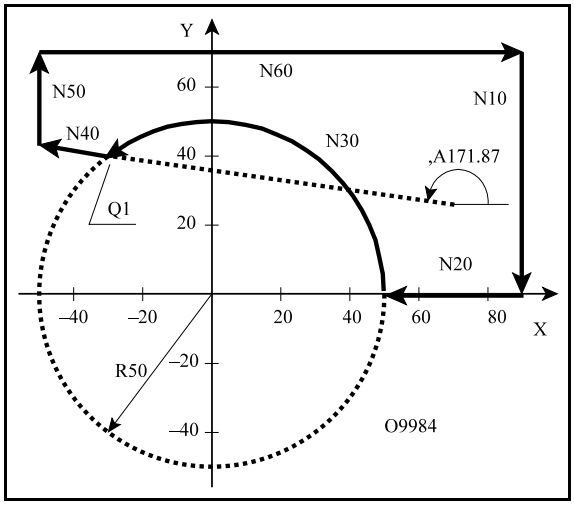


Fig. 17.3.3-4

```
O9983
N10 G17 G0 X90 Y0
N20 G42 G1 X50 D0
N30 G3 X-50 Y0 R50
N40 G1 X-50 Y42.857 ,A171.87
Q-1
N50 G40 G0 Y70
N60 X90
```

```
O9984
N10 G17 G0 X90 Y0
N20 G42 G1 X50 D0
N30 G3 X-50 Y0 R50
N40 G1 X-50 Y42.857 ,A171.87
Q1
N50 G40 G0 Y70
N60 X90
```

The linear block N40 is an overdetermined one, since the endpoint coordinates of the straight line (X–50 Y42.857) as well as its angle of inclination (,A171.87) are specified. For this reason, the coordinates X–50 Y0 of the circle programmed in the previous block N30 are not considered by the control to be endpoint values but as a point only through which the circle passes, and the endpoint will be the calculated intersection point.
In the program O9983, the nearer intersection point (Q–1) in the direction of motion is given, while in the program O9984 the farther one (Q1) is given.
The circle-linear intersection calculation can also be combined with specification of chamfer or corner rounding. For example:

```
O9983
N10 G17 G0 X90 Y0 M3 S200
N20 G42 G1 X50 D0
N30 G3 X-50 Y0 R50 ,R15
N40 G1 X-50 Y42.857 ,A171.87 Q-1
N50 G40 G0 Y70
N60 X90
```

In this example, a corner rounding with radius of 15 mm (,R15) is specified in the block N30. The control calculates the intersection point of the blocks N30 and N40, and fits the programmed corner rounding to the resulting contour.

17.3.4 Circular-Circular Intersection

If two succeeding circular blocks are given ***by specification of the endpoint and the center point coordinates as well as the radius of the second circle***, i.e. ***the second circle is overdetermined***, the control will calculate intersection point between the two circles. ***The calculated intersection point is the end point of the first block and the start point of the second one.***

G17 G41 (G42)
N1 G2 (G3) X_1 Y_1 I_1 J_1
or X_1 Y_1 R_1
N2 G2 (G3) G90 X_2 Y_2 I_2
J_2 R_2 Q

G18 G41 (G42)
N1 G2 (G3) X_1 Z_1 I_1 K_1
or X_1 Z_1 R_1
N2 G2 (G3) G90 X_2 Z_2 I_2
K_2 R_2 Q

G19 G41 (G42)
N1 G2 (G3) Y_1 Z_1 J_1 K_1
or Y_1 Z_1 R_1
N2 G2 (G3) G90 Y_2 Z_2 J_2
K_2 R_2 Q

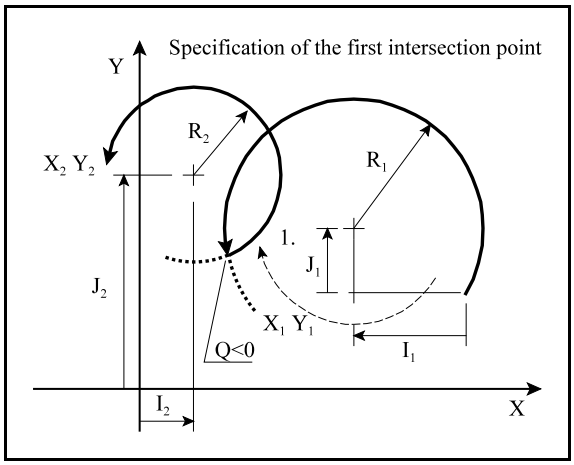


Fig. 17.3.4-1

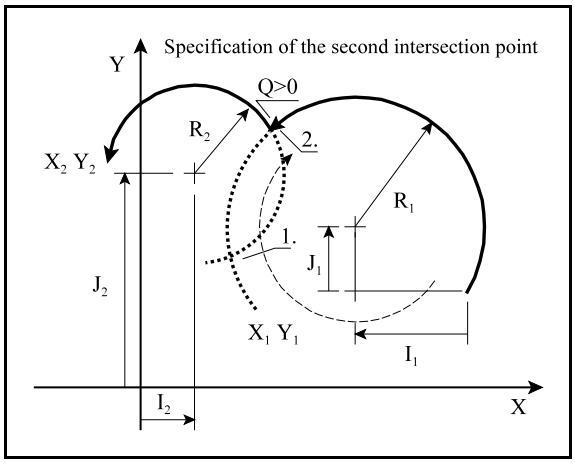


Fig. 17.3.4-2

The intersection point is always calculated by the control in the plane selected by G17, G18 and G19.
The first block (N1) is specified either ***by the coordinates of the center point of the circle*** (I_1 J_1; I_1 K_1; J_1 K_1) or ***by its radius*** (R_1). In this block, the coordinates are interpreted as the relative distance measured from the starting point, like specification of a circle.
The ***coordinates*** specified ***in the second block*** (N2), thus ***the coordinates I, J and K*** defining ***the center point of the circle*** are also always interpreted by the control as ***absolute*** data (G90).
It can be specified ***at the address Q***, which one between the two resulting intersection points is to be calculated by the control. ***If the value of the address is smaller than zero (Q<0, for example Q–1), the control will calculate the first intersection point; however, if the value of the address is greater than zero (Q>0, for example Q1), the second one will be calculated.***
The first intersection point is that one being passed through at first, going clockwise (independently of the programmed direction G2, G3).

Let us see the following example:

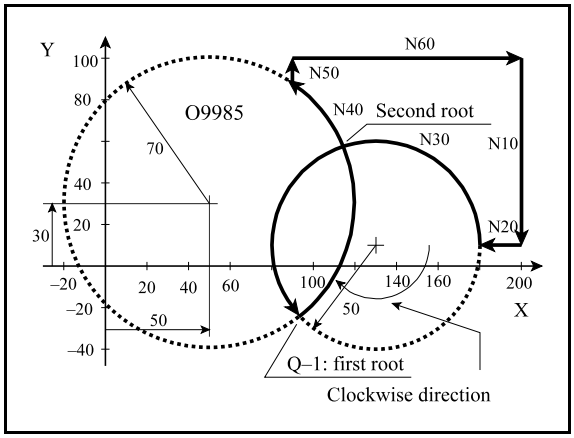


Fig. 17.3.4-3

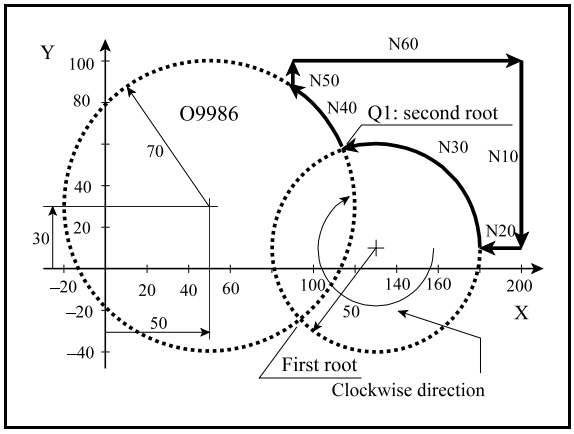


Fig. 17.3.4-4

```
O9985
N10 G17 G54 G0 X200 Y10
N20 G42 G1 X180 D1
N30 G3 X130 Y-40 R-50
N40 X90 Y87.446 I50J30 R70 Q-1
N50 G40 G0 Y100
N60 X200
```

```
O9986
N10 G17 G54 G0 X200 Y10
N20 G42 G1 X180 D1
N30 G3 X130 Y-40 R-50
N40 X90 Y87.446 I50 J30 R70 Q1
N50 G40 G0 Y100
N60 X200
```

The circular block N40 is an overdetermined block since the coordinates of the center point (I50 J30 as absolute value, and I as radius) as well as the radius (R70) is specified. For this reason, the coordinates X130 Y-40 of the circle programmed in the previous block N30 are not considered by the control to be endpoint values but to be a point only through which the circle passes, and the endpoint will be the calculated intersection point.
In the program O9985, the nearer intersection point (Q–1) in the clockwise direction, while in the program O9986 the farther one (Q1) is given.
The circle-circle intersection calculation can also be combined with specification of chamfer or corner rounding. For example:

```
O9986
N10 G17 G54 G0 X200 Y10
N20 G42 G1 X180 D1
N30 G3 X130 Y-40 R-50 ,R20
N40 X90 Y87.446 I50 J30 R70 Q1
N50 G40 G0 Y100
N60 X200
```

In this example, a corner rounding with radius of 20 mm (,R20) is specified in the block N30. The control calculates the intersection point of the blocks N30 and N40, and fits the programmed corner rounding to the resulting contour.

17.3.5 Chaining the Intersection Calculations

Blocks of intersection point calculation can be chained, i.e. ***several succeeding blocks can be selected for intersection point calculation***. The control ***calculates intersection point until it finds overdetermined straight lines or circles***.
Let us see the following example:

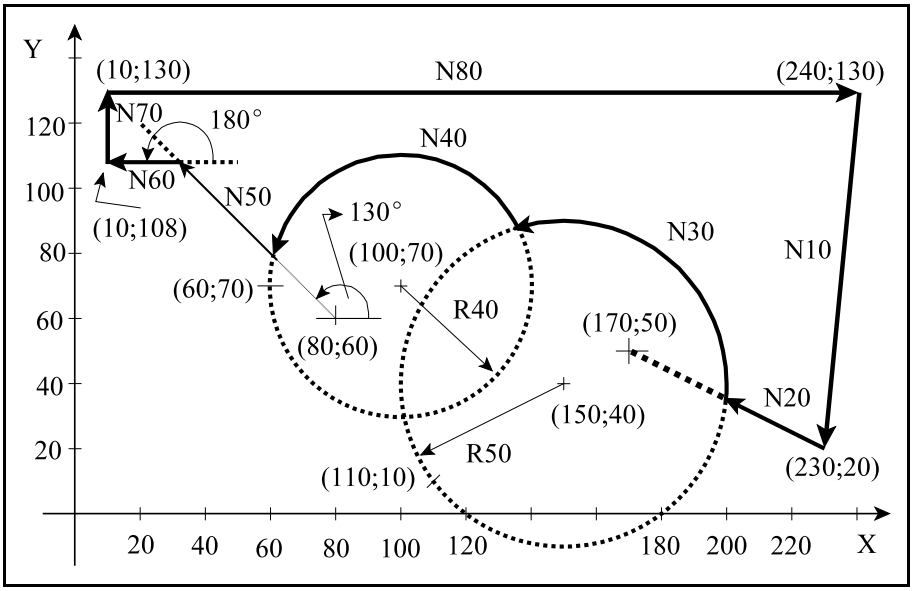


Fig. 17.3.5-1

```
N10 G17 G54 G0 G42 X230 Y20 D1 F300 S500 M3
N20 G1 X170 Y50
N30 G3 X110 Y10 I150 J40 R50 Q-1
N40 X60 Y70 I100 J70 R40 Q1
N50 G1 X80 Y60 ,A135 Q1
N60 X10 Y108 ,A180
N70 G40 G0 Y130
N80 X240
```

The blocks N30, N40, N50 and N60 are overdetermined ones.
The straight line N20 is not guided by the control up to its programmed endpoint (X170 Y50) because the block N30 is overdetermined, i.e. all the addresses I J and R are filled and the intersection point to be found is specified at the address Q.
Neither the circle block N30 is guided up to the programmed endpoint (X110 Y10) because the circular N40 is overdetermined, too.
The last overdetermined block in the program is the straight line N60. Since the succeeding linear block N70 is not overdetermined, thus the coordinates X10 Y108 programmed in the block N60 are considered by the control not to be a transit point of the straight line, but to be the endpoint coordinates of the block N60.
In general, those coordinate points of the overdetermined linear and circular blocks which are in the plane selected will only be considered by the control to be an endpoint coordinate if they are not followed by an overdetermined block.

18 Canned Cycles for Drilling

A canned cycle for drilling can consist of the following operations:

Operation 1: Positioning in the selected plane
Operation 2: Activity after positioning
Operation 3: Rapid traverse to the point R (point of approaching)
Operation 4: Activity at the point R
Operation 5: Drilling to the bottom point of the hole
Operation 6: Activity at the bottom point of the hole
Operation 7: Retracting to the point R
Operation 8: Activity at the point R
Operation 9: Rapid traverse retracting to the initial point
Operation 10: Activity at the initial point

Point R, point of approaching: The point to which the tool approaches the workpiece at rapid traverse rate.

Initial point: The position to which the drilling axis moves before starting the cycle.

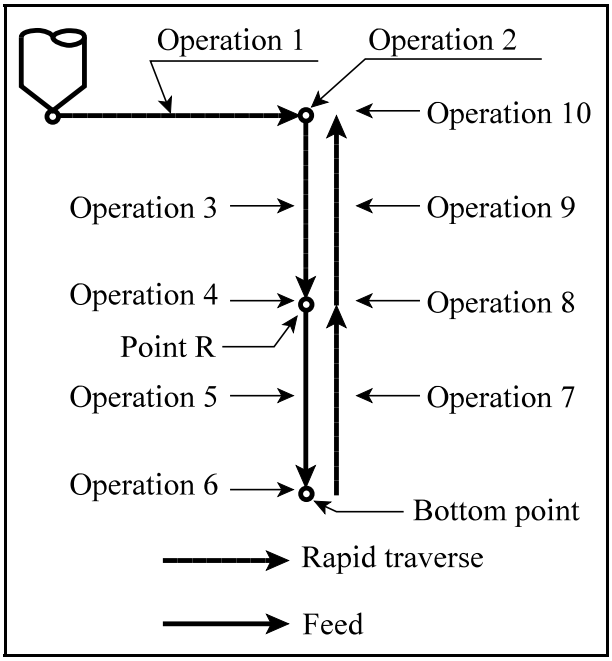


Fig. 18-1

The operations above describe drilling cycles generally; some of them can be omitted in specific cases.

The drilling cycles have ***positioning plane*** and ***drilling axis***. The positioning plane and the drilling axis are designated by the plane selecting instructions G17, G18 and G19.

Code G	Positioning plane	Drilling axis
G17	Plane X_p Y_p	Z_p
G18	Plane Z_p X_p	Y_p
G19	Plane Y_p Z_p	X_p

where: X_p: the axis X or an axis parallel with it
Y_p: the axis Y or an axis parallel with it
Z_p: the axis Z or an axis parallel with it
The axes U, V and W will be considered to be parallel axes if they are defined as parallel ones in the parameter N0103 Axis to Plane.

Drilling cycles can be configured using instructions **G98** and **G99**:
G98: in the course of the drilling cycle, ***the tool is retracted up to the initial point.*** It is a default position the control gets to after switching on, reset or deletion of the cycle mode.
G99: in the course of the drilling cycle, ***the tool is retracted up to the point R,*** therefore the operations 9 and 10 will be omitted.

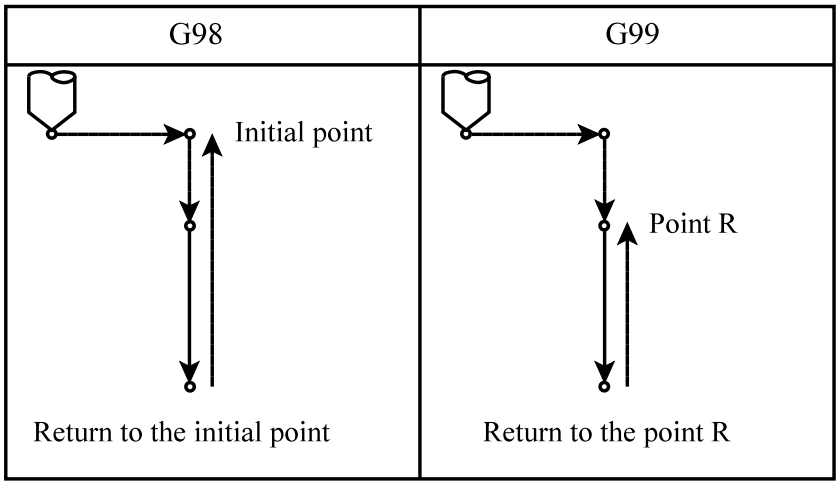
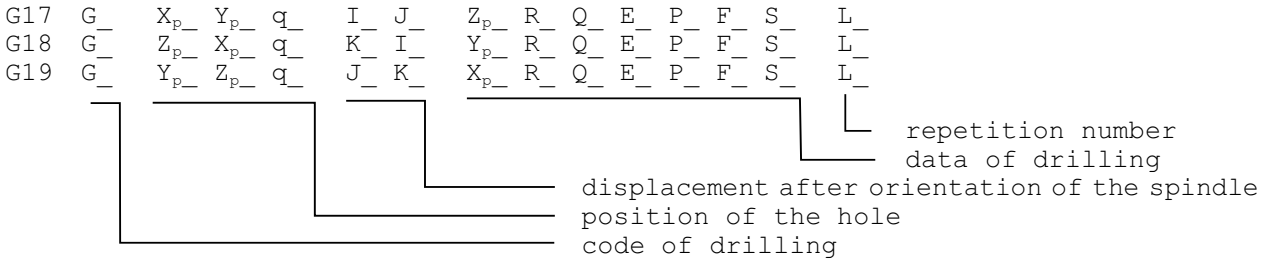


Fig. 18-2

The codes of the drilling cycles are: **G73, G74, G76, G81, ..., G89**
These codes set up the cycle mode enabling the cycle variables to be modal.
The code G80 cancels the cycle mode and deletes the stored cycle variables.

Coding used in the lathe control can also be applied in the milling machine control if the bit CSM of the parameter N1503 Drilling Cycles Config. is set in the state 1. In this case, the codes G83.1, G84.1 and G86.1 can be used instead of the codes G73, G74 and G76. It could be necessary when the intention is to use the turning cycles (G70, ..., G79) in the milling machine control too.

Addresses used in the drilling cycles and their meaning:



The code of drilling:

Interpretation of the codes will be given later.

The codes will be modal until instruction G80 or a code belonging to the group 1 of the codes G (interpolation group: G01, G02, G03, G33) is programmed.

So long as the cycle state is on by the instructions G73, G74, G76, G81, ..., G89, the modal cycle variable will also be modal between the drilling cycles of various type.

The initial point:

The initial point is the position of the axis selected for drilling, and it will be recorded:

– when the cycle mode is set up. For example, in the following case:

```
N1 G17 G90 G0 Z200
N2 G81 X0 Y0 Z50 R150
N3 X100 Y30 Z80
```

the position of the initial point will be Z=200 in either of the blocks N2 and N3.

– or, when a new drilling axis is selected. For example, in the following case:

```
N1 G17 G90 G0 Z200 W50
N2 G81 X0 Y0 Z50 R150
N3 X100 Y30 W20 R25
```

the position of the initial point is Z=200 in the block N2

the position of the initial point is W=50 in the block N3

In the case of changing the drilling axis, it is mandatory to program R otherwise the control will send the error message ‘2053 No bottom or R point’.

Position of the hole: X_p, Y_p, Z_p q

The input coordinate values, excluding the drilling axis, will be considered by the control to be the position of the hole. They can be ***the main axes of selected plane***, ***the axes parallel with them***, or ***other axis not selected for drilling (q: for example C)***.

The entered values can be incremental or absolute ones given as orthogonal or polar coordinates with metric or inch dimension.

The mirroring, rotating and scaling instructions are applicable to the entered coordinate values.

The control moves to the position of the hole at rapid traverse rate independently of which of the code of the group 1 is valid.

Displacement after orientation of the spindle: I, J, K

If the spindle can be oriented on a given machine, in the boring cycles G76 and G87 the tool can be retracted from the hole being shifted away from the surface in order not to scratch it. In this case, the direction of shifting can be specified at the addresses I, J and K. The addresses are interpreted by the control according to the selected plane:

G17: I, J
G18: K, I
G19: J, K

The addresses are ***always*** interpreted as ***incremental orthogonal data***. The address can be given in metric unit or in inch.

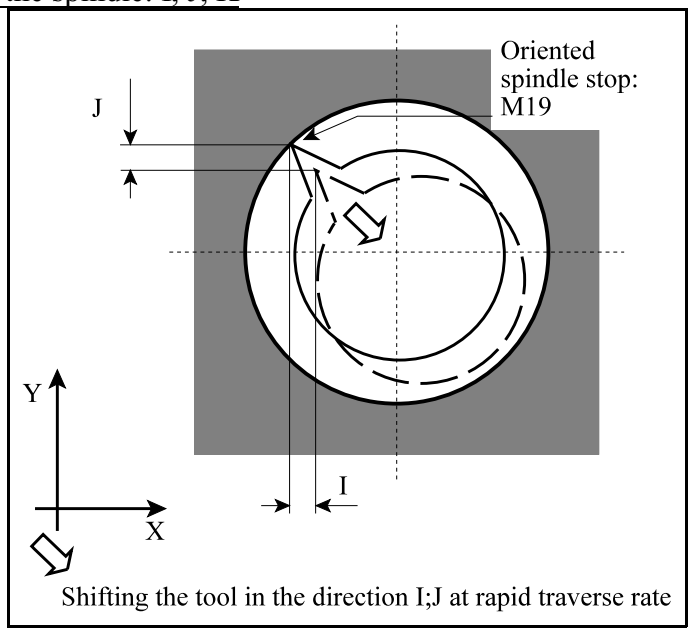


Fig. 18-3

The mirroring, rotating and scaling instruction are not applicable to the data I, J and K. The I, J and K are modal values. The code G80 or the codes of the interpolation group delete their values. The shifting is executed at rapid traverse rate

Data of drilling:

Bottom point of the hole: X_p, Y_p, Z_p

The bottom point of the hole has to be specified at the address of the drilling axis. The coordinate of the bottom point of the hole is always interpreted as orthogonal data. It can be absolute or incremental given in metric unit or in inch. If the value of the bottom point is given as incremental one, the displacement will be calculated from the point R.

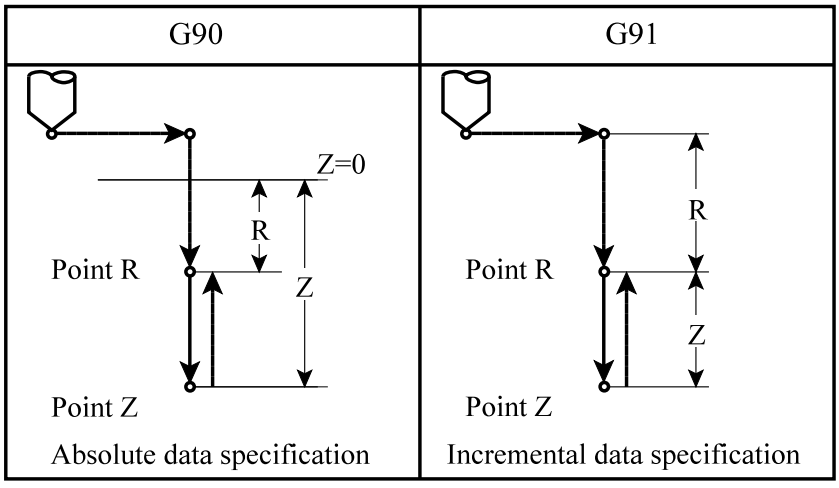


Fig. 18-4

The mirroring and scaling instructions are applicable to the data of the bottom point. The data of the bottom point is a modal value. The code G80 or the codes of the interpolation group delete

its value. The bottom point is always approached by the control at the actual feed rate being valid.

Point of approaching: R

The point of approaching has to be specified at the address R. The address R is always interpreted as orthogonal data. It can be absolute or incremental given in metric unit or in inch. If the address R is given as incremental data, its value will be calculated from the initial point. The mirroring and scaling data are applicable to the data of the point R. The data of the point R is a modal value. The code G80 or the codes of the interpolation group delete its value. The point R is always approached by the control at rapid traverse rate.

Depth of cut: Q

It is the value of the depth of cut in the cycles G73 and G83. It is always an incremental orthogonal positive data. The value of the depth of cut is a modal data. The code G80 or the codes of the interpolation group delete its value. The scaling instruction is not applicable to the depth of cut.

Auxiliary data: E

It is the extent of retraction in the cycle G73, and, in the cycle G83, it is the value motion to which executed at rapid traverse rate before infeed. It is always a positive incremental orthogonal data. The scaling instruction is not applicable to the auxiliary data. The value of the auxiliary data is a modal data. The code G80 or the codes of the interpolation group delete its value. If it not programmed, the control will have the required value from the parameters N1500 Return Val G73 and N1501 Clearance Val G83.

Dwell: P

It specifies the time of wait at the bottom of the hole. For specification of it, the rules described at G04 are valid. The value of dwell is a modal one. The code G80 or the codes of the interpolation group delete its value

Feed: F

It specifies the feed. Its value is modal. Only another data F rewrites it. The code G80 or another code does not delete it.

Extraction override: I (%)

In the cycles G85, G89, G84.2, G84.3, extraction is generally executed at a feed rate programmed at the address F. At the address I, the extraction override can be specified as a percental value. If it is not programmed, the control will have the value of override from parameter.

Spindle speed: S

Its value is modal. Only programming another data S rewrites it. The code G80 or another code does not delete it.

Repetition number: L

It defines the number as many times the cycle is repeated throughout the range of 1–99999999. If L is not filled, the value L=1 will be taken into account by the control. In the case of L=0, the data of the cycle will be stored but will not be executed. The value of L is valid only in the block where it is specified.

Examples on modality of drilling codes and cycle variables:

```
N1 G17 G0 Z_ M3
N2 G81 X_ Y_ Z_ R_ F_
```

At the beginning of the cycle mode, it is mandatory to specify the drilling data (Z, R).

```
N3 X_
```

Since the drilling data were specified in the block N2 and the same ones are required in the block N3, it is not necessary to fill them, i.e. G81, Z_, R_, F_ can be omitted. The position of the hole changes in the direction X only, the drill bit moves in this direction, and then it drills such a hole just like one was drilled in the block N2.

```
N4 G82 Y_ Z_ P_
```

The position of the hole is shifted in the direction Y. The method of drilling complies with the G82, the bottom point Z takes on a new value, the point of approaching and the feed (R and F) are taken from the block N2.

```
N5 G80 M5
```

The cycle mode and the modal cycle variables will be deleted excluding F.

```
N6 G85 Y_ Z_ R_ P_ M3
```

Since the drilling data were deleted due to the instruction G80 in the block N5, the values Z, R and P have to be specified again.

```
N7 G0 X_ Y_
```

The cycle mode and the modal cycle variables will be deleted excluding F.

Examples on using the cycle repetition:

If holes of the same kind with equal spacing between them are to be drilled using the same parameters, the repetition number can be specified at the address L. The L is valid only in the block in which it is specified.

```
N1 G90 G17 G0 X0 Y0 Z100 M3
N2 G91 G81 X100 Z-40 R-97 F50 L5
```

Due to the instructions above, the control drills 5 holes of the same kind along the axis X with spacing of 100 mm between them. The position of the first hole is X=100 and Y=0.

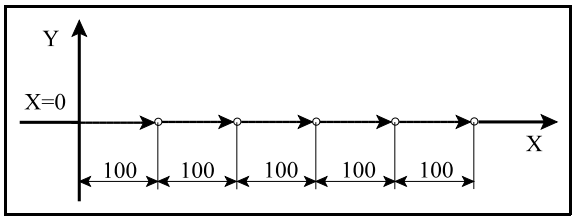


Fig. 18-5

Due to the G91, the position of the hole is given incrementally. If it had been given as an absolute data (G90), the drilling operation would have been executed five times in succession in the point of the coordinates X100, Y0.

```
N1 G90 G17 G16 G0 X200 Y-60 Z50
N2 G81 YI60 Z-40 R3 F50 L6
```

In accordance with the instructions above, the control drills 6 holes along the bolt hole with the diameter of 200 mm, with spacing of 60° between them. The position of the first hole is X=100 and Y=173.205.

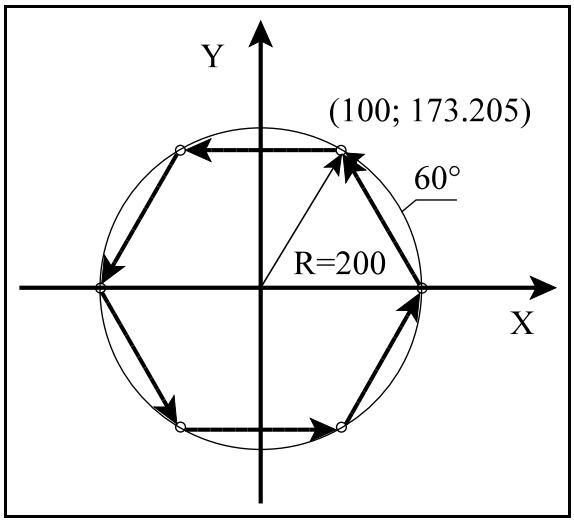


Fig. 18-6

18.1 Detailed Description of the Drilling Cycles

18.1.1 High-speed Peck Drilling Cycle (G73)

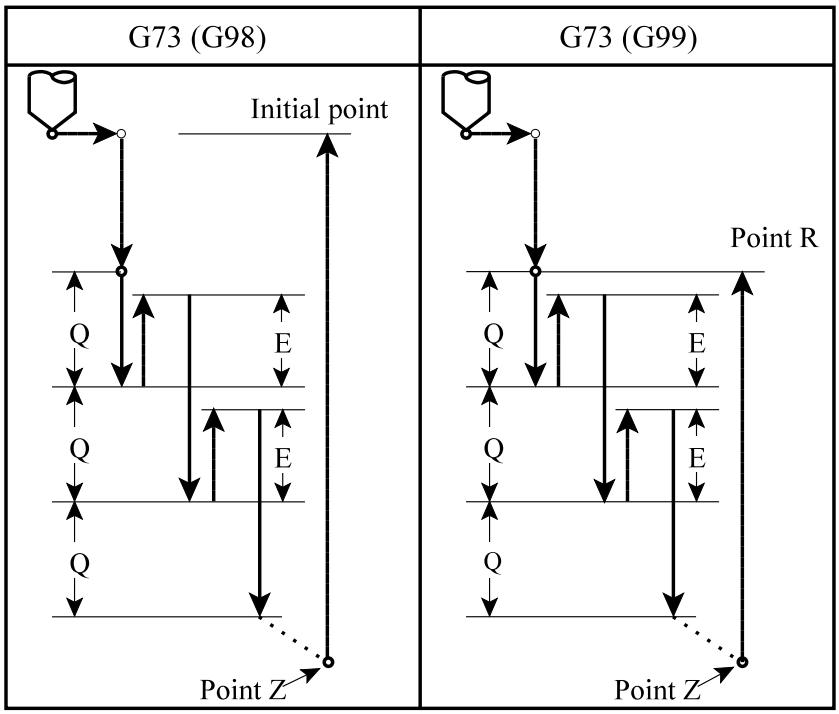


Fig. 18.1.1-1

The variables used in the cycle are the following:

G17 **G73** X_p__ Y_p__ Z_p__ R__ Q__ E__ F__ L__
G18 **G73** Z_p__ X_p__ Y_p__ R__ Q__ E__ F__ L__
G19 **G73** Y_p__ Z_p__ X_p__ R__ Q__ E__ F__ L__

The operations of the cycle are the following:

Operation 1: Positioning in the selected plane at rapid traverse rate
Operation 2: –
Operation 3: Rapid traverse to the point R (point of approaching)
Operation 4: –
Operation 5: Drilling to the bottom point at the feed rate F
Operation 6: –
Operation 7: In the case of G99: Retracting to the point R at rapid traverse rate
Operation 8: –
Operation 9: In the case of G98: Retracting to the initial point at rapid traverse rate
Operation 10: –

Execution of the Operation 5 of drilling is as follows:

– drilling the ***depth of cut*** specified at the address ***Q*** into the workpiece at the feed rate;
– ***retracting*** with a value specified at the address ***E*** or in the parameter N1500 Return Val G73, at rapid traverse rate;
– drilling the depth of cut Q again into the workpiece from the bottom point of the previous drilling;
– retracting with a value specified at the address E at rapid traverse rate.

The drilling proceeds up to the bottom point specified at the address Z.

18.1.2 Left-Handed Tapping Cycle Using Spring Tap (G74)

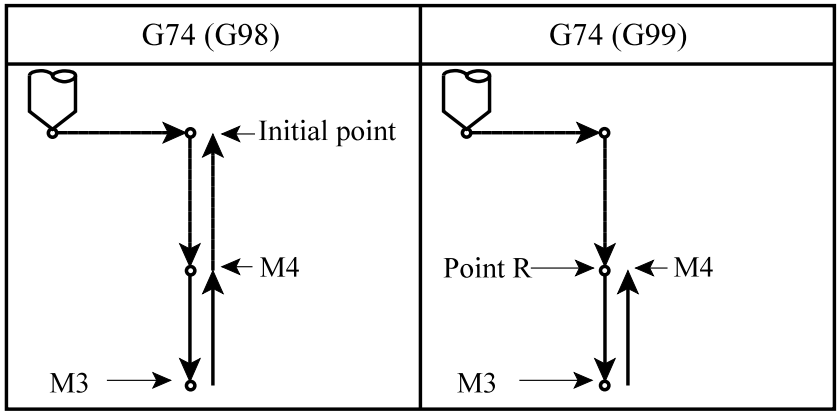


Fig. 18.1.2-1

Using this cycle is allowed when spring tap is applied.

The variables used in the cycle are the following:

G17 **G74** X_p__ Y_p__ Z_p__ R__ P__ F__ L__
G18 **G74** Z_p__ X_p__ Y_p__ R__ P__ F__ L__
G19 **G74** Y_p__ Z_p__ X_p__ R__ P__ F__ L__

Prior to starting the cycle, the direction of spindle rotation M4 (counter-clockwise) must be switched on and programmed.

The value of feed has to be specified according to the lead of the tap:

– in the state G94 (feed per minute):

$$F=P\times S$$

where: P: the lead of the thread in mm/rev or inch/rev;
S: the spindle speed in rev/min.

– in the state G95 (feed per revolution):

$$F=P$$

where: P: the lead of the thread in mm/rev or inch/rev.

The operations of the cycle are the following:

Operation 1: Positioning in the selected plane at rapid traverse rate
Operation 2: –
Operation 3: Rapid traverse to the point R (point of approaching)
Operation 4: –
Operation 5: Drilling to the bottom point at the feed rate F, ***override and stop are disabled***
Operation 6: Dwell for the value specified at the address P
Reversal of spindle rotation: M3
Operation 7: Retracting to the point R at feed rate F, ***override and stop are disabled***
Operation 8: Reversal of spindle rotation: M4
Operation 9: In the case of G98: Retracting to the initial point at rapid traverse rate
Operation 10: –

18.1.3 Boring Cycle with Automatic Tool Shift (G76)

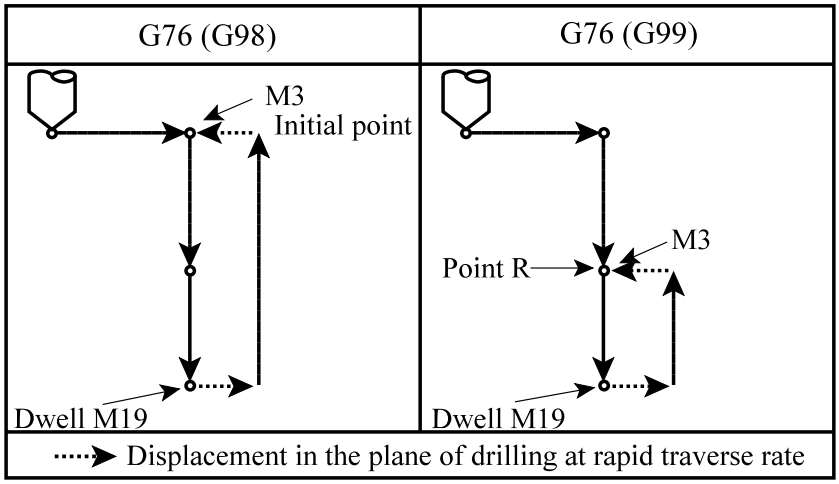


Fig. 18.1.3-1

The cycle G76 can only be used if spindle orientation is built in the machine tool. It is the state 1 of the bit ORI of the parameter N0607 Spindle Config which indicates this capability to the control. Otherwise, the control will send the error message '2137 G76, G87 error'.
Since spindle orientation and tool offset specified at the addresses I, J and K are executed by the cycle after the boring, the surface will not be scratched during tool retraction.
The variables used in the cycle are the following:

G17 **G76** X_p__ Y_p__ I__ J__ Z_p__ R__ P__ F__ L__
G18 **G76** Z_p__ X_p__ K__ I__ Y_p__ R__ P__ F__ L__
G19 **G76** Y_p__ Z_p__ J__ K__ X_p__ R__ P__ F__ L__

Prior to starting the cycle, the instruction M3 has to be issued.
The operations of the cycle are the following:

Operation 1: Positioning in the selected plane at rapid traverse rate
Operation 2: –
Operation 3: Rapid traverse to the point R (point of approaching)
Operation 4: –
Operation 5: Boring to the bottom point at the feed rate F
Operation 6: Dwell for the value specified at the address P
Spindle orientation: M19
In the selected plane, tool shift with the values specified at the addresses I, J and K, at rapid traverse rate
Operation 7: In the case of G99: Retracting to the point R at rapid traverse rate
Operation 8: In the case of G99:
In the selected plane, tool reset with the values specified at the addresses I, J and K, at rapid traverse rate
Restarting the spindle in the direction M3
Operation 9: In the case of G98: Retracting to the initial point at rapid traverse rate
Operation 10: In the case of G98:
In the selected plane, tool reset with the values specified at the addresses I, J and K, at rapid traverse rate
Restarting the spindle in the direction M3

18.1.4 Cancelling the Cycle State (G80)

Due to this code, the cycle state will be cancelled, and the cycle variables will be deleted.
The Z and the R take on incremental 0, all the rest variables take on 0.
If coordinates are programmed in the block **G80** and any other instruction is nott issued, the motion will be executed according to the interpolation code (the group 1 of the codes G or the interpolation group) that was valid prior to activating the cycle.

18.1.5 Drilling Cycle with Retraction at Rapid Traverse Rate (G81)

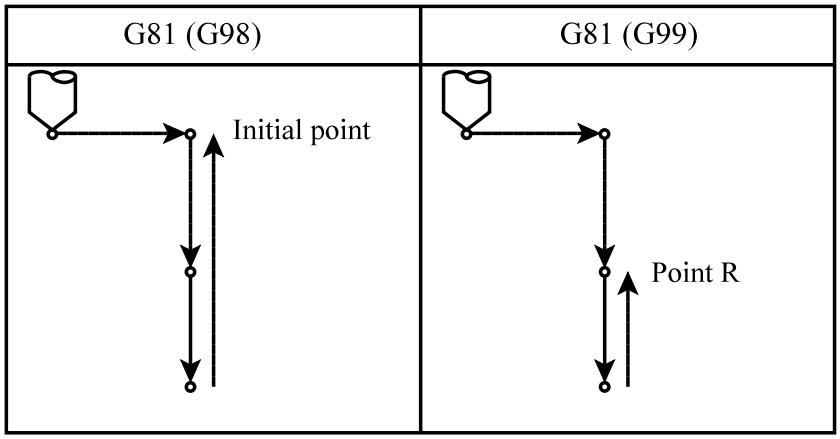


Fig. 18.1.5-1

The variables used in the cycle are the following:

G17 **G81** X_p__ Y_p__ Z_p__ R__ F__ L__
G18 **G81** Z_p__ X_p__ Y_p__ R__ F__ L__
G19 **G81** Y_p__ Z_p__ X_p__ R__ F__ L__

The operations of the cycle are the following:

Operation 1: Positioning in the selected plane at rapid traverse rate
Operation 2: –
Operation 3: Rapid traverse to the point R (point of approaching)
Operation 4: –
Operation 5: Drilling to the bottom point at the feed rate F
Operation 6: –
Operation 7: In the case of G99: Retracting to the point R at rapid traverse rate
Operation 8: –
Operation 9: In the case of G98: Retracting to the initial point at rapid traverse rate
Operation 10: –

18.1.6 Drilling Cycle with Dwell and with Retraction at Rapid Traverse (G82)

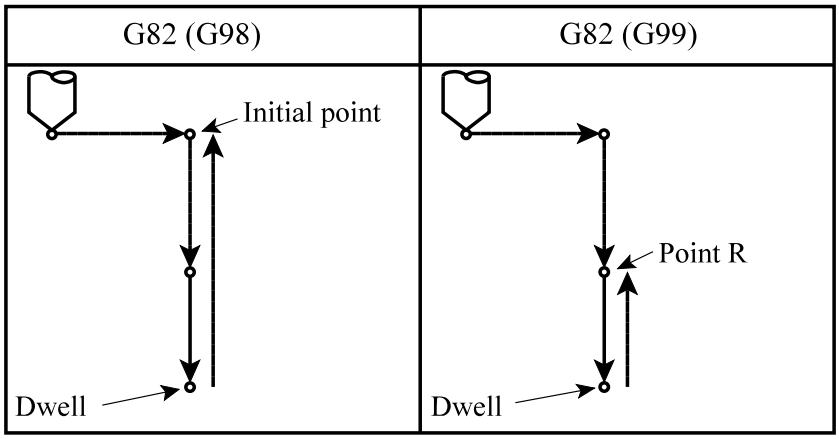


Fig. 18.1.6-1

The variables used in the cycle are the following:

G17 **G82** X_p__ Y_p__ Z_p__ R__ P__ F__ L__
G18 **G82** Z_p__ X_p__ Y_p__ R__ P__ F__ L__
G19 **G82** Y_p__ Z_p__ X_p__ R__ P__ F__ L__

The operations of the cycle are the following:

Operation 1: Positioning in the selected plane at rapid traverse rate
Operation 2: –
Operation 3: Rapid traverse to the point R (point of approaching)
Operation 4: –
Operation 5: Drilling to the bottom point at the feed rate F
Operation 6: Dwell for the value specified at the address P
Operation 7: In the case of G99: Retracting to the point R at rapid traverse rate
Operation 8: –
Operation 9: In the case of G98: Retracting to the initial point at rapid traverse rate
Operation 10: –

18.1.7 Peck Drilling Cycle (G83)

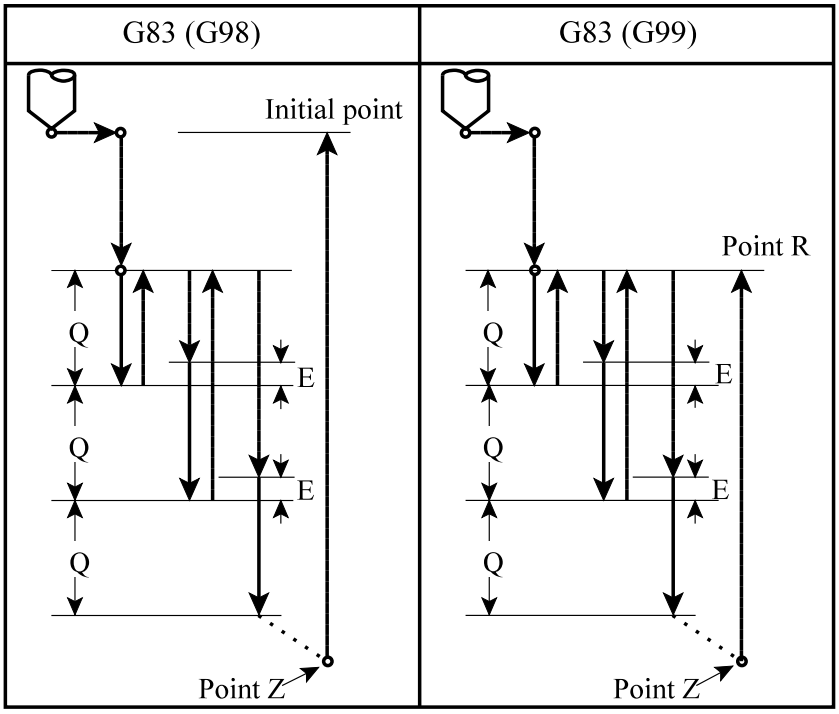


Fig. 18.1.7-1

The variables used in the cycle are the following:

G17 **G83** X_p__ Y_p__ Z_p__ R__ Q__ E__ F__ L__
G18 **G83** Z_p__ X_p__ Y_p__ R__ Q__ E__ F__ L__
G19 **G83** Y_p__ Z_p__ X_p__ R__ Q__ E__ F__ L__

The operations of the cycle are the following:

Operation 1: Positioning in the selected plane at rapid traverse rate
Operation 2: –
Operation 3: Rapid traverse to the point R (point of approaching)
Operation 4: –
Operation 5: Drilling to the bottom point at the feed rate F
Operation 6: –
Operation 7: In the case of G99: Retracting to the point R at rapid traverse rate
Operation 8: –
Operation 9: In the case of G98: Retracting to the initial point at rapid traverse rate
Operation 10: –

Execution of the Operation 5 of drilling is as follows:

– drilling the ***depth of cut*** specified at the address ***Q*** into the workpiece at the feed rate;
– retracting to the point R, at rapid traverse rate;
– ***approaching*** the previous depth ***to a distance E***;
– drilling the depth Q again into the workpiece from the bottom point of the previous drilling, at the feed rate (displacement is E+Q);
– retracting to the point R, at rapid traverse rate.

The drilling proceeds up to the bottom point specified at the address Z.
The distance E is taken from the address ***E*** or from the parameter N1501 Clearance Val G83.

18.1.8 Right-Handed Tapping Cycle Using Spring Tap (G84)

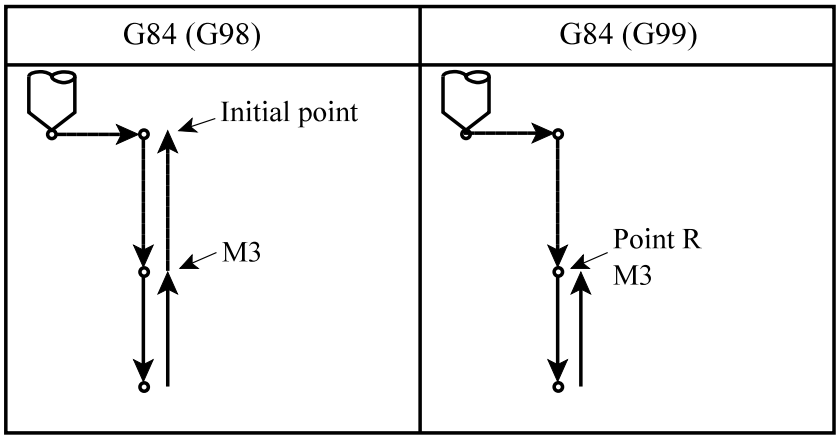


Fig. 18.1.8-1

Using this cycle is allowed when spring tap is applied.

The variables used in the cycle are the following:

G17 **G84** X_p__ Y_p__ Z_p__ R__ P__ F__ L__
G18 **G84** Z_p__ X_p__ Y_p__ R__ P__ F__ L__
G19 **G84** Y_p__ Z_p__ X_p__ R__ P__ F__ L__

Prior to starting the cycle, the direction of spindle rotation M3 (clockwise) must be switched on.
The value of feed has to be specified according to the lead of the tap:
– in the state G94 (feed per minute):

$$F=P\times S$$

where: P: the lead of the thread in mm/rev or inch/rev;
S: the spindle speed in rev/min.

– in the state G95 (feed per revolution):

$$F=P$$

where: P: the lead of the thread in mm/rev or inch/rev.

The operations of the cycle are the following:

Operation 1: Positioning in the selected plane at rapid traverse rate
Operation 2: –
Operation 3: Rapid traverse to the point R (point of approaching)
Operation 4: –
Operation 5: Drilling to the bottom point at the feed rate F, ***override and stop are disabled***
Operation 6: Dwell for the value specified at the address P
Reversal of spindle rotation: M4
Operation 7: Retracting to the point R at feed rate F, ***override and stop are disabled***
Operation 8: Reversal of spindle rotation: M3
Operation 9: In the case of G98: Retracting to the initial point at rapid traverse rate
Operation 10: –

18.1.9 Rigid Tapping Cycle (G84.2, G84.3)

Cycles of rigid tapping thread can be applied only on the machines the spindle of which is equipped with an encoder for positioning, and the spindle drive can be fed back for position control (the value of the bit INX of the parameter N0607 Spindle Config is 1). Otherwise, the control will send the error message '2138 Spindle can not be indexed'.

In the case of rigid tapping, the quotient of the feed of the drill axis and the spindle speed must be equal to the thread lead of the tap. In other words, under ideal conditions of rigid tapping, the quotient F=P/S must be constant from moment to moment,

where: P: the lead of the thread in mm/rev or inch/rev;
F: the feed in mm/min or inch/min;
S: the spindle speed in rev/min.

In the cycles G74 and G84 (cycle of tapping left-hand thread using spring tap and cycle of tapping left-hand thread using spring tap, respectively), the spindle speed and the feed of the drill axis are controlled independently of each other. Accordingly, the requirement mentioned above cannot be met to the full. It is particularly true at the bottom of the hole where the feed of the drill axis and the spindle speed should be decreased up to zero and then, in the opposite direction, they should be increased, in synchrony with each other. In the case above, as far as the control technique is concerned, complying with this condition is not possible at all. This problem can be avoided by inserting the tap into a spring balancing adapter to be set in the spindle; this adapter compensates fluctuation of the value of the quotient F/S.

The principle of control is different in the case of the cycles G84.2 and G84.3 that make it possible to avoid the use of spring tap. In these cycles, the control continuously maintains the constant value of the quotient F/S from moment to moment.
As far as the control technique is concerned, in the former case the control regulates only the spindle speed, while in the latter case, it regulates the spindle position too. In the cycles G84.2 and G84.3, the motions of the drill axis and the spindle are connected with each other using linear interpolation. With this method, the constant value of the quotient F/S is maintained in the phases of acceleration and deceleration too.

G84.2: right-handed rigid tapping cycle
G84.3: left-handed rigid tapping cycle

The variables used in the cycle are the following:

G17 **G84._** X_p__ Y_p__ Z_p__ R__ P__ F__ I__ S__ L__
G18 **G84._** Z_p__ X_p__ Y_p__ R__ P__ F__ I__ S__ L__
G19 **G84._** Y_p__ Z_p__ X_p__ R__ P__ F__ I__ S__ L__

At the end of the cycle, the spindle remains in the indexed state (the position control loop is closed); it is the programmer who has to restart it, if necessary.

The value of feed has to be specified according to the lead of the tap:
– in the state G94 (feed per minute):

$$F=P\times S$$

where: P: the lead of the thread in mm/rev or inch/rev;
S: the spindle speed in rev/min.

– in the state G95 (feed per revolution):

$$F=P$$

where: P: the lead of the thread in mm/rev or inch/rev.

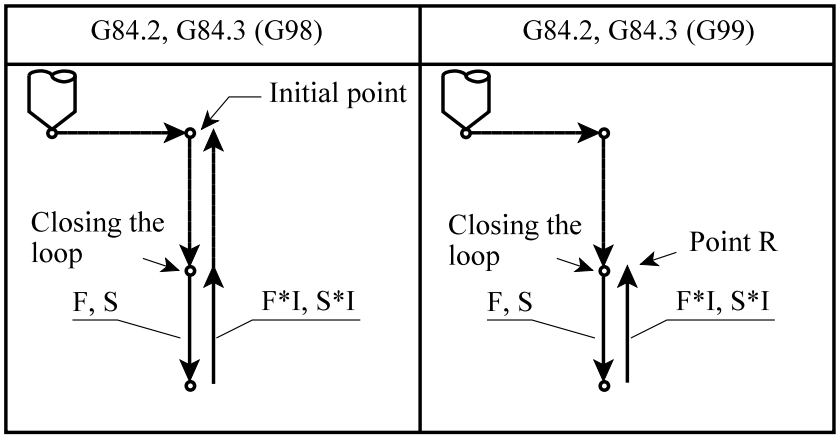


Fig. 18.1.9-1

The operations of the cycle in the case of G84.2 are the following:

Operation 1: Positioning in the selected plane at rapid traverse rate

Operation 2: –

Operation 3: Rapid traverse to the point R (point of approaching)

Operation 4: If the value of the bit #1 TSC of the parameter N1503 Drilling Cycles Config.
=0, the control will not orientate the spindle and will only close the position control loop. The code of closing the loop is determined by the parameter N0823 M Code for Closing S Loop. This code will be transmitted by the control to the PLC (faster execution).
=1, the control will orientate the spindle prior to tapping and will transmit the instruction M19 to the PLC (finding the way back into the thread).

Operation 5: Linear interpolation between the drill axis and the spindle in the clockwise direction of rotation (+) in the case of G84.2, and in the counter-clockwise direction of rotation (–) in the case of G84.3.

Operation 6: Dwell for the value specified at the address P

Operation 7: Linear interpolation between the drill axis and the spindle in the counter-clockwise direction of rotation (–) in the case of G84.2, and in the clockwise direction of rotation (+) in the case of G84.3.

Extraction override I (%)
The extraction feed override being either programmed at the address I or taken from parameter will be effective in the only case if it is enabled by writing the value 1 at the bit #4 EOE of the parameter N1503 Drilling Cycles Config.
If value is not given at the address I in the block, the extraction feed override in the operation 7 will be the value of the parameter N1506 Extraction Override in Tapping given in %.The feed override button also produces effect on speed calculated in such a way, so the value of feed

will be calculated according to the following formula:

Fextraction=Fprogrammed×Feed override×Extraction Override in Tapping/100

Acceleration during extraction
During extraction, a value different from (smaller than) the acceleration value set for the spindle can be specified in the parameter N1507+n Rn Acc in Tapping if it is enabled by writing the value 1 at the bit #6 EAE of the parameter N1503 Drilling Cycles Config.. In this case, the acceleration value will be taken from the parameters N1523+n Rn Acc in Extract being different range by range (n=1 ... 8).

Operation 8: –
Operation 9: In the case of G98, retracting to the initial point at rapid traverse rate
Operation 10: –

18.1.10 Peck Rigid Tapping Cycle (G84.2, G84.3)

The code of the cycle is as follows:

G84.2 Q__: right-handed peck rigid tapping cycle
G84.3 Q__: left-handed peck rigid tapping cycle

In the cycle, the control will apply ***chip breaking in the case*** if ***depth of cut*** is programmed ***at the address Q***.

The variables used in the cycle are the following:

G17 **G84._** X_p__ Y_p__ Z_p__ R__ Q__ E__ P__ F__ S__ I__ L__
G18 **G84._** Z_p__ X_p__ Y_p__ R__ Q__ E__ P__ F__ S__ I__ L__
G19 **G84._** Y_p__ Z_p__ X_p__ R__ Q__ E__ P__ F__ S__ I__ L__

At the end of the cycle, the spindle remains in the indexed state (the position control loop is closed); it is the programmer who has to restart it, if necessary. The meaning of the codes **G98** and **G99** is the same as it was in the case of all the other drilling cycles.

It is the bit #3 PTC of the parameter N1503 Drilling Cycles Config. that determines the ***manner of chip breaking***. If the value of the bit #3 PTC is:

=0, fast chip breaking (according to the G73) will be applied;
=1, normal chip breaking (according to the G83) will be applied, and extraction will be executed to the point R.

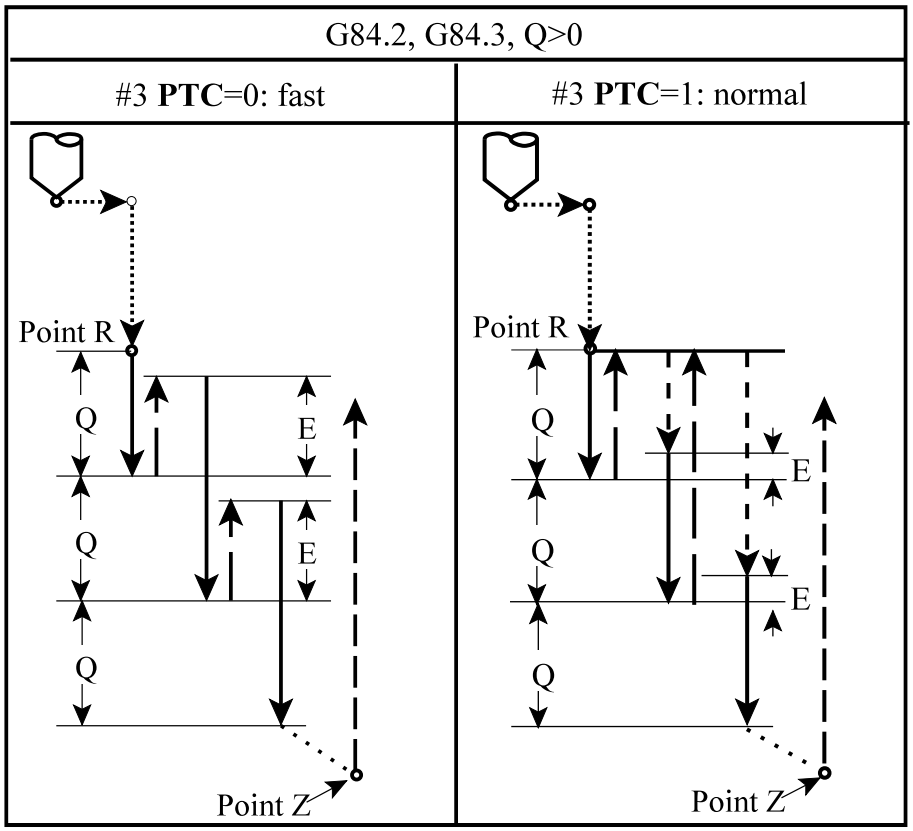


Fig. 18.1.10-1

Interpretation of the address E

In the case of the ***fast chip breaking*** (PTC=0), it is the address E where the ***value of the returning the tool*** can be specified. If the address E is not filled, the control will have the distance from the parameter N1504 Return Val in Tapping.

In the case of the ***normal chip breaking*** (PTC=1), it is the address E where the ***value of the approaching distance*** after return can be specified. If the address E is not filled, the control will have the distance from the parameter N1505 Clearance Val in Tapping.

Extraction override I (%)

In all the extraction phases, the extraction feed override, being either programmed at the address I or taken from parameter, will be effective in the only case if it is enabled by writing the value 1 at the bit #4 EOE of the parameter N1503 Drilling Cycles Config.

If value is not given at the address I in the block, the extraction feed override will be the value of the parameter N1506 Extraction Override in Tapping given in %.The feed override button also produces effect on speed calculated in such a way, so the value of feed will be calculated according to the following formula:

Fextraction=Fprogrammed×Feed override×Extraction Override in Tapping/100

Acceleration during extraction

During extraction, a value different from (smaller than) the acceleration value set for the spindle can be specified in the parameter N1507+n Rn Acc in Tapping if it is enabled by writing the value 1 at the bit #6 EAE of the parameter N1503 Drilling Cycles Config.. In this case, the acceleration value will be taken from the parameters N1523+n Rn Acc in Extract being different range by range (n=1 ... 8).

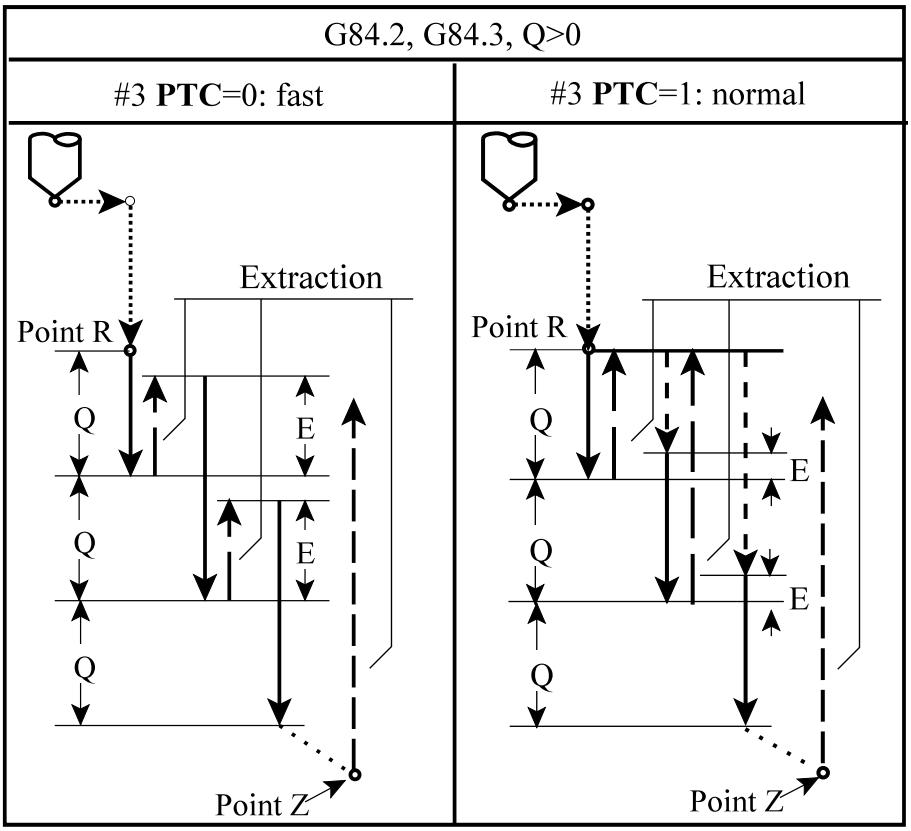


Fig. 18.1.10-2

Return override (%) in the case of normal chip breaking

In the case of normal chip breaking (the bit #3 PTC of the parameter N1503 Drilling Cycles Config. is 1), in the course of ***returning,*** application of an override different from 100% for the programmed F will be allowed if it is enabled by writing the value 1 at the bit #5 ROE of the parameter N1503 Drilling Cycles Config..

Thus, the control will have the value of override from the value in % of the parameter N1507 Return Override in Tapping.

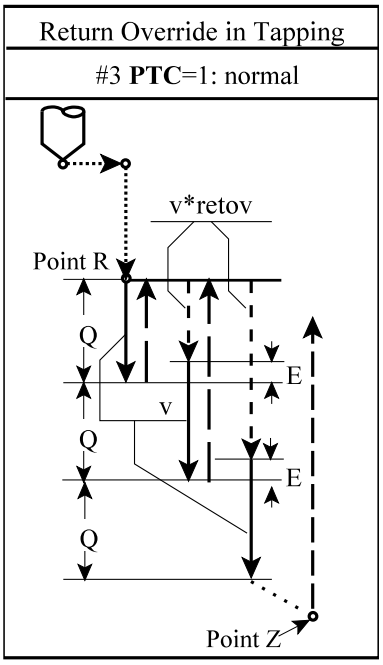


Fig. 18.1.10-3

The subject-matter mentioned in the previous subchapter apply to the other input parameters of the cycle.

18.1.11 Boring Cycle with Retraction at Feed Rate (G85)

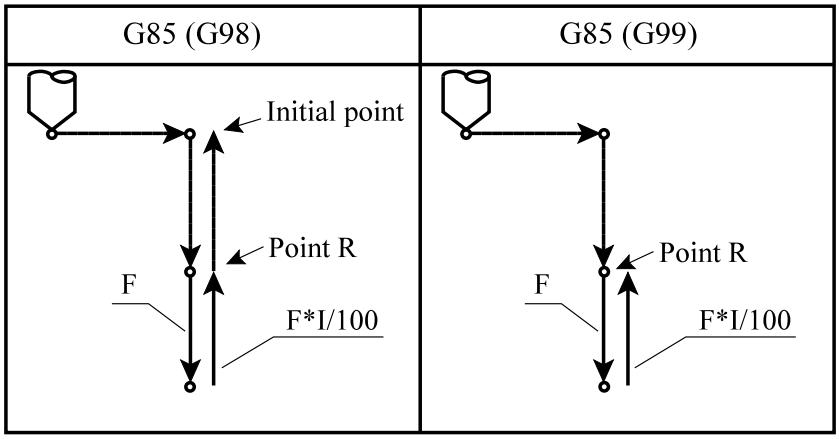


Fig. 18.1.11-1

The variables used in the cycle are the following:

G17 **G85** X_p__ Y_p__ Z_p__ R__ F__ I__ L__
G18 **G85** Z_p__ X_p__ Y_p__ R__ F__ I__ L__
G19 **G85** Y_p__ Z_p__ X_p__ R__ F__ I__ L__

The operations of the cycle are the following:

Operation 1: Positioning in the selected plane at rapid traverse rate
Operation 2: –
Operation 3: Rapid traverse to the point R (point of approaching)
Operation 4: –
Operation 5: Drilling to the bottom point at the feed rate F
Operation 6: –
Operation 7: Retracting to the point R at the feed rate F×I/100
Extraction override I (%)
If value is not given at the address I in the block, the extraction feed override in the operation 7 will be the value of the parameter N1502 Extraction Override in G85, G89 .The feed override button also produces effect on speed calculated in such a way, so the value of feed will be calculated according to the following formula:

Fextraction=Fprogrammed×Feed override×Extraction Override in G85, G89/100

Operation 8: –
Operation 9: In the case of G98, retracting to the initial point at rapid traverse rate
Operation 10: –

18.1.12 Boring Cycle with Retraction with Spindle in Standstill (G86)

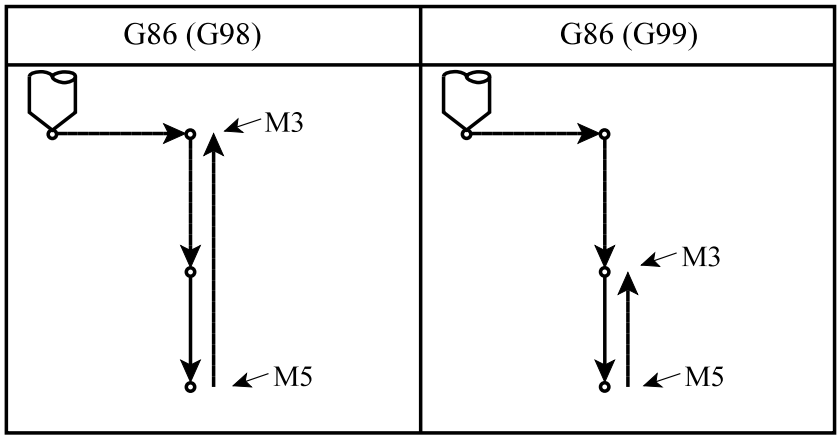


Fig. 18.1.12-1

The variables used in the cycle are the following:

G17 **G86** X_p__ Y_p__ Z_p__ R__ F__ L__
G18 **G86** Z_p__ X_p__ Y_p__ R__ F__ L__
G19 **G86** Y_p__ Z_p__ X_p__ R__ F__ L__

During starting the cycle, the direction of rotation M3 has to be specified for the spindle.
The operations of the cycle are the following:

Operation 1: Positioning in the selected plane at rapid traverse rate
Operation 2: –
Operation 3: Rapid traverse to the point R (point of approaching)
Operation 4: –
Operation 5: Boring to the bottom point at the feed rate F
Operation 6: Stopping the spindle: M5
Operation 7: In the case of G99: Retracting to the point R at rapid traverse rate
Operation 8: In the case of G99: Restarting the spindle in the direction M3
Operation 9: In the case of G98: Retracting to the initial point at rapid traverse rate
Operation 10: In the case of G98: Restarting the spindle in the direction M3

18.1.13 Manual Control/Back Boring Cycle (G87)

The cycle is executed by the control in two different ways.

A. Drilling cycle with manual control at the bottom point
In the case, if ***spindle orientation is not built in the machine tool***, i.e. the bit #1 ORI of the parameter N0607 Spindle Config is 0, the the control will operate according to the case A.

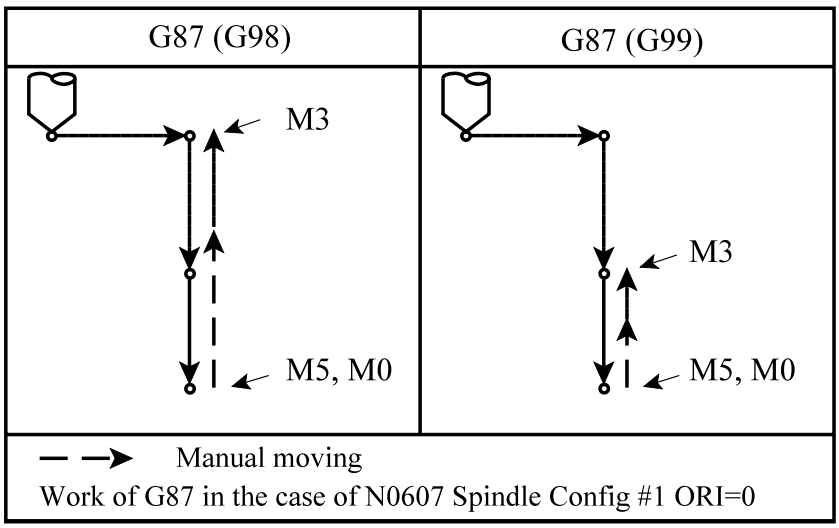


Fig. 18.1.13-1

The variables used in the cycle are the following:

G17 **G87** X_p__ Y_p__ Z_p__ R__ F__ L__
G18 **G87** Z_p__ X_p__ Y_p__ R__ F__ L__
G19 **G87** Y_p__ Z_p__ X_p__ R__ F__ L__

During starting the cycle, the direction of rotation M3 has to be specified for the spindle.
The operations of the cycle are the following:

Operation 1: Positioning in the selected plane at rapid traverse rate
Operation 2: –
Operation 3: Rapid traverse to the point R (point of approaching)
Operation 4: –
Operation 5: Boring to the bottom point at the feed rate F
Operation 6: Stopping the spindle: M5
The control gets to the state STOP (M0), from where, having transferred to one of the manual modes (JOG, INCREMENTAL JOG, HANDWHEEL), the operator can operate the machine manually, i.e. he can shift the tool tip from the surface of the hole and extract the tool from the hole. Then, having returned to the AUTO mode, machining can be continued by start.
Operation 7: In the case of G99: After START, retracting to the point R at rapid traverse rate
Operation 8: In the case of G99: Restarting the spindle in the direction M3
Operation 9: In the case of G98: After START, retracting to the initial point at rapid traverse rate
Operation 10: In the case of G98: Restarting the spindle in the direction M3

B. Back boring with automatic shifting the tool

In the case, if ***spindle orientation is built in the machine tool***, i.e. the bit #1 ORI of the parameter N0607 Spindle Config is 1, the the control will operate according to the case B.

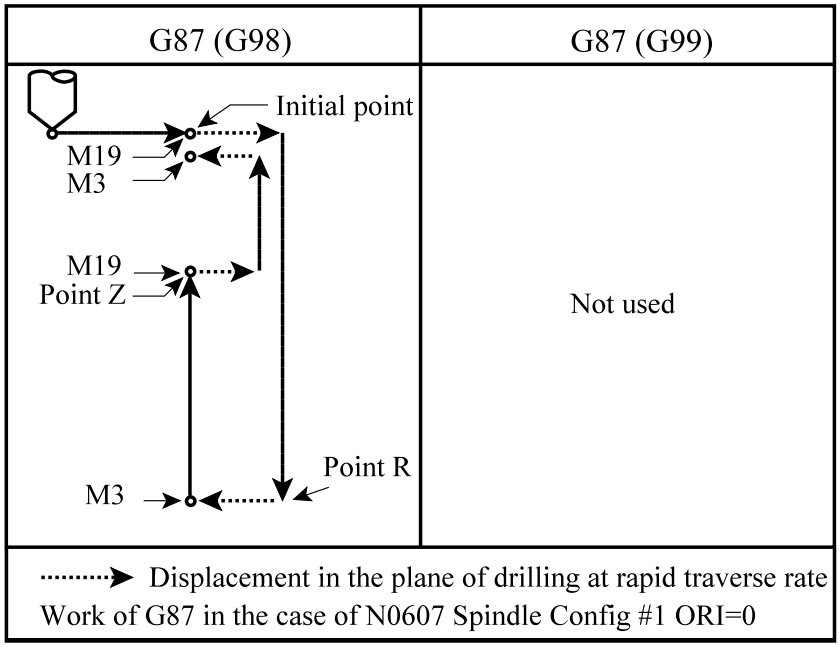


Fig. 18.1.13-2

The variables used in the cycle are the following:

G17 **G87** X_p__ Y_p__ I__ J__ Z_p__ R__ F__ L__
G18 **G87** Z_p__ X_p__ K__ I__ Y_p__ R__ F__ L__
G19 **G87** Y_p__ Z_p__ J__ K__ X_p__ R__ F__ L__

During starting the cycle, the direction of rotation M3 has to be specified for the spindle.
The operations of the cycle are the following:

Operation 1: Positioning in the selected plane at rapid traverse rate
Operation 2: Orientating the spindle.
In the selected plane, tool shift with the values specified at the addresses I, J and K, at rapid traverse rate
Operation 3: Rapid traverse to the point R (point of approaching)
Operation 4: In the selected plane, tool reset with the values specified at the addresses I, J and K, at rapid traverse rate.
Restarting the spindle in the direction M3
Operation 5: Boring to the bottom point at the feed rate F
Operation 6: Orientating the spindle: M19.
In the selected plane, tool shift with the values specified at the addresses I, J and K, at rapid traverse rate
Operation 7: –
Operation 8: –
Operation 9: Retracting to the initial point at rapid traverse rate
Operation 10: In the selected plane, tool reset with the values specified at the addresses I, J and K, at rapid traverse rate.
Restarting the spindle in the direction M3

It follows from the nature of the cycle, that, in contrast with the foregoing, ***the point of approaching, i.e. the point R is located lower than the bottom point***. This has to be taken into account in programming the addresses of the drill axis and R
Since spindle orientation and tool shift specified at the addresses I, J and K are executed by the cycle prior to the boring, the tool break can be avoided during entering the hole.

18.1.14 Boring Cycle with Dwell and Manual Operation at the Bottom (G88)

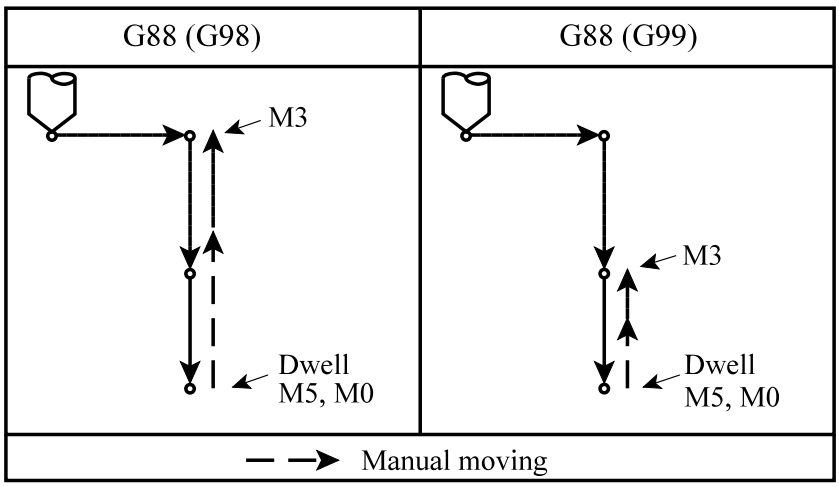


Fig. 18.1.14-1

The variables used in the cycle are the following:

G17 **G88** X_p__ Y_p__ Z_p__ R__ P__ F__ L__
G18 **G88** Z_p__ X_p__ Y_p__ R__ P__ F__ L__
G19 **G88** Y_p__ Z_p__ X_p__ R__ P__ F__ L__

During starting the cycle, the direction of rotation M3 has to be specified for the spindle.
The operations of the cycle are the following:

Operation 1: Positioning in the selected plane at rapid traverse rate
Operation 2: –
Operation 3: Rapid traverse to the point R (point of approaching)
Operation 4: –
Operation 5: Boring to the bottom point at the feed rate F
Operation 6: Dwell for the value specified at the address P
Stopping the spindle: M5
The control gets to the state STOP (M0), from where, having transferred to one of the manual modes (JOG, INCREMENTAL JOG, HANDWHEEL), the operator can operate the machine manually, i.e. he can shift the tool tip from the surface of the hole and extract the tool from the hole. Then, having returned to the AUTO mode, machining can be continued by start.
Operation 7: In the case of G99: After START, retracting to the point R at rapid traverse rate
Operation 8: In the case of G99: Restarting the spindle in the direction M3
Operation 9: In the case of G98: After START, retracting to the initial point at rapid traverse rate
Operation 10: In the case of G98: Restarting the spindle in the direction M3

The cycle is the same as the case A of the G87, but there is dwell prior to stopping the spindle.

18.1.15 Boring Cycle with Dwell and with Retraction at Feed Rate (G89)

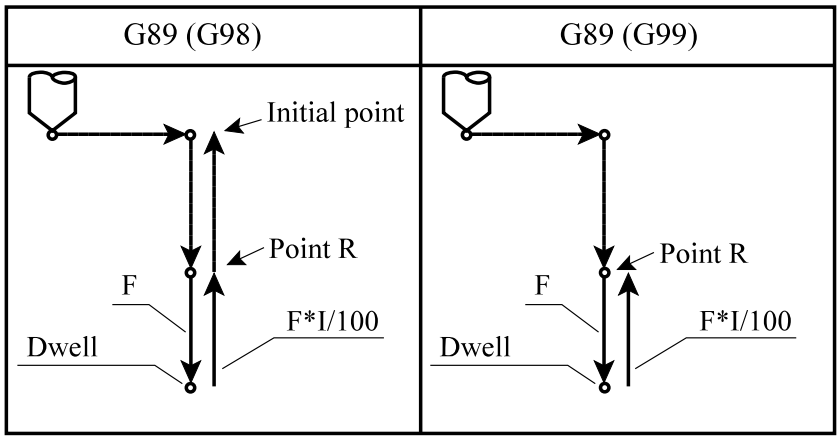


Fig. 18.1.15-1

The variables used in the cycle are the following:

G17 **G89** X_p__ Y_p__ Z_p__ R__ P__ F__ I__ L__
G18 **G89** Z_p__ X_p__ Y_p__ R__ P__ F__ I__ L__
G19 **G89** Y_p__ Z_p__ X_p__ R__ P__ F__ I__ L__

A ciklus műveletei:
The operations of the cycle are the following:

Operation 1: Positioning in the selected plane at rapid traverse rate
Operation 2: –
Operation 3: Rapid traverse to the point R (point of approaching)
Operation 4: –
Operation 5: Boring to the bottom point at the feed rate F
Operation 6: Dwell for the value specified at the address P
Operation 7: Retracting to the point R at the feed rate F×I/100
Extraction override I (%)
If value is not given at the address I in the block, the extraction feed override in the operation 7 will be the value of the parameter N1502 Extraction Override in G85, G89 .The feed override button also produces effect on speed calculated in such a way, so the value of feed will be calculated according to the following formula:

Fextraction=Fprogrammed×Feed override×Extraction Override in G85, G89/100

Operation 8: –
Operation 9: In the case of G98: Retracting to the initial point at rapid traverse rate
Operation 10: –

The cycle is equal to the G85, excluding dwell.

18.2 Remarks on the Use of the Drilling Cycles

– In cycle mode, the drilling cycle will be executed if a block without code G contains any of the following addresses:

X_p, Y_p, Z_p or q

where q is an optional axis, but not a drill axis.

Otherwise, the drilling cycle will not be executed.

– In cycle mode, if dwell block G04 P is programmed, the instruction will be executed according to the P, but the cycle variable related to dwell ***will not be deleted and rewritten***.

– The value of I, J, K, Q, E, Phas to be given in the blocks where drilling also occurs, otherwise the values ***will not be stored***.

An example to illustrate the foregoing:

```
G81 X_ Y_ Z_ R_ F  (the drilling cycle will be executed)
    X              (the drilling cycle will be executed)
    F_             (the drilling cycle will not be
                   executed, the F will be rewritten)
    M S_           the drilling cycle will not be
                   executed, the codes M and S will be
                   executed)
 G4 P_             the drilling cycle will not be
                   executed, the dwell will be executed,
                   the dwell variable of the cycle will
                   not be rewritten)
    I Q            (the drilling cycle will not be
                   executed, the programmed values will
                   not be recorded as cycle variables)
```

– If a function is also programmed together with the drilling cycle, the function will be executed at the end of the first operation, after completion of positioning. If L is also programmed in the cycle, the function will be executed only in the first pass.

– The repetition number L is not modal.

– In the block by block mode, the control stops after the operations 1, 3 and 10 within the cycle.

– The button STOP does not produces effect in the operations 5, 6 and 7 of the cycles G74 and G84. If the button STOP is pushed during these operations, the control will continue its functioning, and it can be stopped at the end of the operation 7.

– In the operations 5, 6 and 7 of the cycles G84.1 and G84, the feed override and the spindle override is always 100% independently of the position of the switch.

– If G43, G44, G49 are programmed in the cycle block or a new value H is specified, the length compensation will be taken into account in the operation 3, always along the drill axis.

19 Canned Cycles for Turning

The turning cycles work with the milling channel/control too. They work exactly as they do with the lathe control. The only difference is that the code of the cycles ends with the number 7 in the milling channel. Detailed description of the cycles can be found in the ‘NCT® 3xxT Control for Lathes Programmer's Manual’.

19.1 Single Cycles

The single cycles are the G77.7 longitudinal turning cycle, the G78.7 simple thread turning cycle and the G79.7 face turning cycle.

19.1.1 Longitudinal Turning Cycle (G77.7)

The G77.7 longitudinal turning cycle works exactly in the same way as the G77 longitudinal turning cycle described in the ‘NCT® 3xxT Control for Lathes Programmer's Manual’ does.

19.1.2 Simple Thread Turning Cycle (G78.7)

The G78.7 simple thread turning cycle works exactly in the same way as the G78 Simple thread turning cycle described in the ‘NCT® 3xxT Control for Lathes Programmer's Manual’ does.

19.1.3 Face Turning Cycle (G79.7)

The G79.7 face turning cycle works exactly in the same way as the G79 face turning cycle described in the ‘NCT® 3xxT Control for Lathes Programmer's Manual’ does.

19.2 Multiple Repetitive Cycles

The multiple repetitive cycles simplify the writing of the part program. For example, the contour of the part machined to size has to be described for finishing. This contour, at the same time, defines basis of the cycles performing the roughing of the part (G71.7, G72.7, G73.7). In addition to the roughing cycles, a finishing cycle (G70.7), a threading cycle (76.7) and two groowing cycles (G74.7, G75.7) are also available.

19.2.1 Roughing Cycle (G71.7)

The G71.7 roughing cycle works exactly in the same way as the G71 roughing cycle described in the ‘NCT® 3xxT Control for Lathes Programmer's Manual’ does.

19.2.2 Face Roughing Cycle (G72.7)

The G72.7 face roughing cycle works exactly in the same way as the G72 face roughing cycle described in the ‘NCT® 3xxT Control for Lathes Programmer's Manual’ does.

19.2.3 Pattern Repeating Cycle (G73.7)

The G73.7 pattern repeating cycle works exactly in the same way as the G73 pattern repeating cycle described in the 'NCT® 3xxT Control for Lathes Programmer's Manual' does.

19.2.4 Finishing Cycle (G70.7)

The G70.7 finishing cycle works exactly in the same way as the G70 finishing cycle described in the 'NCT® 3xxT Control for Lathes Programmer's Manual' does.

19.2.5 Face Grooving Cycle (G74.7)

The G74.7 face grooving cycle works exactly in the same way as the G74 face grooving cycle described in the 'NCT® 3xxT Control for Lathes Programmer's Manual' does.

19.2.6 Grooving Cycle (G75.7)

The G75.7 grooving cycle works exactly in the same way as the G75 grooving cycle described in the 'NCT® 3xxT Control for Lathes Programmer's Manual' does.

19.2.7 Multiple Threading Cycle (G76.7)

The G76.7 multiple threading cycle works exactly in the same way as the G76 multiple threading cycle described in the 'NCT® 3xxT Control for Lathes Programmer's Manual' does.

An example

```
...
N10 G10.9 Y1                                (Diameter programming
                                            Y)
N20 G43.7                                   (Lathe compensation)
N30 T1 M6                                   (Roughing tool T1)
N40 G95 T2
N50 G19 G54 G0 X0 Y100 Z100 H1 D1           (Plane Y-Z)
N60 G92 S3000
N70 G96 S100 P2 M3                          (Constant cutting speed
                                            for Y)
N80 G0 Y4 Z80
N90 G72.7 V5 R1
N100 G72.7 P110 Q170 W0.5 V1 M3 S200 F1     (Face roughing)
N110 G0 G41 Y-80 F1.5
N120 G1 Z50
N130 Y-60 Z40
N140 Z30 ,R5
N150 G2 Y0 Z0 R30
N160 G40 G1 Z-1
N170 Y4
N180 G0 Z80
N190 Y-56
N200 G71.7 W5 R1
N210 G71.7 P220 Q310 W1 V2 S250 F1.2        (roughing)
N220 G42 G0 Z50
```

```
N230 G0 Y-76
N240 G1 Y-80 F2
N250 Z55 ,C2
N260 Y-130
N270 G2 Z65 Y-150 R10
N280 G1 Z70 ,R2
N290 Y-170 ,C5
N300 Z77.5
N310 G40 Z80
N320 G0 Z100 Y100
N330 M6                              (Finishing tool T2 on)
N340 G0 X0 Z80 Y4 H2 D2
N350 G70.7 P110 Q170                 (Finishing cycle 1)
N360 G0 Z80
N370 Y-56
N380 G70.7 P220 Q310                 (Finishing cycle 2)
N390 G0 X0 Y100 Z100
N400 G10.9 Y0                        (Radius programming Y)
N410 G43                             (Milling machine
                                     compensation)
...                                  (Milling)
```

20 Functions to Control Axes

20.1 Electronic Gear Box. Gear Hobbing (G81.8)

Gear hobbing (Pfauter milling) can be carried out in an electronic way using so-called electronic gear box. The ***electronic gear box*** establishes linkage of a programmable ratio between the tool gripped in the ***spindle***, i.e. the ***master axis***, and the workpiece (the gear) clamped on a ***rotary table***, i.e. the ***slave axis***. Synchronization of these two motions is realized by the control in an electronic way.

An electronic gear box can be established between a spindle defined in the parameter N1801 EGB Master and a a rotary table defined in the parameter N1802 EGB Slave. The function G81.8 establishes linkage between these two axes.

Synchronization can be started by the function

G81.8 T_ L_ R_ Q_ P_

In the block the following parameters can be given:

T: ***number of teeth***. The range of specifiable values: 1 ... 1000 (min). ***It is always obligatory to specify it***.

L: ***number of starts and lead direction of the hob***. The specifiable values: 21 ... +21. If the sign of the L is positive, the lead direction of the hob will be right; if it is negative, the lead direction will be left.

The sign of the L determines the workpiece rotation direction during synchronization. If L+ (right-handed hob), the rotation direction will be positive and if L− (left-handed hob), the rotation direction will be negative.

If the value of the L is not specified, the value L1 will be taken into account by the control.

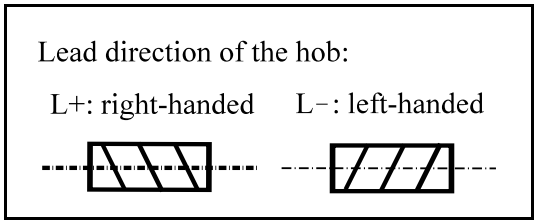


Fig. 20.1-1

R: ***the way of linking the axes***:

R=1 request for synchronization in order that the zero pulses will run in phase;

R=2 request for synchronization and phase shift. The value of phase shift is determined by the instantaneous position of the workpiece (valid at the moment of issuing the instruction G81.8).

If programming the R is omitted, the value R1 will be taken into account by the control.

Q: ***module or diametral pitch***. It has to be given only in the case of machining helical gear.

In the case of metric data specification G21, it is interpreted as module (normal module). The specifiable values: 0,05...60 mm;

In the case of inch data specification G20, it is interpreted as diametral pitch. The specifiable values: 0,1 ... 508 1/inch.

P: ***the value of the helix angle of the tooth and the direction of the tooth profile.*** It is to be done only in the case of machining helical gear. The range of specifiable values: −90°...+90°.

If the sign of the P is negative, the direction of the tooth profile will be right; if it is positive, the direction of the tooth profile will be left.

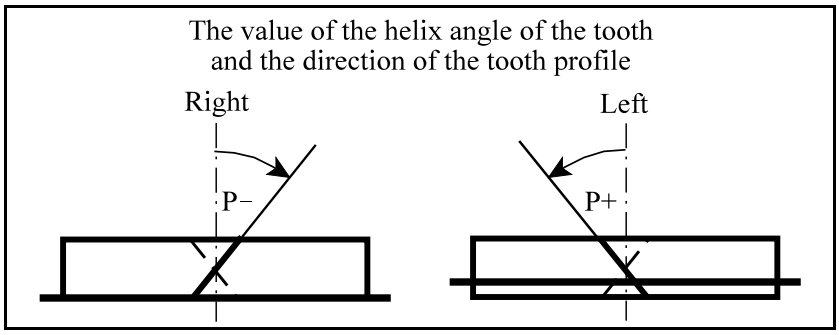


Fig. 20.1-2

When synchronization is being started, a valid reference point record on the slave axis is required. When the electronic gear box is on, the instruction G81.8 can be issued again:
In the case of issuing

G81.8 T_ L_ R_

the synchronization will occur again using the new parameters specified, while in the case of issuing

G81.8 Q_ P_

only the tooth helix angle compensation will be switched on, but the system will not be synchronized again.
If the instruction G81.8 Q_ P_ is issued in the off-state of synchronization G80.8, the control will indicate error.

In the on-state of the electronic gear box, ***displacement*** (phase shift) can be programmed on the ***workpiece axis*** (on the rotary table rotating the workpiece) without stopping synchronization.

Synchronization can be cancelled by the function

G80.8

Due to the instruction, the slave axis stops rotating, while rotation of the spindle (master axis) remains unchanged. After that, the slave axis can be used as normal control axis.

20.1.1 Feed per Minute in the State G81.8

In synchronized state (G81.8), it is the bit #1 FRS of the parameter N1800 EGB Contr where the user can set whether the ***feed per revolution*** (state G95) will be interpreted by the control for a revolution of the ***tool*** or the ***workpiece***, i.e. whether the value F will be taken into account directly from the pulse number of the spindle encoder or as the product of the pulse number and the ratio L/T.

20.1.2 Starting and Cancelling the Synchronization

Due to starting the synchronization,

– in the case of the R1, ***the control moves the zero pulse of the slave spindle to the zero pulse of the master spindle encoder*** (taking into account the shift specified in the parameter

N0906 Reference Shift in the case of the slave axis and in the parameter N0684 Spindle Grid Shift in the case of the master axis), or;

– in the case of the R2, the control shifts the zero pulses by the value of the pulse shift;
– the ***workpiece*** (slave) axis ***takes on the speed determined by the tool*** (master);
– then, the slave axis follows correctly the motion of the master axis, in accordance with the position.

Cancelling the synchronization, the motion of the workpiece stops.

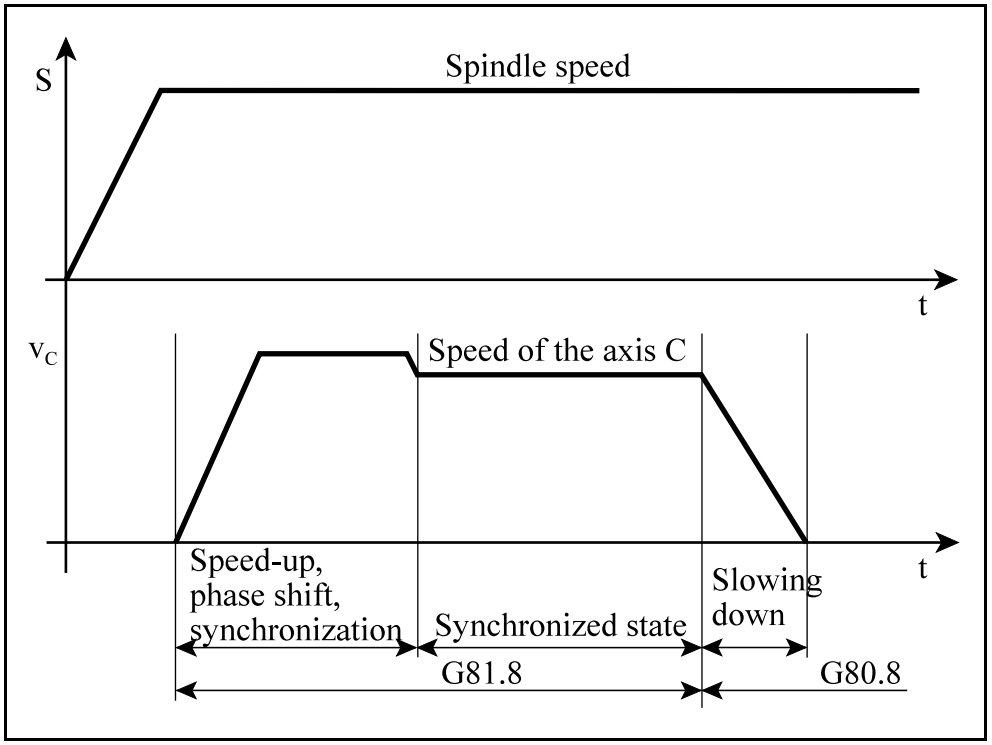


Fig. 20.1.2-1

During gear cutting, ***the maximum speed of the spindles*** is limited by ***the rapid traverse rate***. Generally, this rate is smaller than the maximum spindle speed. The workpiece (rotary table) speed of 360×S×T/L also cannot be higher than the rapid traverse value permitted for the axis.

Let the C be the slave axis.

Example of the zero pulses running in synchronism:

```
M3 S300              (starting the spindle)
G0 C90               (positioning the slave to 90°)
G81.8 T20 L2 R1      (starting the synchronization,
                     ratio=20/2)
...                  (machining, with zero pulses running in
                     synchronism)
G80.8                (cancelling the synchronization)
```

Example of phase shift:

```
M3 S300                (starting the spindle)
G0 C90                 (positioning the slave to 90°)
G81.8 T20 L2 R2        (starting the synchronization and phase
                       shift)
...                    (machining with phase shift of 90°)
G80.8                  (cancelling the synchronization)
```

Changing the synchronization ratio

In the case of machining compound gears (several gears on one shaft), it can be necessary to change ratio, i.e. to modify the ratio T/L in synchronized state. It can be realized by acceleration or deceleration to the new speed without stopping the workpiece, what is followed by setting the encoder of the workpiece axis to the required position:

```
G54 G95 G0 X400 Z-50 Y0
S300 M3
G81.8 T20 L2 R1        (starting the synchronization,
                       T/L=20/2)
G0 X-5                 (infeed)
G1 Z100 F1             (gear cutting)
G0 X100                (retracting)
G81.8 T16 L2           (modifying the ratio, T/L=16/2)
...
G80.8
```

Modifying the phase shift

In synchronized state, motion command can be issued to the slave axis. This instruction will modify the phase shift!

```
G54 G95 G0 X400 Z-50 Y0
S300 M3
G81.8 T20 L2 R1        (starting the synchronization )
G0 X-2                 (infeed)
G1 Z100 F1             (gear cutting)
G0 X100                (retracting)
G81.8 T16 L2 R1        (modifying the ratio, T/L=16/2)
G91 C15                (phase shift of 15°)
G90 Z-50
...
G80.8
```

20.1.3 Compensation of the Tooth Helix Angle

After starting the synchronization, if ***the gear is a helical gear***, ***the tilting of the workpiece will have to be compensated*** in accordance with the displacement of the axis (the tool) perpendicular to the workpiece in order that the tool runs always in the groove. The compensating displacements issued to the workpiece axis will modify both the instruction position and the phase shift position!
The axis to be used for compensation of the tooth helix angle can be defined in the parameter N1803 Helical Comp. Axis.

In the case of the following equations it is assumed that the axis c rotates the workpiece and the axis Z parallel with the rotating axis C rotates the tool. Compensation is performed according to the following function:

in the metric case of **G21**:

$$\Delta C = \frac{360^\circ \cdot \sin P}{\pi \cdot T \cdot Q} \cdot \Delta Z$$

$$k = \frac{360^\circ \cdot \sin P}{\pi \cdot T \cdot Q}$$

where P: helix angle
T: number of teeth
Q: module (normal module)
k: compensation coefficient [°/mm]

in the inch case of **G20**:

$$\Delta C_c = \frac{360^\circ \cdot Q \cdot \sin P}{\pi \cdot T} \cdot \Delta Z$$

$$k = \frac{360^\circ \cdot Q \cdot \sin P}{\pi \cdot T}$$

where P: helix angle
T: number of teeth
Q: diametral pitch
k: compensation coefficient [°/inch]

The ***direction*** (the sign) ***of compensation*** is determined by the ***motion direction of the mill*** (Z+ or Z−) and the ***direction of the tooth profile*** (P+/− left, right):

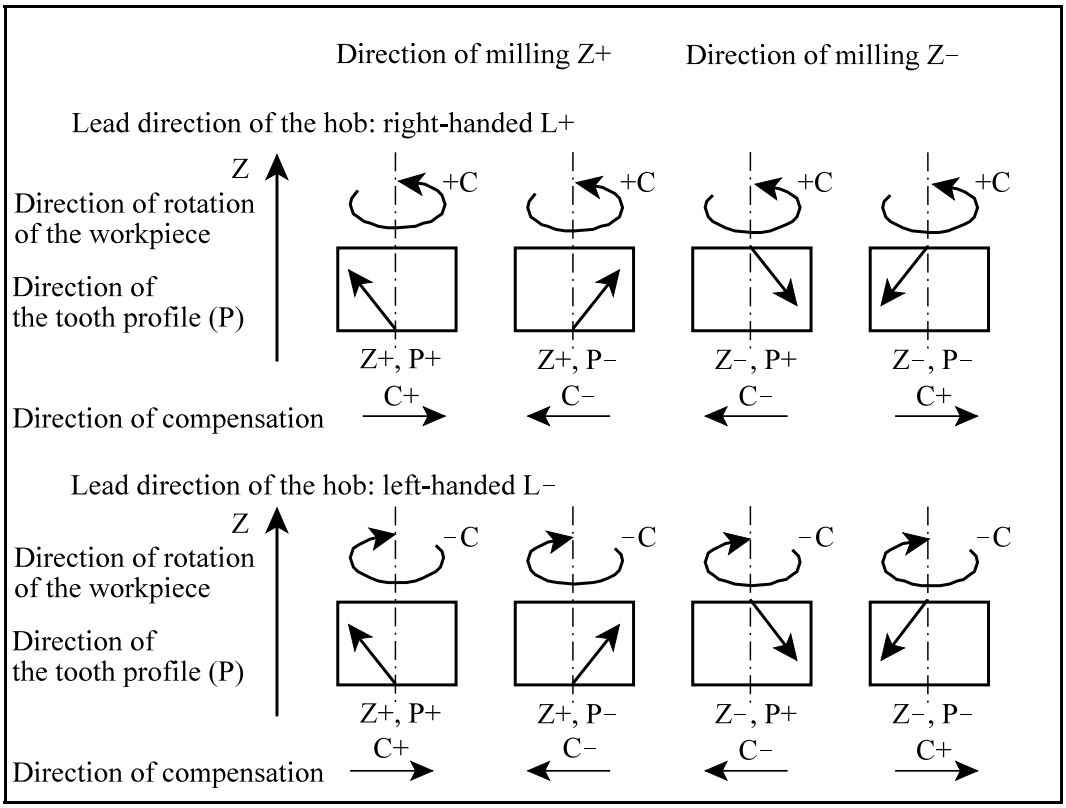


Fig. 20.1.3-1

Starting the compensation of the tooth helix angle and the synchronisation at the same time

Compensation of the tooth helix angle and synchronisation can be started simultaneously:

```
G54 G95 G0 X400 Z-50 Y0
S300 M3
G81.8 T20 L2 R1 Q5 P15    (starting the synchronisation)
G0 X-5                    (infeed)
G1 Z100 F1                (gear cutting with compensation of
                          tooth helix angle)
G0 X100                   (retracting)
...
G80.8
```

Starting the compensation of the tooth helix angle in synchronized state:

Compensation of the tooth helix angle can be started in synchronized state of the axes too, without resynchronizing them. In this case, programming the T and the L can be omitted:

```
G54 G95 G0 X400 Z-50 Y0
S300 M3
G81.8 T20 L2 R1     (starting the synchronisation,
                    T/L=20/2)
G0 X-2              (Infeed)
G1 Z100 F1          (gear cutting )
G0 X100             (retracting)
```

```
G81.8 Q5 P15          (starting the compensation of tooth
                      helix angle without resynchronization)
...
G80.8
```

20.1.4 Retracting the Gear Cutting Tool

In the synchronized state G81.8, it could be necessary to retract the tool from the teeth. Such cases are the following:
– emergency stop;
– errors generating emergency, e.g. servo error, spindle error etc.;
– operator's intervention, e.g. tilting the switch;
– reset.
Retraction of the tool from the teeth due to reset will occur only in the case if the value of the bit #0 RCE of the parameter N1800 EGB Contr is 1, i.e. the function G81.8 will be stopped in the case of reset. Otherwise (RCE=0), synchronization will not ceased.
On the machine, development of emergency has to be delayed by a few hundred main spindle in order that there will be time for retraction.
The amount, the axis and the speed of retraction are values given in parameter.
The ***amount of retraction*** (the ***incremental*** displacement) is given by the parameter N1804 Retr. Dist. for each axis, with correct sing. Those ***axes*** participate in retraction of the tool for which the value of the parameter N1804 Retr. Dist. is not 0.
The speed of retraction can be set in the parameter N1805 Retr. Feed. If the value of the parameter is 0, the speed of retraction will be the rapid traverse rate valid for the given axis.

20.2 Synchronous Control of Axes

In the course of machining a workpiece, it could be necessary to move two axes synchronously. The axes can be either in the same channel or in different channels.
The axis for which the data are given in the part program is called ***master*** axis.
The axis which moves in synchronism with the master axis is called ***slave*** axis**.** So long as an axis is synchronous slave axis, this axis cannot be moved either from program or manually.
Synchronous moving is initiated by the PLC program by execution of a code M, for example.

Parameters and PLC flags used in the case of synchronous moving

If two axes are to be linked together for synchronous moving, it will ***have to be specified in the parameter*** N2101 Synchronous Master ***belonging to the slave axis which axis is its master axis***.
The master axis can be either in the same channel or in a different channel. The master axis can have several slave axes. A slave axis can also be the master axis of another axis.
In the case of synchronous control of axes, it is only the master axis that can be programmed or for which manual motion command (jog, handwheel) can be issued.

Synchronous control is requested by the PLC setting the flag AP_SYNCR belonging to the slave axis to1. The PLC waits until the control acknowledges the request through the flag AN_SYNCRA. From that moment, displacements of the master axis will be received by the slave axis too.
This flag can ***usually be switched on or off by functions M.***
For example:

```
...
M41 (synchronous moving on)
...
M40 (synchronous moving off)
...
```

Hereafter, the code pair M40, M41 is used in this manual for switching synchronization on and off. In the case of a given machine, the code of ***the function and operation description should be asked the builder of the machine tool for.***

Moving in synchronism can only start strictly ***by the function of buffer emptying***.
If moving in synchronism occurs ***between two channels***, in the case of the other channel a code for waiting will have to be programmed:

Program of the channel 1

```
...
M502 P12
M41
M503 P12
...
```

Program of the channel 2

```
...
M502 P12
(waiting for synchrony
on)
M503 P12
...
```

Another condition of requesting for synchronous moving is the existence of valid reference point both on the master axis and on the slave axis.

In the case of moving in synchronism, ***the motion direction of the slave axis*** can be the same as that of the master axis, but it can be different too. The motion direction of the slave axis can be

set at the bit #0 MSY of the parameter N2102 Synchron Config which bit belongs to the slave axis. If the value of the bit of the parameter

=0: both the master axis and the slave axis move in the same direction;

=1: both the master axis and the slave axis move in opposite direction.

Example of moving the axes in synchronism

To illustrate moving the axes in synchronism, let us have the following machine. The table, i.e. the axis X moves together with the workpiece. There is a vertical column on both sides of the table, opposite each other; on each column there is a beam moving in two directions perpendicular to each other. The axes of the right column are Y and Z, and the axes of the left column are V and W.

The master axis of the slave axis V is the axis Y, and the master axis of the slave axis W is the axis Z.

The positive direction of the axis V is opposite to the positive direction of the axis Yin order that the coordinate system XVW will be right-handed, for this reason the parameter N2102 Synchron Config is #0 MSY=1.

Let the spindle with rotational axis parallel with the axis Z be S1, and the spindle with rotational axis parallel with the axis W be S2.

All five axes and both spindles can be moved and programmed freely.

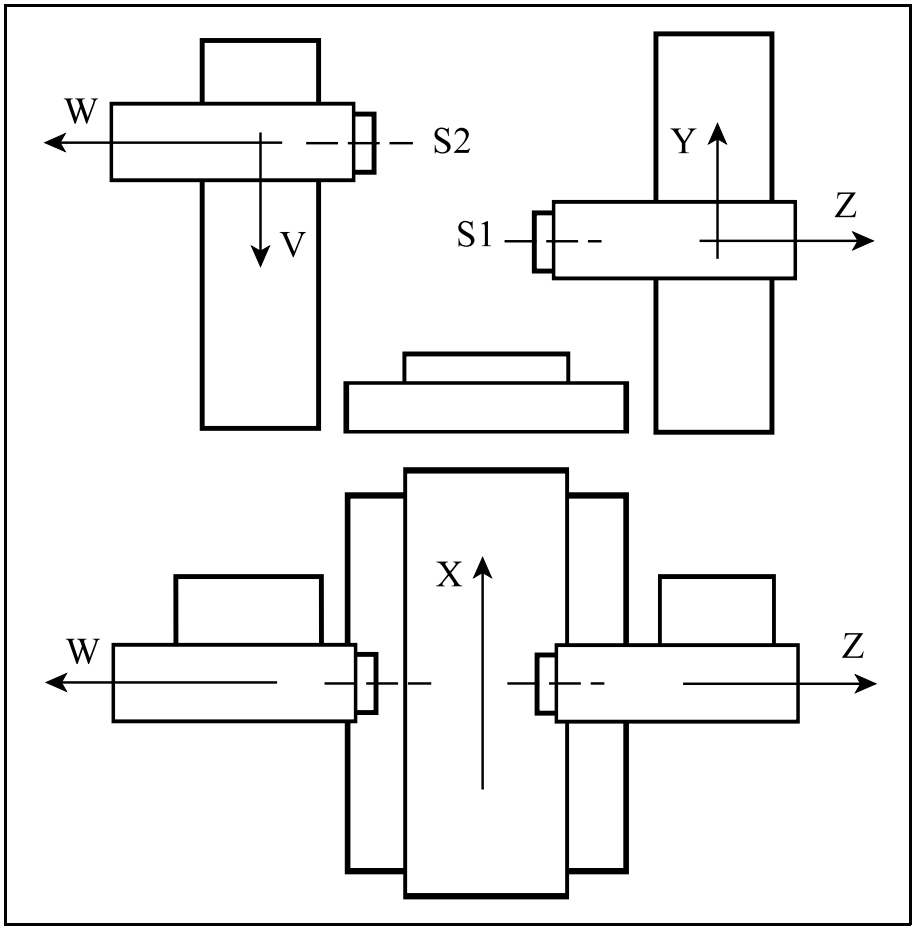


Fig. 20.2-1

Let us assume that

it is the code **M41** interlinking the axes Y-V and Z-W for synchronous moving at the same time,

and

it is the code **M40** cancelling the synchronous operation.

On the machine tool above, a workpiece having right and left sizes symmetrical to the axis X will be machined using synchronous moving.

For machining, the tool T1 being in the spindle S1 and the tool T101 being in the spindle S2 will have to be moved to the same position on the axes Y and Z, and on the axes V and W, respectively. Then, the synchronous cutting will follow. *Diameters of the two tools have to be equal!*

The part program is as follows:

```
...
T1 M6           (T1 into the spindle S1)
T101 M6         (T101 into the spindle S2)
...
G0 X0 Y0 V0     (positioning along the axes X Y V)
G43 Z10 H1      (calling the compensation and positioning
                along Z on the right side)
G43 W10 H101    (calling the compensation and positioning
                along W on the left side)
M41             (interlinking the axes Y-V and Z-W for
                synchronous operation)
G1 Z-5          (From this point the program is to be
                written using the addresses X, Y and Z until
                cancelling synchronization. Both sides move)
G41 G1 X100 D1  ( Diameters of the tools T1 and T101 are
                equal!)
G3 X0 Y100 R100
G1 Y0
...
M40             (cancelling the synchronous operation)
...
```

Parking the axes in the case of moving the axes in synchronism

It could be necessary that a program written with the use of moving in synchronism runs only on one side, either on the master side or on the slave side; and not to write a new program.

In this case, the side not intended to be moved has to be ‘parked’. Either the slave side or the master side can be parked. In such a case too, the control calculates appropriate displacements both for the master axis and for the slave axis, but it does not issue motion command for the parking side.

Parking an axis participating in synchronous moving is requested by the PLC setting the flag of request for parking AP_PARKR to1. The control acknowledges the acknowledgment of the request for parking by setting the acknowledging flag AN_PARKA to1.

Parking can be initiated ***from push-button or from function M***. In the case of a given machine, ***description of parking should be asked the builder of the machine tool for.***

20.3 Interchanging the Axes

In the control, it is possible to interchange the axes. The axes may be in the same channel or in the different ones, too.
Master axis is the axis with which the slave axis is interchanged.
Slave axis is the axis that initiate the interchange.
It is the PLC program that initiate interchanging the axis by execution of a code M, for example.

Parameters and PLC flags applied to interchanging the axes

If two axes should be interchanged, it will be necessary to specify ***in the parameter*** N2104 Composit Axis ***belonging to the slave axis, which axis is its master axis***.
The master axis may be in the same channel or in a different one, too.
In the case of ***interchanging the axis, both the master axis and the slave axis can be programmed***, or, manual motion command (jog, handwheel) can be issued.

Interchanging the axes is requested by the PLC by setting the flag AP_MIXR belonging to the slave axis to 1. The PLC waits until the NC acknowledges the request through the flag AN_MIXA. From that moment the two axes are interchanged.
In general, this flag ***can be switched*** on or off ***by a functions M***.

For example:

```
...
M42 (to switch the interchanging the axes on)
...
M40 (to switch the interchanging the axes off)
...
```

In the case of a given machine, please ***ask the builder of the machine for the code and operation of the function.***

Interchanging the axes can only start strictly by a ***function of buffer emptying***.
If interchanging the axes occurs ***between two channels***, waiting code will have to be programmed for the other channel:

The program for the channel 1

```
...
M502 P12
M42 (interchanging the axes on)
M503 P12
...
M504 P12
M40 (interchanging the axes off)
M505 P12
...
```

The program for the channel 2

```
...
M502 P12
(waiting for inter-change)
M503 P12
...
M504 P12
(Waiting for cancel of interchange)
M505 P12
...
```

After interchanging the axes, the ***motion direction of the slave axis*** may be the same as its original direction, but it may be opposite to it. The motion direction of the slave axis can be set at the bit #0 MMI of the parameter N2105 Composit Config which bit belongs to the slave axis. If the value of the bit of the parameter:

=0: the slave axis will move in the original direction, after the interchanging the axes

=1: the slave axis will move in the direction opposite to the original one, after the interchanging the axes.

After interchanging the axes, the ***motion direction of the master axis*** may be the same as its original direction, but it may be opposite ti it. The motion direction of the master axis can also be set at the bit #0 MMI of the parameter N2105 Composit Config which bit belongs to the master axis. If the value of the bit of the parameter:

=0: the master axis will move in the original direction, after the interchanging the axes

=1: the master axis will move in the direction opposite to the original one, after the interchanging the axes.

After interchanging the axes, ***the axes will have their original zero-point offsets and length compensations***. If, for example, the axes Y and Z are interchanged, and length compensation for the axis Z is entered, the length compensation will appear for the axis Y. So, it is advisable to cancel the length compensation before interchanging the axes, and, to replace a new zero-point offset measured after the axis change and a new length compensation.

20.4 Changing the Axis Direction

In the control, it is possible to change the movement direction of an axis, from the PLC program: the positive direction command will cause a negative movement and vice versa.

It is the PLC that requests changing the axis direction by setting the flag AP_MIRR belonging to the given axis to 1. The PLC waits until the NC acknowledges the request through the flag AN_MIRA. From then on, the movement direction of the axis will be opposite to the direction set in parameter.

In general, this flag ***can be switched*** on or off ***by the functions M***.
In the case of a given machine, please ***ask the builder of the machine for the code and operation of the function.***
Changing the axis direction can only start strictly by a ***function of buffer emptying***.

Changing the axis direction does not affect the direction of movement and the positions of the positionings (G53) programmed in the machine coordinate system and of the moving to the zero point (G28, G30 P).

20.5 Interchanging the Vertical and Horizontal Axes and Changing the Axis Direction

Interchanging the axes and changing the axis direction can, for example, be applied on those machines the head of which can be adjusted in both vertical and horizontal direction.
Let us assume that designation and direction of the machine axes X, Y and Z have been set to the ***vertical*** head position. In this case, the ***tool points in the direction of the axis Z***, ***the part programs*** are written ***in the plane G17***, and ***the length compensation*** is taken into account ***on the axis Z***.
Then, if the head is moved to the ***horizontal*** position, ***the tool will point in the direction of the axis Y***, therefore ***the programs*** will have to be written ***in the plane G18***, and ***the length compensation*** will have to be taken into account ***on the axis Y***.

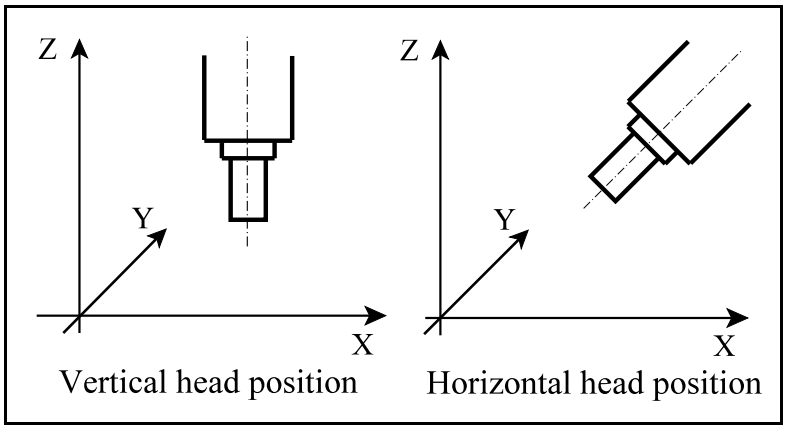


Fig. 20.5-1

In order to prevent the programs from adapting to the current head position when taking into account plane selection and length compensation, ***the axes Y and Z*** set originally in the parameter ***have to be interchanged***. Changing the Y and Z axes has the consequence that ***the direction of the axis X should be reversed*** in order that the coordinate system remains right-handed, i.e. the circular interpolation, the radius compensation etc. functions in usual way.

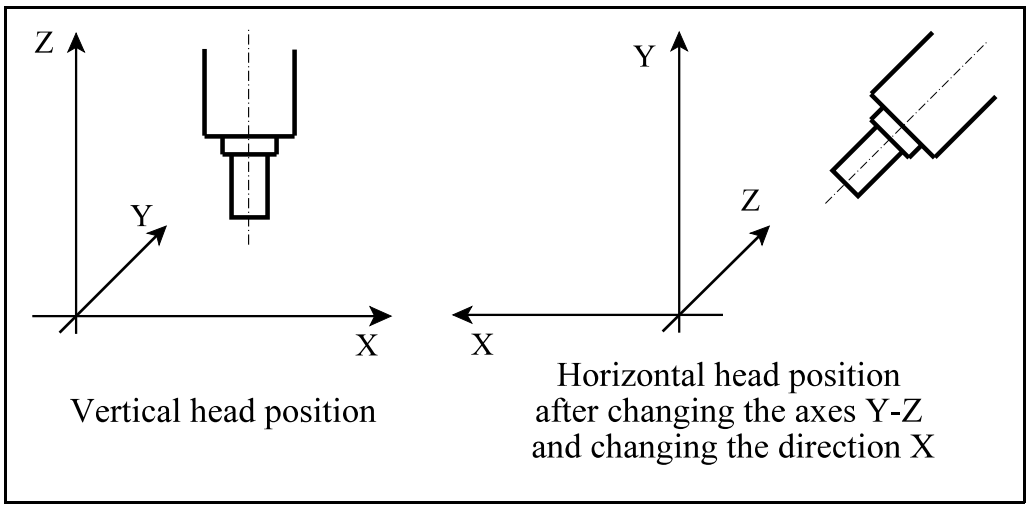


Fig. 20.5-2

In general, changing the head can be ***switched on or off by functions M***, but it is executed always by the PLC program.

For example:

```
...
G54
...
G49 (cancelling the compensation)
M42 (changing Y-Z and the direction of X On)
G55... (new workpiece zero point)
G43 Z H... (new compensation)
...
G49 (cancelling the compensation)
M40 (changing Y-Z and the direction of X Off)
```

```
G54... (new workpiece zero point)
G43 Z H... (new compensation)
...
```

In the case of a given machine, please ***ask the builder of the machine for the code and operation of the function.***

If interchanging the heads is started from program by a function, that function has to strictly be a ***function of buffer emptying***.

Cancelling the length compensation before interchange and calling for new workpiece zero point and new compensation sould not be forgotten (see the chapter 20.3 Interchanging the axes on page 260).

20.6 Co-axes Control

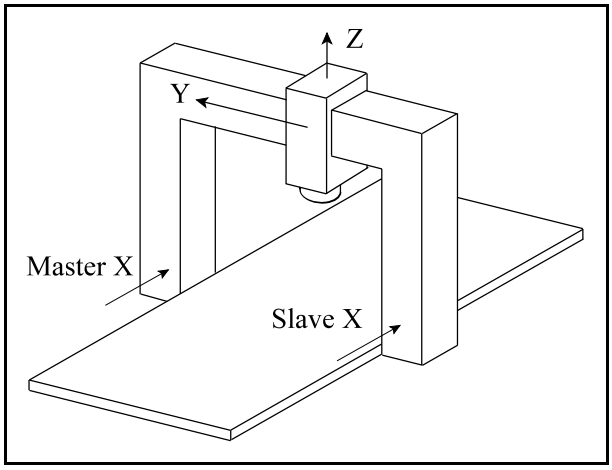


Fig. 20.6-1

If moving one axis is realized by two servo motors and two servo drives, and both servo motors have their separate encoders (rotary encoder or scale)***, it will be the case of co-axes.*** Such cases take place on large portal or gantry-type machine tools.
As it is illustrated in this picture, the axis X is driven by a motor on both sides. One side is the master axis, the other side is the slave axis . Each motion command arriving at the axis X will be received by the servo loops of the master axis and the slave axis.
When co-axes are applied, ***the mechanical joint between the master and slave axes is not rigid*** in order to avoid mechanical jamming. ***The rigid joint is realized by control*** in such a way that if the slave axis lags behind its master axis, the slave axis will try to control itself to the master axis. If the position deviation exceeds a value specified in parameter, the control will send the error message '1008 # Axis synchron error', delete record on positioning to the reference point and shut down the machine. The # is the address of the given axis.

Managing the co-axes should not be confused with moving the axes in synchronism.
Starting and cancelling of the moving the axes in synchronism can be done, but these actions cannot be done in the case of managing the co-axes, that remains started as long as the control is on.
When moving in synchronism is disconnected, a motion command independent of that of the master axis can also be issued to the slave axis, but in the case of the co-axes the slave axis cannot be moved separately.
Generally, in the case of co-axes, the position of the slave axis is not displayed.
In the case of co-axes, in the course of positioning to the reference point, after that the master axis positioned to the reference point, the slave axis positions to the reference point automatically.
In the case of co-axes, the machine position of the slave axis and the machine position of the master axis is the same; the machine position of the slave axis will be controlled by the control to that of the master axis. In the case of synchronous moving, the machine positions are different, there is no control.

When two ayes are to be interconnected as co-axes, ***it has to be defined in the parameter*** N2101 Synchronous Master ***belonging to the slave axis which axis is its master axis***. In addition to this, ***it will be indicated by bit setting #4 PSN=1 of the parameter #4 PSN=1 that this axis is co-axis of the master axis***.

Position compensation between the master axis and the slave axis can be set in the parameter N0200 Reference Position1 belonging to the slave axis. This parameter will be taken into account by the control only after positioning to the reference point. Using the example above, if the intention is that the beam (the axis Y) will be perpendicular to the axis X, the parameter N0200 Reference Position1 of the slave axis of the X will have to be rewritten for the parameter of the master axis.

Controlling the machine position of the slave axis for that of the master axis will start only after positioning to the reference point and position compensation.

20.7 Tandem Control

If moving one axis is realized by two servo motors and two servo drives, but there is only one encoder (rotary encoder or scale) ***for this axis, it will be the case of tandem control.***
Tandem control is used in the case, when one axis can only be driven by two servo motors because of the high torque requirement, for example. Using the tandem control for two motors, the gear backlash can also be compensated if the two motors drive the same gear rack, for example. In this case, position control is executed by the motor of the master axis, the slave axis carries out only torque control.

20.8 Managing Non-perpendicular Axes

If the movement of one axis on a machine is not perpendicular to the other, but ***makes an angle with respect to the right angle***, managing angular or slant axis is to be applied. See description of this function in the Control for Lathes Programmer's Manual.

21 Measurement Functions

21.1 Skip Function (G31)

The instruction

G31 v (P) (F)

starts ***the motion*** to the point of coordinate ***v*** using ***linear interpolation***. The motion continues until an ***external skip signal*** (e.g. the touch probe signal) arrives or ***the endpoint position of the coordinates v*** is reached by the control. After arriving of the skip signal or at the programmed endpoint of the block the control will decelerate and stop.

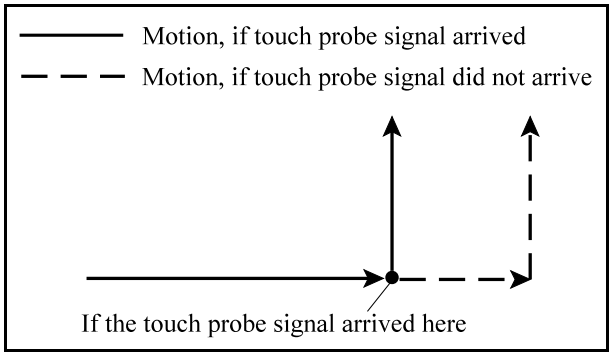


Fig. 21.1-1

The touch probe signal among the 8 ones inputable to the control which is to be taken into account in the course of the motion, can be specified ***at the address P***:

P1: using the touch probe signal 1;
P2: using the touch probe signal 2;
...
P8: using the touch probe signal 8.

Filling the address P is not obligatory; if the address P is not filled, the touch probe signal 1 will be taken into account by the control.

The function G31 has to always be used in the state ***G94 (feed per minute)***. During the motion, ***the feed F*** will be:

– a specified on modal value, if the bit #0 SKF of the parameter N3001 G31 Config is 0;
– a value taken on from the parameter N0311 G31 Feed, if the bit #0 SKF of the parameter N3001 G31 Config is 1.

The instruction ***G31 is not a modal one,*** it is valid only in the block in which is programmed.

At the moment the external signal arrives, ***the positions of the axes will be stored in the following macro variables***:

#5061 or #100151 or #_ABSKP[1]: position of the axis 1;
#5062 or #100152 or #_ABSKP[2]: position of the axis 2.
...

The positions stored in the macro variable above will be the following:

– the position taken ***at the moment the signal arrives, if the external signal arrived;***
– the position of the programmed ***endpoint*** of the block G31, if the external signal ***did not arrive***.

The position data will be stored

– always ***in the coordinate system of the actual workpiece***;
– ***without taking the actual length compensation*** (G43, G44) ***into account***.

After the external signal arrives, motion will stop with deceleration. At this time, the endpoint position of the block G31deviates to a small extent from the positions stored in the variables #5061... at the moment the signal arrives, according to the feed used in the block. The endpoint

positions of the block can be accessed in the variables #5001... . The succeeding motion block will be valid from these endpoint positions.

Execution of the block G31is possible only in the states ***G15, G40, G50, G50.1 G69, G94***. Otherwise, the error message '2055 Skip function in Gnn state' will be induced.

The value given at the coordinates v can be both ***incremental and absolute.*** If coordinate specification of the succeeding block is incremental, displacement will be calculated by the control from that point of the block G31where the motion stopped in the previous block.

For example:

```
N1 G31 G91 X100
N2 X30 Y50
```

In the block N1, an incremental motion in the direction X is started by the control. If, after arriving of the external signal, the control stops at the point with the coordinate of X=86.7, the following incremental motions from that point will be executed in the block N2: 30 in the direction X and 50 in the direction Y.

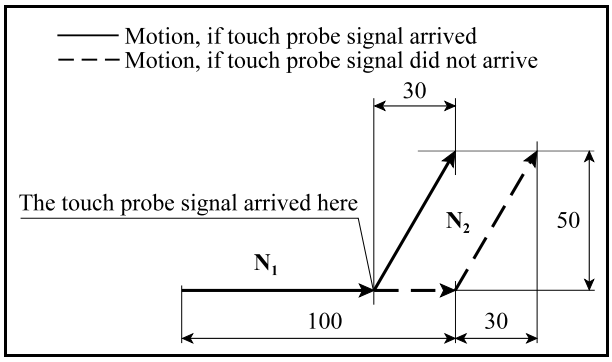


Fig. 21.1-2

If absolute data specification is programmed, the motion will be as follows:

```
N1 G31 G90 X200
N2 X300 Y100
```

The block N1 starts a motion in the direction X to the point with the coordinate of X=200. If, after arriving of the external signal, the control stops at the point with the coordinate of X=130, the displacement in the direction X in the block N2 will be X=300 - 130, i.e. X=170.

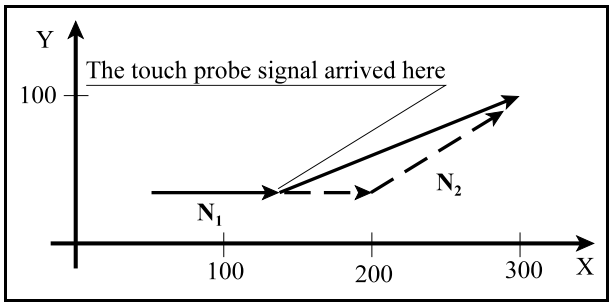


Fig. 21.1-3

21.2 Torque Limit Skip (G31)

The function G31 can also be used in the way by ***skipping the motion*** not by an external signal (for example by the signal of the probe) but when ***the torque of the motor at a specified axis reaches a given value***, for example, when the tool is pressed against a fixed surface. At the moment of reaching the torque limit or at the programmed endpoint of the block the control stops, and then it goes on to execute the next block.
It is the function

G31 P98 Qq v Ff

that realizes the actions mentioned above, where:

Q: the value of the programmable torque limit expressed in percent of the maximum torque of the motor. Q0 corresponds to 0% and Q255 corresponds to 100%. The values that can be given: Q1-Q254

v: the name and the endposition of the programmed axis on which the torque limit is monitored by the control. ***Only one axis address has to be specified***

F: the value of feed in mm/min or inch/min. (G94 state is needed).
During motion, the value of the ***feed F*** will be:
– the specified or modal value F if the bit #0 SKF of the parameter N3001 G31 Config is 0;
– the value F taken from the parameter N0311 G31 Feed if the bit #0 SKF of the parameter N3001 G31 Config is 1.

The command ***G31 P98 is not a modal one***; it is valid only in the block, in which it was programmed.

An example:

```
N1 G31 P98 Q50 Z30 F100
N2 G0 Z200
```

In the block N1, ***the tool reaches the workpiece surface at the point 'A'***. As from this point, the axis Z will not move already. Because the motor has not reached the torque limit Q50 (19.6%) yet, the control will not stop the motion.
The ***position error*** on the axis Z ***from the point 'A' to the point 'B'*** will increase continuously while the torque of the motor increases continuously.
Issuing the motion command continues up to ***the point 'B'*** where ***the motor reaches the torque limit***.

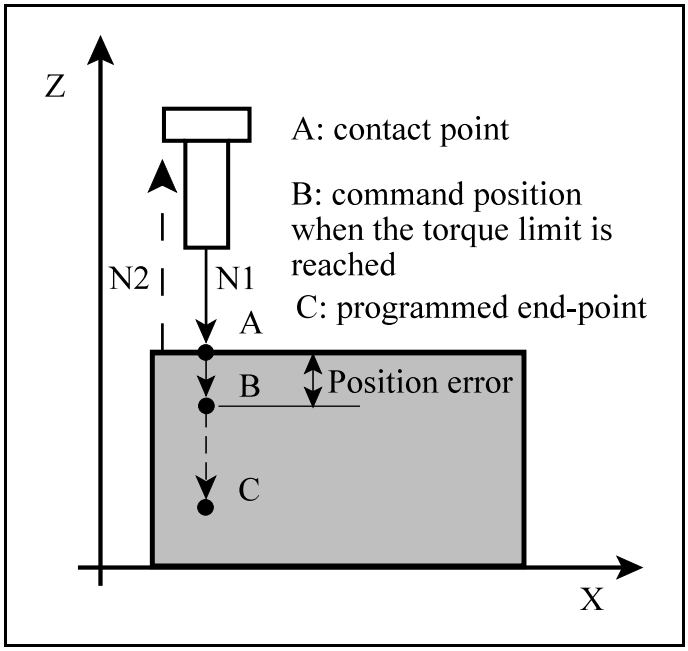


Fig. 21.2-1

At the point 'B', the control stops the motion, ***records the position of the point 'A', and then begins execution of the next block N2***.

The position error increases continuously increases from the point 'A' to the point 'B' while the torque of the motor increases because the tool pushes the workpiece. In order that the control does not run to a servo error during execution of the function, an error limit greater than the limit set in the parameter N0520 Serrl2 can be set in the parameter N3019 Servo Limit during Torque Limit Skip. The control will send the message '3157 Servo error during G31P98' only after exceeding this limit. This servo error does not cause emergency situation but it suspends execution of the program.

At the moment of reaching the torque limit, ***the position of the axis recorded at the point 'A' will be stored*** in the following ***macro variables***:
#5061, or #100151, or #_ABSKP[1]: the position of the axis 1;
#5062, or #100152, or #_ABSKP[2]: the position of the axis 2;
...
The position stored in the abovementioned macro variables will be:
– the position of the point 'A' if ***torque limit signal has been received***;
– the position of programmed endpoint of the block G31 P98 if ***torque limit signal has not been received***.

The position data will be stored
– always ***in the actual workpiece coordinate system***;
– ***without taking the actual length compensation*** (G43 and G44) into account.

Error messages
If the address Q is not filled in, the control will send the message '2004 The data Q is missing'.
If the value of the address Q is less than 1 or greater than 254, the control will send the message '2039 Q definition error'.
If more than one axis address is referred in the function, the control will send the message '2035 <axis address> axis definition is illegal', where <axis address> is the address of the programmed axis 2 according to the axis number order.
If the control does not receive torque (current) information about the programmed axis, it will send the message '2156 Probe status error on the channel 98'.

21.3 Automatic Tool Length Measurement (G37)

Due to the instruction
G37 q,
the length compensation of the tool changed will be measured and corrected along the axis specified at the coordinate q. The value of the q is always interpreted as absolute data, in the coordinate system of the actual workpiece.

The motion is performed to the position ***q - Rapid Distance at the rapid traverse rate***, where Rapid Distance is the distance set in the parameter N3004 Rapid Distance.

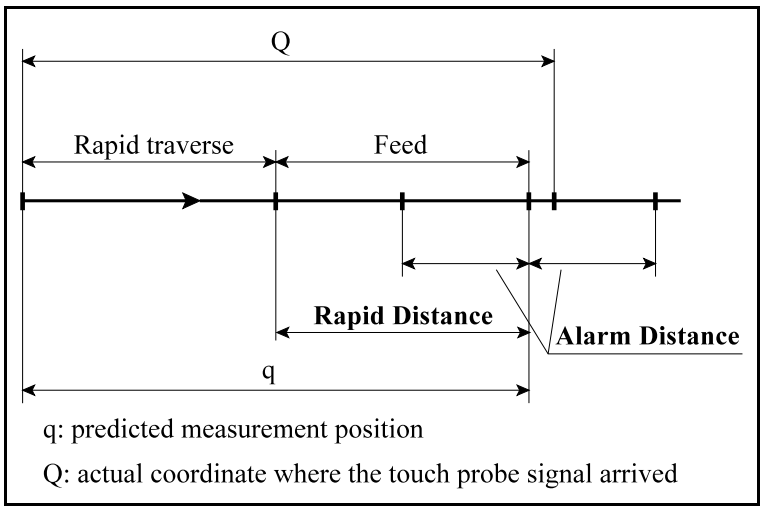


Fig. 21.3-1

Then, the motion continues ***at the feed rate*** until ***touch probe signal arrives or the control indicates error***. The error message '2104 Measurement position out of area' will be sent, if the touch probe signal arrives from a place which is out of the range with the radius of Alarm Distance, being around the position programmed at the address q (predicted measurement position). The distance Alarm Distance is a value set in the parameter N3005 Alarm Distance.

In the block G37, the value of the feed will be:
– a modal value from the program, if the bit #0 TLF of the parameter N3003 G36, G37 Config is 0; or
– a value set in the parameter N0312 G37 Feed Feed, if the bit #0 TLF of the parameter N3003 G36, G37 Config is 1.

If the measurement is completed successfully and the touch probe signal arrived at the point with the coordinate Q, the following compensation modifications will be possible in the length compensation registers called previously at the address H:

– ***geometry compensation*** will be modified if the bit #1 TMW of the parameter N3003 G36, G37 Config is 0;
– ***wear compensation*** will be modified if the bit #1 TMW of the parameter N3003 G36, G37 Config is 1.

Modifying the length compensation is as follows:
– ***the difference q-Q will be subtracted from the*** appropriate ***compensation*** if the bit #2 TCA of the parameter N3003 G36, G37 Config is 0;
– ***the difference q-Q will be added to the*** appropriate ***compensation*** if the bit #2 TCA of the parameter N3003 G36, G37 Config is 1.

Prior to starting the measurement, ***the appropriate value H and length compensation has to be called***.
– The G37 is one-shot instruction.
– The cycle G37 is always executed in the coordinate system of the actual workpiece.
– The parameters Rapid Distance and Alarm Distance are always positive values. For these two parameters, the following condition has to be satisfied: Rapid Distance > Alarm Distance.
–The function can be called only in the state G15, G50, G50.1, G69, G94, otherwise an error message will be sent by the control.

An example:

```
G55 G15 G50 G50.1 G69 G94
...
G43 G0 Z100 H5
X300 Y200
G37 Z50 F200
G0 Z100
...
```

22 Safety Functions

The following three safety zones can be set on the control:
- ***Stroke ends:*** they determine the limit of axis travel, entering beyond which is forbidden. Switches or parameters are used for limitation.
- ***Working area limitation (Stored stroke check 2):*** it can be started and cancelled from program by the functions G22 and G23, respectively. It can be set by specification of parameters too. Both inside and outside forbidden areas can be defined.
- ***Area forbidden internally (Stored stroke check 3)***: an area moving into which is forbidden. It can be set using parameters.

22.1 Stroke end

The stroke ends limit axis travel. Stroke ends can be managed from switch or from parameter. They always forbid outside area.

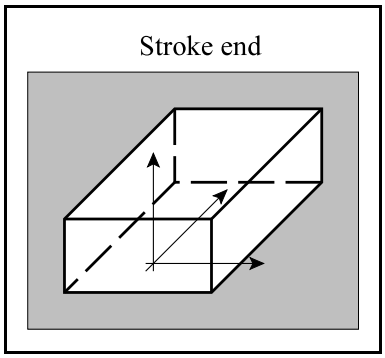


Fig. 22.1-1

Managing stroke end from switch

It is the PLC program that manages signals from the limit switches and transmits them to the control. When an axis moves onto a limit switch, the error message
'3018 Axis # on positive limit switch' or
'3019 Axis # on negative limit switch'
will be sent by the control, where # is the name of the axis.
A ***disadvantage*** of stoke limitation using limit switch is that ***the control begins to decelerate after moving onto the switch***. On the machines featured by high rapid traverse rate, ***it would be necessary to decrease the stroke to a great extent because of the long deceleration distance***, in order that the axis will be able to stop from rapid traverse. Furthermore, monitoring the stroke end prior to motion start does not work either.

Managing stroke end from parameter

In the case of managing stroke end from parameter, the control knows momentarily the distance to the stroke end, and it always begins to decelerate at the right moment in accordance with the axis speed. In this way, ***the machine stroke is fully utilized by parametric stroke end***.
When an axis moves onto a parametric stroke end, the error message
'3010 Positive limit on axis # obtained' or
'3011 Negative limit on axis # obtained'
will be sent by the control, where # is the name of the axis.
The parametric stroke ends are determined by the builder of the machine tool.
Two stroke end ranges, A and B, can be set in parameter. During operation of the machine, ***the PLC program determines*** which stroke end ***(A or B) should be effective*** on which axis and in which direction.
For example, in a normal case, the stroke end range A is effective on the axis Z. If, in the course of tool change, the change arm grips the tool, the PLC program will switch over to the stroke end range B; in this way, the stroke end range of the axis Z is adjusted within such narrow boundary that does not allow tearing the change arm off by the motion of the axis Z.

Leaving the stroke end

If, in the course of program run or manual moving, any of the axes moves onto a stroke end, ***releasing it will be possible by moving manually only***.

22.2 Working Area Limitation from Parameter/program (G22, G23)

Working area limitation from parameter

Working area limitation can be given by setting the following parameters. Parameters can be set only in the Edit mode.

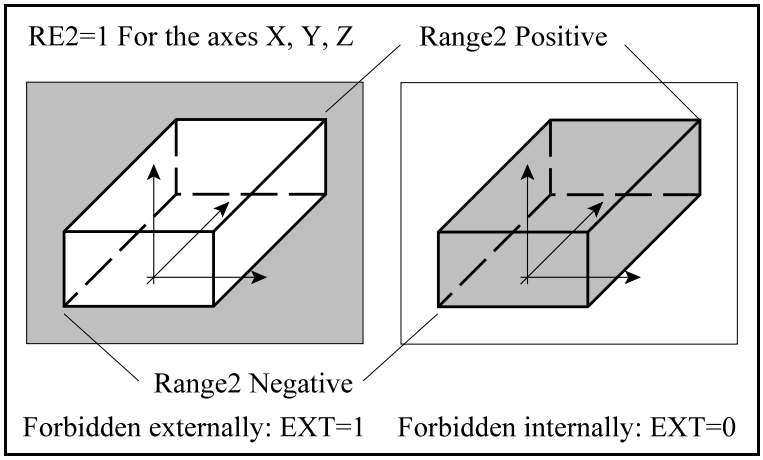


Fig. 22.2-1

With setting the parameter bit #1 RE2 of the parameter N1000 Range Enable to 1, it can be determined ***which axis*** will participate in working area limitation.

The point of positive and negative direction of the working area to be limited has to be given in the parameters N1006 Range2 Positive and N1007 Range2 Negative ***in the machine coordinate system***, for each axis.

The condition

N1006 Range2 Positive > N1007 Range2 Negative

has to be satisfied for all assigned axes!

Using the enable bit and the machine positions***, all the axes on the machine can be assigned in parameter*** for working area limitation.

It can be given at the bit #0 EXT of the parameter N1001 StrkContr ***whether the assigned working area will be forbidden internally*** (when EXT=0) ***or externally*** (when EXT=1).

The working area limitation has to be enabled with setting the bit #1 STE of the parameter N1001 StrkContr to 1. If the bit STE is 0, the working area limitation will not operate.

The bits EXT and STE has to be set for each channel. The working area limitation is always related to the axes belonging to the given channel.

Following the entering or rewriting the parameters, the single block

```
G23
```

has to be issued in one of the manual modes. As a result of this, the entered or modified parameters will be taken into account by the control.

The figure illustrates a working area limitation for the axes X, Y, Z. Certainly, either less or more axes can be assigned for this function.

If the machine moves into an area forbidden internally (EXT=0), the error message

'3042 Internally forbidden area 2'
will be sent by the control.
If the machine moves into an area forbidden externally (EXT=1) in positive or negative direction, the error message
'3040 Forbidden area 2 # +' or
'3041 Forbidden area 2 # -'
will be sent by the control, where # is the name of the axis.

Leaving the area forbidden by parameter
If, in the course of program run or manual moving, any of the axes moves into forbidden area, the following method should be applied:
The case when the ***working area is forbidden externally.***
The area can be left ***by moving manually***, similarly to leaving the stroke end.
The case when the ***working area is forbidden internally***.
By switching the PLC flag on: The program has to be ***reset***, ***the error has to be deleted***. By switching the flag CP_LIM2DIS PLC on (e.g. ***pushing a button and keeping it pushed***), monitoring the forbidden area 2 can be suspended and leaving the area will be possible without rewriting the parameters. *For details the builder of the machine tool has to be asked .*
By parameter setting: Monitoring the working area limitation ***has to be cancelled by parameter setting STE=0***, the instruction G23 has to be issued in single block, ***the area has to be left by moving manually***, ***monitoring has to be started again*** by parameter setting STE=1, and the instruction G23 has to be issued again.

Working area limitation from program
The instruction
G22 X Y Z I J K P
starts monitoring the working area limitation. The motion ranges of the axes can be limited by this instruction. The meaning of the addresses of the instruction are as follows:
X: Limit along the axis X in the positive direction;
I: Limit along the axis X in the negative direction;
Y: Limit along the axis Y in the positive direction;
J: Limit along the axis Y in the negative direction;
Z: Limit along the axis Z in the positive direction;
K: Limit along the axis Z in the negative direction.
The following conditions have to be satisfied for data above:

$$X \geq I,\ Y \geq J,\ Z \geq K$$

All the coordinate data (X, Y, Z, I, J, K) ***have to be given in the machine coordinate system***.
It can be given at the address P whether moving beyond or into the assigned space is forbidden.
In the case of P=0, the internal zone of the assigned space is forbidden.
In the case of P=1, the external zone of the assigned space is forbidden.

Due to the instruction G22, the values of the range 2 set in parameter will be ignored by the control, and only those values will be taken into account that were given in the instruction G22.

The instruction

G23

cancels monitoring the working area limitation.

The instruction G23 deletes the borders of the range set in the instruction G22, and, at the same time, resets the values of monitoring the range 2 set in parameter.

– Working area limitation can be given only for ***main axes***.
– The instructions G22 and G23 has to be given in independent block..
– Working area limitation will be effective after turning on and machine reference point return.
– In the case of X=I, Y=J, Z=K and P=0, the entire area is allowed.
– In the case of X=I, Y=J, Z=K and P=1, the entire area is forbidden.

If the machine moves into an area forbidden internally (G22 P0), the error message

'3042 Internally forbidden area 2'

will be sent by the control.
If the machine moves into an area forbidden externally (G22 P1) in positive or negative direction, the error message

'3040 Forbidden area 2 # +' or
'3041 Forbidden area 2 # -'

will be sent by the control, where # is the name of the axis.

Leaving the forbidden area programmed using the function G22
If, in the course of program run or manual moving, any of the axes moves into forbidden area, the following method should be applied:

The case when the working area ***is forbidden externally.***

The area can be left ***by moving manually***, similarly to leaving the stroke end.

The case when the working area ***is forbidden internally.***

By switching the PLC flag on: The program has to be ***reset***, ***the error has to be deleted***. By switching the flag CP_LIM2DIS PLC on (e.g. ***pushing a button and keeping it pushed***), monitoring the forbidden area 2 can be suspended and leaving the area will be possible. *For details the builder of the machine tool has to be asked .*

From program: In manual mode, monitoring the working area limitation ***has to be cancelled using function G23***, ***the area has to be left by moving manually***, and ***monitoring has to be started again*** by issuing the complete instruction G22 again.

If, on the control, the forbidden area 2 is set from parameter, but working area limitation is issued using the instruction G22, the area specified by the G22 will be forbidden until it will be deleted by the G23. Then, the area 2 specified in parameter will be forbidden again.

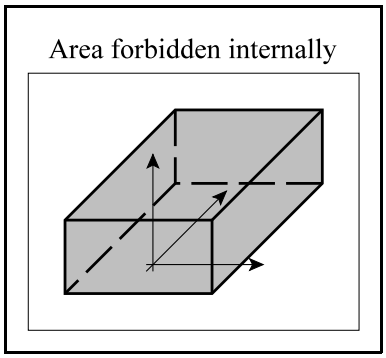


Fig. 22.3-1

22.3 The Area Forbidden Internally

An area always forbidden internally can be defined by parameters on the control. If one or maybe more of the axes moves into this area or to the border of it, the error message

'3042 Internally forbidden area 3'

will be sent by the control.

Leaving the area forbidden internally

By switching the PLC flag on: The program has to be ***reset***, ***the error has to be deleted***. By switching the flag CP_LIM3DIS PLC on (e.g. ***pushing a button and keeping it pushed***), monitoring the area 3 forbidden internally can be suspended and leaving the area will be possible without rewriting the parameters. *For details the builder of the machine tool has to be asked.*

By parameter setting: If the machine moves into the abovementioned area forbidden internally, the bit #2 RE3 of the parameter N1000 Range Enable, i.e. enabling the monitoring the area forbidden internally, will have to be cancelled for each axis by writing the bits RE3 to 0. Then, the area has to be left by moving manually, and the bits #2 RE3 have to be set again.

The bits RE3 can be set only in the Edit mode.

22.4 Monitoring the Forbidden Area Prior to Motion Start

In the bit state #2 CBM=1 of the parameter N1001 StrkCont, ***prior to starting a motion instruction***, ***the control will check*** in automatic or manual data input mode or when a single block is being started, ***whether the endpoint of the given block will fall into one of the forbidden areas or not***.

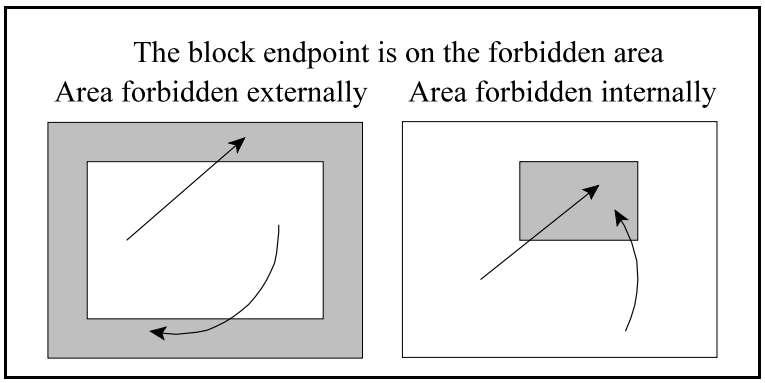


Fig. 22.4-1

If the endpoint of the block ***falls beyond the stroke end whithin the forbidden area 1***,
the error message

‘2056 Endpoint on positive limit on axis #’

‘2057 Endpoint on negative limit on axis #’

will be sent by the control, where # is the name of the related axis; and ***motion will not be started***.

the error message

‘2060 Endpoint in internally forbidden area 2'

will be sent by the control, and ***motion will not be started***.
If the endpoint falls within ***the externally forbidden area 2***,
the error message

‘2058 Endpoint in forbidden area 2 # +’

‘2059 Endpoint in forbidden area 2 # -‘

will be sent by the control, where # is the name of the related axis; and ***motion will not be started***.
If the endpoint falls within ***the internally forbidden area 3***,

the error message
'2061 Endpoint in internally forbidden area 3'
will be sent by the control, and ***motion will not be started***.

In the bit state #2 CBM=1 of the parameter N1001 StrkCont, ***prior to starting a motion instruction***, ***the control will check*** in automatic or manual data input mode or when a single block is being started, ***whether the path of the given block intersects the area forbidden internally***.

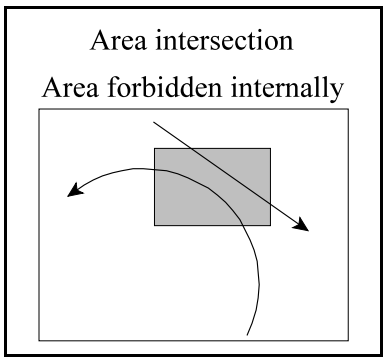


Fig. 22.4-2

If ***the path of the block intersects the internally forbidden area 2***, but the endpoint is not within the forbidden area,
the error message
'2062 Entry into second internally forbidden area'
will be sent by the control, and ***motion will not be started***.
If ***the path of the block intersects the internally forbidden area 3***, but the endpoint is not within the forbidden area,
the error message
'2063 Entry into third internally forbidden area'
will be sent by the control, and ***motion will not be started***.

The errors mentioned in the cases above can be eliminated by rewriting the programmed coordinates and by modifying the zero points or tool compensations.

23 Custom Macro

The conventional NC programming language describes the desired path and switches the various functions on or off by specifying codes G, M, S and T. Specific numerical value is assigned for a given address. For example, if the axes are to be moved to the position X50 Y100, the block

```
G0 X50 Y100
```

will have to be programmed.

With the application of the macro language, ***it is not necessary to assign a specific numerical value***, e.g. X50, to a given address, but instead, ***the value of a variable can also be assigned to it***; for example, it can be written in the program that

```
G#105 X#102 Y#110
```

where #105, #102 and #110 are the values of three different variables to which a value was assigned earlier.

In the programming language, various ***arithmetical expressions*** and ***functions*** can be used, for example addition, square-root extraction, sine function etc.

Assignment instructions, ***condition check*** instructions, ***branch*** instructions and instructions executing ***cycle*** can be used.

The programming language enables such subprograms, ***macros to be called*** to which ***arguments*** (parameters) ***can be passed*** from the calling block.

With parameter specification, such so-called ***system macros or system subprograms*** can be generated that can be used by the user to extend or modify the instructions of conventional programming language of codes G in accordance with his demands.

23.1 Variables of the Programming Language

In the main program, subprograms and macros, ***variables can also be assigned*** to the addresses, instead of specific numerical values.
Value within a permissible range ***can be assigned to the variables***. By the use of variables, programming can be made much flexible.
Variables can be classified as

local variables that are used for passing argument in macro calls;
common variables that can be accessed at every level of macro call; and
system variables.

The system variables are those internal data of the control that can be read out or rewritten from the part program.

23.1.1 Referring to Variables

Variables can be referred by ***number***, but system variables can be referred either by number or by ***symbol***.
Referring to a variable has to always be begun with the number sign #.

Referring to variable by number

The identifier of the variable is ***a number following the number sign #***:

#<number>

For example:

#12
#138
#5106

A variable ***can also be referred indirectly***, by a formula: ***#[<formula>]***
For example:

#[#120] means that the variable No. 120 contains the serial number of the variable referred to.
#[#120-4] means that subtraction 4 from the number contained in the variable No. 120 gives the number of the variable referred to.

Referring to system variables by symbol

Symbolical reference is also begun with the number sign # followed by the character _ (***underscore symbol***):

#_<symbol>

For example:

#_ALM (error message)

In certain cases, an ***index*** has to also be added to the symbolic variables. The index has to be put into ***brackets [ ]***:

#_<symbol>[index]

For example:

#_ABSIO[3] means block end position of the axis 3, axis index: [3].

In the words of the program block, the various addresses can take on not only numerical values, but also values of variables. In the case of referring to variable behind addresses, the minus sign – or the operator I can also be used wherever it is permitted in the case of numerical values. For example:

G#102

if #102=1.0, this reference is equivalent to G1

XI–#24

if #24=135.342, this reference is equivalent to XI–135.342

– It is not permitted to refer to a variable behind addresses of block number N and conditional block /. The first address N written in the block is considered by the control to be block number.

– The number of a variable cannot be substituted by a variable, i.e. writing ##120 is not permitted. The correct specification is #[#120].

– If a variable is used behind an address, the value of the variable will not have to exceed the range of values permitted for the given address. For example, after the value specification

#112=123456789

the reference

M#112

will cause error message.

– If a variable is used behind an address, the value of the variable will be rounded to the significant digit corresponding to the address. For example:

if the #112=3.23, the M#112 will be M3,

if the #112=3.6, the M#112 will be M4.

23.1.2 Number Representation of Macro Variables

Disregarding few exceptions, macro variables are floating-point numbers. The macro variables being not floating-point ones will have special marking in their description.

In the control, representation of the floating-point numbers follows the ***double precision representation of the floating-point numbers*** in accordance with ***the standard IEEE 754***. These numbers are represented on 64 bits.

Using the double precision representation of the floating-point numbers, the numbers

from $\pm 5.0 \times 10^{-324}$ to $\pm 1.7 \times 10^{308}$

and the 0 can be represented with precision of 15-16 digits. The ***decimal point*** (.) has to be used during input, but it is not necessary when the numbers to be input are integer numbers:

#100=256

Neither leading zeros nor following zeros have to be given. The positive sign (+) can be omitted:

#100=134.89654

23.1.3 Local Variables: #1 – #33

The local variables are used by the macro program at a given point, locally.

Generally, the local variables are used ***to pass arguments***.

The local variables are ***multi-level*** variables; different levels belong to the main program and to the various macro calls, and that is because they are called local. For example, the value of the #1 can be different in the main program than, let us say, in the level 2 of the macro calls.

After returning from macro call, the local variables of the given level will be eliminated to #0, they will be deleted to vacant. ***The local variables of the main program will be deleted at the end of the program***.

Fitting the addresses of the arguments and the local variables, and managing the levels are covered by the section 23.3 Calling the Macros, System Macros and System Subprograms on page 325.

The local variables the address of which is not included in the argument assignment is null and can be used freely.

23.1.4 Common Variables: #100 - #499, #500 - #999

Unlike the local variables, ***the common variables are identical in the whole channel*** regardless of whether they are used in the main program, subprogram or macro and in which level of the macro call.
In the system, ***the use of common variables absolutely free***, they do not have any dedicated role.

The following two groups of the common variables are distinguished:

The common variables from #100 to #499 that will be deleted upon power-off.
The common variables from #500 to #599 the value of which will be retained even after power-off.

The common variables from #500 to #999 can be made write-protected using the parameters N1702 Write Prt Low and N1703 Write Prt Hig.
The first element of the array to be protected has to be written in the parameter N1702 Write Prt Low, but the last element of the array declared protected has to be written in the parameter N1703 Write Prt Hig.
For example, if the common variables from#530 to #540 are to be made write-protected, the parameter settings N1702 Write Prt Low=530 and N1703 Write Prt Hig=540 will have to be applied.

If the control manages several channels, ***an array of the common variables can be made accessible in each channel*** using parameter.
The parameter N1700 No. of Common #100 determines the number of the macro variables from #100 to #499 that can be called from each channel. In each channel, the macro variables with the number from 100 to the parameter 100 + No. of Common #100 will be common. This parameter has to be smaller than 400.
If the value of the parameter, let us say, 40, the macro variables from #100 to #139 will be common for each channel.
The parameter N1701 No. Of Common #500 determines the number of the macro variables from #500 to #999 that can be called from each channel.In each channel, the macro variables with the number from 500 to the parameter 500 + No. of Common #500 will be common. This parameter has to be smaller than 500.
If the value of the parameter, let us say, 30, the macro variables from #500 to #529 will be common for each channel.

23.1.5 Notation Used in Description of System Variables

The system variables are those internal data of the control that can be read out or rewritten from the part program.

The axes can be identified only from 1 to 20 on the macro variables belonging to the axes and identified by a number under 10000. For the case of greater axis number, the identification numbers over 100000 has been introduced. Referring to axes can be done on the numbers from 1 to 50. For example:

#100001

can also be used to identify the block end position of the axis 1. Certainly, referring to the data above can also be done using numbers smaller than 10000, and symbols

Notation used in description of system variables is as follows:

[n]: the index of the variable. It can be, for example the number of an axis or a spindle;
R: an attribute of the variable: read-only variable;
W: an attribute of the variable: write-only variable;
R/W: an attribute of the variable: readable and writable variable.

23.1.6 Vacant Variable. Constants

Number	Symbol	Attribute	Description
#0, #3100	#_EMPTY	R	Vacant constant
#3101	#_PI	R	π= 3.14159...
#3102	#_E	R	Base of natural logarithm: e=2.71828...

Vacant variable #0, #3100, #_EMPTY (R)
When a macro is called, if value is not assigned to an address, the value of the local variable belonging to that address will be vacant in the body of the macro.
For example: After calling

G65 P100 X20 Y30,

the value of the local variable #1 in the macro O0100 will be vacant because value was not assigned to the address A in the call G65. It can be decided with the checking

#1 EQ #0

in the body of the macro whether the address A was filled in the course of calling the macro or not.
The vacant variable and the number 0 are not identical ones, they differ from each other!

The difference between the effects of the vacant variable and a variable having the value 0 is as follows:
Referring to the ***vacant*** variable in address:

```
If #1=<vacant>          If #1=0

G90 X20 Y#1          G90 X20 Y#1
     |                    |
  G90 X20              G90 X20 Y0
```

Vacant variable in *assignment* instruction:

```
If #1=<vacant>          If #1=0

  #2=#1                 #2=#1
    |                     |
#2=<vacant>              #2=0

 #2=#1*3               #2=#1*3
    |                     |
  #2=0                  #2=0

 #2=#1+#1              #2=#1+#1
    |                     |
  #2=0                  #2=0
```

The difference between ***the vacant variable*** and ***a variable having the value 0*** in the case of *condition checking*:

```
If #1=<vacant>          If #1=0

#1 EQ #0                #1 EQ #0
    |                       |
satisfied           not satisfied

#1 NE 0                 #1 NE 0
    |                       |
satisfied           not satisfied

#1 GE #0                #1 GE #0
    |                       |
satisfied           not satisfied

#1 GT 0                 #1 GT 0
    |                       |
satisfied           not satisfied
```

23.1.7 Variables Between the Part Program and the PLC Program

Information exchange between the part program and the PLC program can be realized by the variables described below.

☞ ***Warning!*** *The kind of information the PLC program transmits to the part program and receives from the part program through various system variables is determined the builder of the machine tool.*

Number	Symbol	Attribute	Description
#1000...#1031	#_UI[n] n=0 - 31	R	32 ***bit variables*** of the PLC program transmitted to the control. Value set: 0, 1
#1032	#_UIL[n] n=0	R	32 ***bit variables*** of the PLC program transmitted to the control as a ***32-bit integer number***. Value set: 0 ... 2^{32}-1

Number	Symbol	Attribute	Description
#1033...#1035	#_UIL[n] n=1, 2, 3	R	3 ***floating-point variables*** of the PLC program transmitted to the control.
#1100...#1131	#_UO[n] n=0 - 31	R/W	32 ***bit variables*** of the part program transmitted to the PLC. Value set: 0, 1
#1132	#_UOL[n] n=0	R/W	32 ***bit variables*** of the part program transmitted to the PLC as a ***32-bit integer number***. Value set: 0 ... 2^{32}-1
#1133...#1135	#_UOL[n] n=1, 2, 3	R/W	3 ***floating-point variables*** of the part program transmitted to the PLC.

23.1.8 Messages of the Part Program

From the macro program, errors can be indicated and messages can be sent to the operator:

Number	Symbol	Attribute	Description
#3000	#_ALM	W	Error message; it can be deleted by the button Cancel
#3006	#_MSGSTP	W	Stop accompanied by a message; continuation by Start
#3106	#_MSG	W	Displaying the message in the program list window
#3107	#_MSGBOX	R/W	Sending the message to the Windows window

Error message: #3000, #_ALM (W)
With the value assignment
#3000=nnn(ERROR INDICATION)
or
#_ALM=nnn(ERROR INDICATION),
numbered (nnn: maximum three digits) and/or text error message can be sent. The text has to be written between round brackets (,).
If an error in the macro is detected by the program, i.e. the program runs on a branch where value was assigned to the variable #3000, the program will be executed up to the previous block, and then the execution will be suspended, and the error message given between brackets or the code of the message will be displayed on the screen in the form of
ii4nnn00 (ii: the number of the channel in which the error occurred),
i.e. 4000 will be added to the number nnn given at the value #3000. If a number is not given, the code of the message will be 4000; if a text is not given, only the code will appear. The error message can be deleted by the button CANCEL.

Stop with a message: #3006, #_MSGSTP (W)
With the value assignment
#3006=nnn(MESSAGE)
or
#_MSGSTP=nnn(MESSAGE),
execution of the program will stop, and the message given between brackets or the code of the message will be displayed on the screen in the form of
ii5nnn00 (ii: the number of the channel in which the error occurred)
i.e. 5000 will be added to the number nnn given at the value #3006. If a number is not given, the code of the message will be 5000; if a text is not given, only the code will appear. Pushing the button START the execution of the program will continue, and the message will disappear from the screen. This instruction is useful in the cases when operator intervention is needed during execution of the instruction.

Displaying a message in the program list window: #3106, #_MSG (W)
With the value assignment
#3106=nnn(MESSAGE)
or
#_MSG=nnn(MESSAGE)
the program continues without stop, and ***the text of the message will be displayed in the top line of the Program list window*** in the form of:
MSGnnn: (MESSAGE)
The text of the message remains there until it is overwritten by a new #3106 or #_MSG instruction. RESET, program end (M30) deletes the message.
In order that the message text appears on the top line of the program list, **the displaying has to be enabled** using the function keys as follows:
F5 View - F1 Program list - F9 Settings - F3 #_MSG info
It can be used for indication of program parts indication for example:

```
...
#3106=1(roughing by the use of a diam. 30 milling cutter)
...
...
#_MSG=2(finishing by the use of a diam. 20 milling cutter)
...
```

Sending a message to the Windows window: #3107, #_MSGBOX (R/W)
With the value assignment
#3107=nnn(MESSAGE)
or
#_MSGBOX=nnn(MESSAGE)
execution of the program will stop (there will be STOP status), and the message given between brackets and the code of the message will be displayed in the message row of the control in the form of
ii6nnnjj (ii: the number of the channel in which the message was issued).
The message will appear not only in the upper status line but in the centre of the screen, **in a Windows message window** too.
It is the code of the message which determines the type of the message window. If the message assignment is

between **nnn=100 - 199**, the message window will appear **with an 'OK' button**, so the operator can only accept the displayed message. If the message assignment is

between **nnn=200 - 299**, the message window will appear **with a 'Yes' button and a 'No' button**. In this case, the operator has alternative. In case of any other value assignment, an error message will be sent by the control.

After responding to the messages appeared in the Windows window, the message will be deleted and the #3107 variable will take the following values in the different cases:

#3107=0: the **X (closing)** button was clikked or the **CANCEL** button was pushed;
#3107=1: the **'OK'** button was clicked;
#3107=2: the **'Yes'** button was clicked;
#3107=3: the **'No'** button was clicked.

During displaying the message, if the intension is to display a given part of the text in several rows in the message window, it is possible to divide up the text by the use of the '\n' character and insert a line character. In the case of displaying a value, the macro variable formatting described at the instruction DPRNT can be applied.

Example 1:

```
##3107=100(Values of the measurement results\nX length:
     #140[53]\nY length: #141[53])
```

Example 2:

```
#140=0.3458
#141=0.9123
(Measurement)
(Displaying the result of the compensation calculation)
#_MSGBOX=200(Values of the tool wear\nX diameter wear:
     #140[53]\nZ-direction wear: #141[53]\nDo you want to
     input the new compensation?)
IF [[#_MSGBOX] EQ 2] GOTO10
GOTO20
N10 (Yes branch)
(Inputting the wear)
N20
(Continuing the program)
```

23.1.9 Clock, Timers and Counters

Number	Symbol	Attribute	Description
#3001	#_CLOCK1	R/W	Millisecond timer, usable freely
#3002	#_CLOCK2	R/W	Cutting time (ms)
#3011	#_DATE	R	Date: year/month/day
#3012	#_TIME	R	Time: hour/minute/second
#3901	#_PRTSA	R/W	Number of the parts produced
#3902	#_PRTSN	R/W	Number of the parts to be produced

Millisecond timer: #3001, #_CLOCK1 (R/W)
The value of this variable is writable and readable.
The time elapsed between two points in time that can be measured in millisecond. Measuring the value of the variable is started from zero at the moment of turning the control on, and performed up. Time is always measured when the control is on. Using the instruction

#3001=0

or

#_CLOCK1=0,

the variable can be set to zero by value assignment from the program; in this case measurement starts from zero, and later the value can be queried from the program, using the instruction

#100=#3001

or

#100=#_CLOCK1.

Cutting time: #3002, #_CLOCK2 (R/W)
The value of this variable is writable and readable. The time elapsed during machining with feed in automatic mode and start state (G1, G2 etc.) is measured in millisecond, cumulatively from the beginning of the control life. The value of the variable can be read on the display Cutting time of the screen Clock times and part counters.

Date: #3011, #_DATE (R)
The actual date can be read from the variable in the format year/month/day.
After using the instruction

#100=#3011

or

#100=#_DATE,

if the value of the variable #100 is, for example,

20140518,

it will mean that the date is: 2014 (year), 05 (month, May), 18 (day, 18).

Time: #3012, #_TIME (R)
The actual time can be read from the variable in the format hour/minute/second.
After using the instruction

#100=#3012

or

#100=#_TIME,

if the value of the variable #100 is, for example,

20140518,

it will mean that the time is: 15 hours (3 hours p.m.), 32 minutes, 41 seconds.

Number of the parts produced/to be produced: #3901, #_PRTSA / #3902, #_PRTSN (R/W)
The values of these variables are writable and readable. The number of the parts produced is accumulated by the control in the counter #_PRTSA having the number #3901. The control will increase the content of the counter by 1 in the course of execution of each function M02, M03 or function M given in the parameter N2305 Part Count M.
When the number of the parts produced reaches the number of the parts to be produced (the counter #_PRTSN having the number #3902) the PLC will be notified.

The number of the parts produced #3901, #_PRTSA
The number of the parts to be produced #3902, #_PRTSN
The value of the numbers of the parts produced and to be produced can be read on the display Produced and to be produced of the screen Clock times and part counters.

23.1.10 Variables Influencing the Operation of the Automatic Mode

Number	Symbol	Attribute	Description
#3003	#_CNTL1	R/W	Control variable 1 (block-by-block)
#3003 bit 0	#_M_SBK	R/W	Disabling the block-by-block execution. Value set: 0, 1
#3004	#_CNTL2	R/W	Control variable 2
#3004 bit 0	#_M_FHD	R/W	Disabling the Stop. Value set: 0, 1
#3004 bit 1	#_M_OV	R/W	Disabling the Override and the Stop. Value set: 0, 1
#3004 bit 2	#_M_EST	R/W	Disabling the exact Stop. Value set: 0, 1

☞ ***Warning!*** *By the reset and block end, the values of the bits set at the variables will be deleted!*

Control variable 1: #3003, #_CNTL1 (R/W)
If the value of the variable #3003 or #_CNTL1 is 1 (or an uneven number), the control, in the state of the block-by-block execution, will not stop after execution of a block until the value of this variable will be 0.
With writing the variable
#_M_SBK ,
disabling the block-by-block execution can also be referred using bits (assigning0 or 1).
By the power on or reset the value of the variable will be 0.
The block-by-block execution
will not be disabled if the value of the variable is 0;
will be disabled if the value of the variable is 1.

Control variable 2: #3004, #_CNTL2 (R/W)
The following values cab be assigned to the variable #3004 or #_CNTL2:

Value of the #3004 or the #_CNTL2	Disabling the exact stop G61,G9	Disabling the Override and the Stop	Disabling the Stop button
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

The Exact stop, the Override and the Stop
will not be disabled if the number written in the right column is 0;
will be disabled if the number written in the right column is 1.
The suppressions above can also be specified by assigning 0 or 1 to the bit-type variables:
The case of the variable **#_M_FHD** (disabling the Stop):
0: there is no disabling;
1: there is disabling.
The block-by-block execution is not disabled.
The case of the variable **#_M_OV** (disabling the Override and the Stop) (the state G63):
0: there is no disabling;
1: there is disabling.
The block-by-block execution is also disabled.
The case of the variable **#_M_EST** (disabling the Exact stop G61, G9) (the state G64):
0: there is no disabling;
1: there is disabling.

23.1.11 Querying the Block Search and Test Statuses

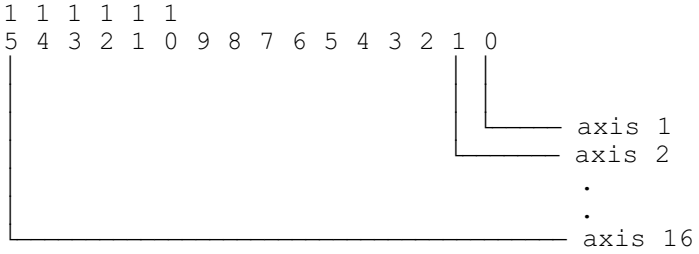
Number	Symbol	Feature	Description
	#_CNTBS	R	Querying the block search and test statuses
bit 0	#_BSEARCH	R	Block search status
bit 1	#_TEST	R	Test mode
bit 2	#_MCHLOCK	R	Machine locked mode

Querying the block search and test statuses: #_CNTBS (R)
All three of statuses in one can be queried at the variable #_CNTBS.
Bit-type querying is also possible for the variable #_CNTBS:
#_BSEARCH: The control executes block search:
=0: it does not execute block search;
=1: it executes block search.
#_TEST: Test mode:
=0: the control is not in test mode;
=1: the control is in test mode.
#_MCHLOCK: The machine is locked:
=0: the machine is not locked;
=1: the machine is locked.

23.1.12 Status of Mirror Image

Number	Symbol	Feature	Description
#3007	#_MIRIMG	R	Status of Mirror Image

By the reading of the variable #3007 it can be determined on which axis a valid mirroring command is entered. The variable is only readable.
Interpreting the value of the variable binary:



The meaning of the bits is as follows:
0: there is no mirroring;
1: mirroring is turned on.
For example, if the value of the variable is 5, it means that mirroring is turned on at the axes 1 and 3. The axis number means physical axis number; it is determined by parameter which axis of what name belongs to which physical axis number.

23.1.13 Number of the Main Program

Number	Symbol	Attribute	Description
#4000	#_MAINO	R	The number of the main program being executed

The number of the actual main program being executed will always be indicated even if a subprogram is in progress. If the automatic mode is interrupted and a program is executed in the MDI mode, the number of the program being executed in the MDI mode will be indicated. If the name of the main program begins with the letter other than O and contains more than 8 digits, the value of the #4000 will be 0.

23.1.14 Modal Information

The modal information valid in the previous block can be obtained by reading the system variables from #4001 to #4199.

Number	Symbol	Attribute	Description
#4001...#4039	#_BUFG[n] n=1-39	R	For each group, the code of the modal function G given last. n: the number of the group of codes G
#4101	#_BUFA	R	The value of the auxiliary function given last and the address of which was defined in the parameter N1333 Aux Fu Addr1.
#4102	#_BUFB	R	The value of the auxiliary function given last and the address of which was defined in the parameter N1334 Aux Fu Addr2.
#4103	#_BUFC	R	The value of the auxiliary function given last and the address of which was defined in the parameter N1334 Aux Fu Addr3.
#4107	#_BUFD	R	The value of the address D given last.
#4108	#_BUFE	R	The value of the address E given last.
#4109	#_BUFF	R	The value of the address F given last.
#4111	#_BUFH	R	The value of the address H given last.
#4113	#_BUFM	R	The value of the address M given last.
#4114	#_BUFN	R	The value of the block number N given last.
#4115	#_BUFO	R	The value of the program number O given last.
#4119	#_BUFS	R	The value of the address S given last.
#4120	#_BUFT	R	The value of the address T given last.
#4130	#_BUFWZP	R	The number of the additional coordinate system given last (G54.1 Pn)

The modal information valid in the block being executed just at the moment can be obtained by reading the variables from #4201 to #4399.

Number	Symbol	Attribute	Description
#4201...#4239	#_ACTG[n] n=1-39	R	For each group, the code of the modal function G valid in the block being executed. n: the number of the group of codes G
#4301	#_ACTA	R	The value of the auxiliary function valid in the block being executed and the address of which was defined in the parameter N1333 Aux Fu Addr1.
#4302	#_ACTB	R	The value of the auxiliary function valid in the block being executed and the address of which was defined in the parameter N1334 Aux Fu Addr2.
#4303	#_ACTC	R	The value of the auxiliary function valid in the block being executed and the address of which was defined in the parameter N1334 Aux Fu Addr3.
#4307	#_ACTD	R	The value of the address D valid in the block being executed.
#4308	#_ACTE	R	The value of the address E valid in the block being executed.
#4309	#_ACTF	R	The value of the address F valid in the block being executed.
#4311	#_ACTH	R	The value of the address H valid in the block being executed.
#4313	#_ACTM	R	The value of the address M valid in the block being executed.
#4314	#_ACTN	R	The value of the block number N valid in the block being executed.
#4315	#_ACTO	R	The value of the program number O valid in the block being executed.
#4319	#_ACTS	R	The value of the address S valid in the block being executed.
#4320	#_ACTT	R	The value of the address T valid in the block being executed.
#4330	#_ACTWZP	R	The number of the additional coordinate system valid in the block being executed

The modal information valid in the interrupted block at the moment of calling the interruption macro can be obtained by reading the variables from #4401 to #4599.

Number	Symbol	Attribute	Description
#4401...#4439	#_INTG[n] n=1-39	R	For each group, the code of the modal function G valid in the interrupted block. n: the number of the group of codes G
#4501	#_INTA	R	The value of the auxiliary function valid in the interrupted block and the address of which was defined in the parameter N1333 Aux Fu Addr1.
#4502	#_INTB	R	The value of the auxiliary function valid in the interrupted block and the address of which was defined in the parameter N1334 Aux Fu Addr2.
#4503	#_INTC	R	The value of the auxiliary function valid in the interrupted block and the address of which was defined in the parameter N1334 Aux Fu Addr3.
#4507	#_INTD	R	The value of the address D valid in the interrupted block.
#4508	#_INTE	R	The value of the address E valid in the interrupted block.
#4509	#_INTF	R	The value of the address F valid in the interrupted block.
#4511	#_INTH	R	The value of the address H valid in the interrupted block.
#4513	#_INTM	R	The value of the address M valid in the interrupted block.
#4514	#_INTN	R	The value of the block number N valid in the interrupted block.
#4515	#_INTO	R	The value of the program number O valid in the interrupted block.
#4519	#_INTS	R	The value of the address S valid in the interrupted block.
#4520	#_INTT	R	The value of the address T valid in the interrupted block.

Number	Symbol	Attribute	Description
#4530	#_INTWZP	R	The number of the additional coordinate system valid in the interrupted block (G54.1 Pn).

☞ ***Warning!*** *In the case of modal codes G containing dot, for example G51.1, the code will be given back by the control in such a way so that the number standing after the dot will be written before the dot. Due to the instruction*

```
#100=#4022,
```

the value of the ***#100 will be 151 in the state G51.1, and it will be 150 in the state G50.***

The following table illustrates ***sorting the modal codes G by group number***.
The one-shot and non-modal codes G belong to the group 0. The group 0 cannot be queried, so the codes G belonging to it are not shown in the table

Group number	The modal codes G belonging to the group	Function
1	G0, G1, G2, G3, G33	Interpolation
2	G17, G18, G19	Plane selection
3	G90, G91	Absolute/incremental programing
4	G22, G23	Forbidden area on/off
5	G94, G95	Feed per minute/revolution
6	G20, G21	Inch/metric data specification
7	G40, G41, G42	Radius compensation off/on
8	G43, G44, G43.1, G43.4, G43.5, G43.7, G43.8, G43.9, G49	Length compensation, controlling the center point of the tool on/off
9	G73, G74, G76, G80, G81, G82, G83, G84, G84.2, G84.3, G85, G86, G87, G88, G89	Drilling cycles
10	G98, G99	Returning from drilling cycle to the initial/R point
11	G50, G51	Scaling off/on
12	G66, G66.1, G67	Modal macro call on/off
13	G96, G97	Constant surface speed on/off
14	G54, G54.1, G54.2, G55, G56, G57, G58, G59	Selecting the workpiece coordinate system

Group number	The modal codes G belonging to the group	Function
15	G61, G62, G63, G64	Feed control functions
16	G68, G69	In-plane rotation on/off
17	G15, G16	Polar coordinate off/on
18	G68.1, G68.2, G69.1	3D rotations on/off
19		
20		
21	G12.1, G13.1	Polar interpolation on/off
22	G50.1, G51.1	Mirroring off/on
23		
24		
25		
26		
27		
28		
29		
30		
31	G50.2, G51.2	Polygonal turning off/on
32		
33		
34	G80.8, G81.8	Electronic gear box off/on
35		
36		
37		
38		
39		

23.1.15 Position Information

Number	Symbol	Attribute	Description
#5001...#5020 #100001...#100050	#_ABSIO[n] n=1-50	R	The block end position of the axis 1 ... n.
#5021...#5040 #100051...#100100	#_ABSMT[n] n=1-50	R	The position of the axis 1 ... n in the machine coordinate system.
#5041...#5060 #100101...#100150	#_ABSOT[n] n=1-50	R	The position of the axis 1 ... n in the workpiece coordinate system.
#5061...#5080 #100151...#100200	#_ABSKP[n] n=1-50	R	The skip position of the axis 1 ... n in the workpiece coordinate system.

☞ ***Warning!*** The read-out of the position information is done in accordance with the axis number. Prior to using this manual, you have to ask the builder of the machine tool for information about the axis numbers.

For example, let us have a two-channel control. There are axes X, Y and Z in the first channel, and axes X and Z in the second channel. Assignment of the axes by the builder of the machine tool is as follows:

The channel 1:
- X – the axis 1
- Y – the axis 2
- Z – the axis 3

The channel 2:
- X – the axis 4
- Z – the axis 5

If the position information is to be queried in the channel 1, the index 3 will have to be used in the case of the axis Z; but the index 5 will have to be used if query is to be done in the channel 2.

Block end position of the axes: #5001...#5020, #100001...#100050, #_ABSIO[n] (R)
The block end position of the axes will be given back
in the coordinate system of the actual workpiece (G54, G55, ...);
using orthogonal coordinates;
disregarding all the compensations (length, radius).

Machine position of the axes: #5021...#5040, #100051...#100100, #_ABSMT[n] (R)
The machine position of the axes will be given back
in the coordinate system of the machine (G53). If the given axis moves, the value given back will change continuously.

Workpiece position of the axes: #5041...#5060, #100101...#100150, #_ABSOT[n] (R)
The workpiece position of the axes will be given back

in the coordinate system of the actual workpiece (G54, G55, ...);
using orthogonal coordinates;
having regard to all the compensations (length, radius). If the given axis moves, the value given back will change continuously.

Skip position: #5061...#5080, #100151...#100200, #_ABSKP[n] (R)
The position in the block G31 where the touch probe signal is received. The position will be given back

in the coordinate system of the actual workpiece (G54, G55, ...);
using orthogonal coordinates;
having regard to all the compensations (length, radius).

If the touch probe signal is not received, the variables above will take on the end point position programmed in the block G31.
After receiving the touch probe signal, de axis will decelerate and stop. The end point position of the block G31 #_ABSIO(n) will take on the value being valid after deceleration and stop.

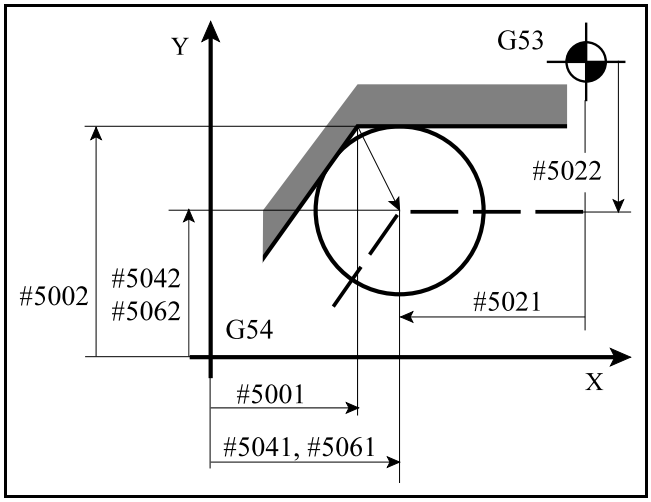


Fig. 23.1.15-1

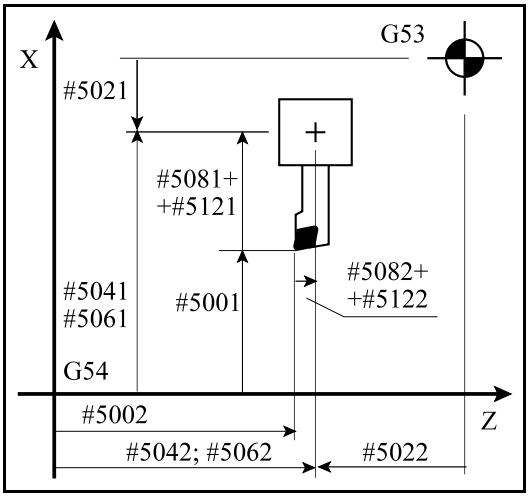


Fig. 23.1.15-2

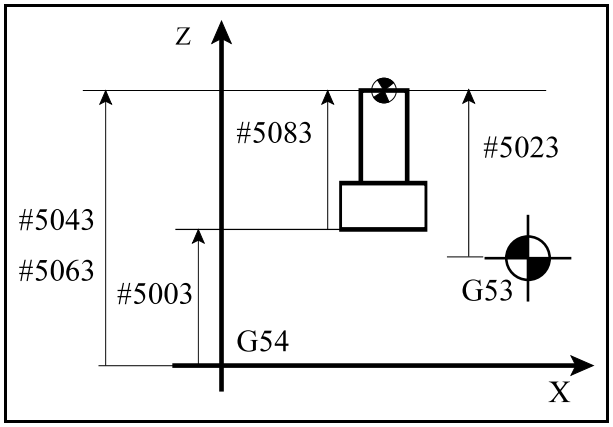


Fig. 23.1.15-3

23.1.16 Value of the Actual Length Compensation

Number	Symbol	Attribute	Description
#5081...#5100	#_TOFS[n]	R	The value of length compensation taken into account at the axis 1 ... n.
#100201...#100250	n=1-50		

From the variables above, the value of length compensation (geometry+wear) taken into account in the block being executed can be read out, for each axis, using compensation table extended with either mill or lathe compensations.

23.1.17 Other Position Information

Number	Symbol	Attribute	Description
#5101...#5120	#_SVERR[n] n=1-50	R	Lag of the axis 1 ... n.
#100251...#100300			
#100651...#100700	#_MIRTP[n] n=1-50	R	Position offset made by handwheel at the axis 1 ... n.
#5181...#5200	#_DIST[n] n=1-50	R	The value of the distance to go at the axis 1 ... n.
#100801...#100850			

Lag of the axes: #5101...#5120, #100251...#100300, #_SVERR[n] (R)
From the variables, the lag of the axis servo loop (servo-follower error) can be read out, in input unit of measurement.

Position offset made by handwheel: #100651...#100700, #_MIRTP[n] (R)
If, in automatic mode, even in the course of motion, position (zero point) of the axes are being corrected using handwheel, the extent of correction can be read out from the variables above for each axis, in input unit of measurement.

The value of the distance to go: #5181...#5200, #100801...#100850, #_DIST[n] (R)
From the variables, the values of the distance to go, i.e. the distance to go to the end position in the block being executed, can be read out for each axis, in input unit of measurement. It is the value the control displays on the screen of the position as distance to go.

23.1.18 Values of the Tool Compensation Memory

Reading or modifying the compensation values when **only mill compensations** are used
The value of the bit #2 TOL of the parameter N1416 Comp. Config on Mills is 0.

Number	Symbol	Attribute	Description
#10001...#10999	#_OFSHG[n] n=1...999	R/W	The geometry value of length compensation. n=1...999: the number of the compensation cell.
#11001...#11999	#_OFSHW[n] n=1...999	R/W	The wear value of length compensation. n=1...999: the number of the compensation cell.
#12001...#12999	#_OFSDG[n] n=1...999	R/W	The geometry value of radius compensation. n=1...999: the number of the compensation cell.
#13001...#13999	#_OFSDW[n] n=1...999	R/W	The wear value of radius compensation. n=1...999: the number of the compensation cell.
#21001...#21999	#_CORR_G[n] n=1...999	R/W	The geometry value of the corner radius. n=1...999: the number of the compensation cell
#22001...#22999	#_CORR_W[n] n=1...999	R/W	The wear value of the corner radius. n=1...999: the number of the compensation cell

Using the macro variables above, value can be assigned to all the compensation cells, and, all the compensation cells can be read out from program.

Reading or modifying the compensation values when **lathe compensations** are also used
The value of the bit #2 TOL of the parameter N1416 Comp. Config on Mills is 1.

Number	Symbol	Attribute	Description
#10001...#10999	#_OFSHG[n], or #_OFSZG[n] n=1...999	R/W	The geometry value of length compensation. The geometry value of length compensation in the direction Z. n=1...999: the number of the compensation cell

Number	Symbol	Attribute	Description
#11001...#11999	#_OFSHW[n], or #_OFSZW[n] n=1...999	R/W	The wear value of length compensation. The wear value of length compensation in the direction Z. n=1...999: the number of the compensation cell
#12001...#12999	#_OFSDG[n], or #_OFSRG[n] n=1...999	R/W	The geometry value of radius compensation. n=1...999: the number of the compensation cell
#13001...#13999	#_OFSDW[n], or #_OFSRW[n] n=1...999	R/W	The wear value of radius compensation. n=1...999: the number of the compensation cell
#21001...#21999	#_CORR_G[n] n=1...999	R/W	The geometry value of the corner radius. n=1...999: the number of the compensation cell
#22001...#22999	#_CORR_W[n] n=1...999	R/W	The wear value of the corner radius. n=1...999: the number of the compensation cell
#23001...#23999	#_OFST[n] n=1...999	R/W	The code of the tool position. n=1...999: the number of the compensation cell
#135001... #135999	#_OFSXW[n] n=1...999	R/W	The wear value of length compensation in the direction X. n=1...999: the number of the compensation cell
#136001... #136999	#_OFSYW[n] n=1...999	R/W	The wear value of length compensation in the direction Y. n=1...999: the number of the compensation cell
#137001... #137999	#_OFSXG[n] n=1...999	R/W	The geometry value of length compensation in the direction X. n=1...999: the number of the compensation cell
#138001... #138999	#_OFSYG[n] n=1...999	R/W	The geometry value of length compensation in the direction Y. n=1...999: the number of the compensation cell

Using the macro variables above, value can be assigned to all the compensation cells, and, all the compensation cells can be read out from program.

23.1.19 Workpiece Zero Point Offsets

Values of zero point offset

Number	Symbol	Attribute	Description
#5201...#5220 #100301...#100350	#_WZCMN[n] n=1-50	R/W	Common workpiece zero point offset for each axis
#5221...#5240 #100351...#100400	#_WZG54[n] n=1-50	R/W	Workpiece zero point offset G54 for each axis
#5241...#5260 #100401...#100450	#_WZG55[n] n=1-50	R/W	Workpiece zero point offset G55 for the axis 1...n
#5261...#5280 #100451...#100500	#_WZG56[n] n=1-50	R/W	Workpiece zero point offset G56 for the axis 1...n
#5281...#5300 #100501...#100550	#_WZG57[n] n=1-50	R/W	Workpiece zero point offset G57 for the axis 1...n
#5301...#5320 #100551...#100600	#_WZG58[n] n=1-50	R/W	Workpiece zero point offset G58 for the axis 1...n
#5321...#5340 #100601...#100650	#_WZG59[n] n=1-50	R/W	Workpiece zero point offset G59 for the axis 1...n

Using the macro variables above, value can be assigned to all the zero point offsets for each axis, and, all the zero point offsets for each axis can be read out from program.

Values of misalignment compensation

Number	Symbol	Attribute	Description
#120001	#_WRG54[1]	R/W	Misalignment compensation G54 in the plane XY
#120002	#_WRG54[2]	R/W	Misalignment compensation G54 in the plane ZX
#120003	#_WRG54[3]	R/W	Misalignment compensation G54 in the plane YZ
#120011	#_WRG55[1]	R/W	Misalignment compensation G55 in the plane XY
#120012	#_WRG55[2]	R/W	Misalignment compensation G55 in the plane ZX
#120013	#_WRG55[3]	R/W	Misalignment compensation G55 in the plane YZ
#120021	#_WRG56[1]	R/W	Misalignment compensation G56 in the plane XY
#120022	#_WRG56[2]	R/W	Misalignment compensation G56 in the plane ZX
#120023	#_WRG56[3]	R/W	Misalignment compensation G56 in the plane YZ
#120031	#_WRG57[1]	R/W	Misalignment compensation G57 in the plane XY
#120032	#_WRG57[2]	R/W	Misalignment compensation G57 in the plane ZX
#120033	#_WRG57[3]	R/W	Misalignment compensation G57 in the plane YZ
#120041	#_WRG58[1]	R/W	Misalignment compensation G58 in the plane XY
#120042	#_WRG58[2]	R/W	Misalignment compensation G58 in the plane ZX
#120043	#_WRG58[3]	R/W	Misalignment compensation G58 in the plane YZ
#120051	#_WRG59[1]	R/W	Misalignment compensation G59 in the plane XY
#120052	#_WRG59[2]	R/W	Misalignment compensation G59 in the plane ZX
#120053	#_WRG59[3]	R/W	Misalignment compensation G59 in the plane YZ

Using the macro variables above, misalignment compensation can be assigned in an arbitrary main plane for each workpiece coordinate system, where the center point of the misalignment compensation is the origin of the coordinate system.

Values of additional zero point offset

Number	Symbol	Attribute	Description
#7001...#7020 #101001...#101050	#_WZP1[n] n=1-50	R/W	Workpiece zero point offset G54.1 P1 for the axis 1...n
#7021...#7040 #101051...#101100	#_WZP2[n] n=1-50	R/W	Workpiece zero point offset G54.1 P2 for the axis 1...n
...	...	...	...

Number	Symbol	Attribute	Description
#7941...#7960 #103351...#103400	#_WZP48[n] n=1-50	R/W	Workpiece zero point offset G54.1 P48 for the axis 1...n
#103401...#103450	#_WZP49[n] n=1-50	R/W	Workpiece zero point offset G54.1 P49 for the axis 1...n
#103451...#103500	#_WZP50[n] n=1-50	R/W	Workpiece zero point offset G54.1 P50 for the axis 1...n
...	...	...	...

Using the macro variables above, value can be assigned to all the zero point offsets for each axis, and, all the zero point offsets for each axis can be read out from program.

Values of additional misalignment compensation

Number	Symbol	Attribute	Description
#120061	#_WRP1[1]	R/W	Misalignment compensation G54.1 P1 in the plane XY
#120062	#_WRP1[2]	R/W	Misalignment compensation G54.1 P1 in the plane ZX
#120063	#_WRP1[3]	R/W	Misalignment compensation G54.1 P1 in the plane YZ
#120071	#_WRP2[1]	R/W	Misalignment compensation G54.1 P2 in the plane XY
#120072	#_WRP2[2]	R/W	Misalignment compensation G54.1 P2 in the plane ZX
#120073	#_WRP2[3]	R/W	Misalignment compensation G54.1 P2 in the plane YZ
...	...	...	...

Using the macro variables above, misalignment compensation can be assigned in an arbitrary main plane for each additional workpiece coordinate system, where the center point of the misalignment compensation is the origin of the coordinate system.

Values of dynamic zero point offset

The values of dynamic zero point offset in the table can be written and read using the following variables:

Number	Symbol	Attribute	Description
#5521...#5540 #117051...#117100	#_FOFS1[n] n=1-50	R/W	Dynamic zero point offset G54.2 P1for each axis
#5541...#5560 #117101...#117150	#_FOFS2[n] n=1-50	R/W	Dynamic zero point offset G54.2 P2 for each axis
...	...		...
#5661...#5680 #117401...#117450	#_FOFS8[n] n=1-50	R/W	Dynamic zero point offset G54.2 P8 for each axis

Number of the actual dynamic zero point offset

While the program runs, the number of the valid dynamic zero point offset can be read out using the instruction below:

Number	Symbol	Attribute	Description
#5500 #117000	#_FOFSP	R	The number of the actual dynamic zero point offset (G54.2 Pp P=1-8)

Value of the actual dynamic zero point offset

While the program runs, the value of the valid dynamic zero point offset can be read out using the commands below. These values are not identical with the values written in the table of the dynamic zero point offset, but they are values modified (rotated) in accordance with the actual position of the work table(s).

Number	Symbol	Attribute	Description
#5501...#5520 #117001...#117050	#_FOFSVAL[n] n=1-50	R	The value of the actual dynamic zero point offset for each axis

23.1.20 Reading out the Tool Data of the Spindle and the Stand-by Magazines

Number	Symbol	Attribute	Description
#8400		R/W	The address of the spindle or the stand-by magazine

The spindle or the tool being in the stand-by magazine the tool management data of which are to be read out using macro variables #8401, #8402, ... can be given in the macro variable

#8400 (10, 11, 20, 21, ... writable, readable).

Only the magazine numbers given in brackets can be defined. If there are several spindles or stand-by magazines on the machine, ask the builder of the machine tool for information.
If there is only one spindle and no stand-by magazine on the machine, value will not have to be assigned to the macro variable #8400 and the macro variables #8401, ... will always be related to the tool being in the spindle.

The data that can be read out are the following:

Number	Symbol	Attribute	Description
#8401		R	Data number (serial number of the tool management table)
#8402		R	Tool type number (code T)
#8403		R	Tool life counter value
#8404		R	Maximum tool life value
#8405		R	Warning tool life value
#8406		R	Tool life status
#8407		R	Customer bit-type data
#8408		R	Tool information
#8409		R	H: number of the length compensation cell (for milling machine channel)
#8410		R	D: number of the radius compensation cell (for milling machine channel)
#8411		R	S: spindle speed
#8412		R	F: feed
#8413		R	G: number of the geometry compensation cell (for lathe channel)
#8414		R	W: number of the wear compensation cell (for lathe channel)
#8431		R	Customer data 1
#8432		R	Customer data 2

Number	Symbol	Attribute	Description
#8433		R	Customer data 3
#8434		R	Customer data 4
#8435		R	Customer data 5
#8436		R	Customer data 6
#8437		R	Customer data 7
#8438		R	Customer data 8
#8439		R	Customer data 9
#8440		R	Customer data 10
#8441		R	Customer data 11
#8442		R	Customer data 12
#8443		R	Customer data 13
#8444		R	Customer data 14
#8445		R	Customer data 15
#8446		R	Customer data 16
#8447		R	Customer data 17
#8448		R	Customer data 18
#8449		R	Customer data 19
#8450		R	Customer data 20

The number of customer data can be given in the parameter N2903 No. of Custom Columns. The first customer data is always a bit-type data.
The macro variables above are readable only.
If the spindle or the stand-by magazine selected by the macro variable #8400 is empty (there is no tool in it), the value of the macro variables will be the following:

#8401=0 (data number);
#8402= #8403= ...= #8450= #0 (empty).

Using the macro variables, compensations (H, D) or technological parameters (F, S) assigned to the tool can be called. If, for example, subprogram call is assigned to the code of tool change (M06), the following can be written in the subprogram:

```
...
M6
#8400=10 (1. Spindle magazine)
H#8409 D#8410 S#8411 F#8412
...
```

23.1.21 Reading the Data of the Pallet Being in the Working Space and in the Loading-Unloading Point

Number	Symbol	Feature	Description
#8500		R/W	Address of the working space of loading-unloading point

It is the macro variable

#8500 (10, 11 writable, readable)

at which it can be given whether the data of the pallet being in the working space (#8500=10) or the data of the pallet being in the loading-unloading point (#8500=11) have to be read out via macro variables #8501, #8502, ... Only the magazine numbers given in bracket can be defined.

If there is only working space and there is no loading-unloading point, value has not to be given to the macro variable #8500; the macro variables #8501, ... apply always to the pallet being in the working space.

After turn-on, the control starts with the initial value #8500=10.

The data that can be read out are as follows:

Number	Symbol	Feature	Description
#8501		R	Pallet identifier
#8502		R	Status
#8503		R	Priority
#8504		R	Number of executed programs
#8505		R	Number of assigned programs
#8531		R	Custom 1
#8532		R	Custom 2
#8533		R	Custom 3
#8534		R	Custom 4

23.2 Instructions of the Program Language

For describing various instructions, the expression

#i = <formula>

is used. The <formula> may include arithmetic operations, functions, variables, constants.
In general, variables #j and #k are referred to in the <formula>.
The <formula> may stand not only in the right side of the assignment statement, but in an NC block, the various addresses may take on a formula too, instead of concrete numerical value or variable.
The order of the operation execution can be influenced by using brackets [,].

23.2.1 Definition or Replacement: #i = #j

The code of this instruction: **=**
Due to this instruction, the variable #I takes on the value of the variable #j, i.e. the value of the variable #j gets into the variable #i.
The assignment

#i=#i<operation>#j

is also permitted. An operation will be executed in the value of the variable #I by the variable #j , and the result of the operation will get into the variable #i.
For example:

```
#100=#100+1
```

means that the new value of the #100 will be its old value plus 1.
Value can be assigned to bit-type variables too:

```
#1100=1
#_UO[3]=0
#_M_SBK=1
```

If the value assigned to the bit-type variable is

$0 < \text{value} \leq 1$,

the value will be taken on as 1. If te value is greater than 1

$1 < \text{value}$,

the error message ‘2092 Value truncation’ will be sent by the control.

☞ ***Warning! Only writable or writable/readable (W or W/R) variable must stand on the left side of the assignment statement.*** If a value is to be assigned to a variable that is readable (R) only, the error message ‘2161 Macro variable #nnnn is read-only’ will be sent by the control.

23.2.2 Arithmetic Operations

Unary minus: #i = - #j

The code of the operation: -
As a result of the operation, the variable #i will be identical with the variable #j in absolute value, but it will have opposite sign.

```
#100=-#12
```

Addition: #i = #j + #k

The code of the operation: **+**
As a result of the operation, the variable #i will take on the sum of the variables #j and #k.

```
#1=#2+3.25
G0 X[#100+#101]
```

Subtraction: #i = #j - #k

The code of the operation: -

As a result of the operation, the variable #i will take on the difference of the variables #j and #k.

```
#100=#100-#102
G1 Z[25.34-2.48]
```

Multiplication: #i = #j * #k

The code of the operation: *

As a result of the operation, the variable #i will take on the product of the variables #j and #k.

```
#3=#1*#2
#100=#101*5.65
```

Division: #i = #j / #k

The code of the operation: /

As a result of the operation, the variable #i will take on the quotient of the variables #j and #k. The value of the variable #k must not be 0, otherwise the error message '2093 Zero divide error' will be sent by the control.

```
#3=#1/#2
#100=542.23/#3
```

Producing remainder: #i = #j MOD #k

The code of the operation: **MOD**

As a result of the operation, the variable #i will take on the division remainder of the variables #j and #k. The value of the variable #k must not be 0, otherwise the error message '2093 Zero divide error' will be sent by the control. In the case of

```
#120=27MOD4
```

the value of the variable #120 will be 3.

Examples of using arithmetic operations

Using the brackets [,], the order of precedence can be influenced.

```
#100 = #101 + #102
#100 = #100 - 3
#100 = [#101 + #102*5.27]/4.1
G0 X[[#100 + 2]/4] Y100
```

23.2.3 Logic Operations

Logical operations always manipulate the variables as 32-bit signed integer numbers. If logical operation is executed with an originally floating-point variable, e.g. with #100, it will be checked by the control whether the value of the variable #i satisfies the following condition:

$-(2^{32} - 1) \leq \#i < 2^{32}$,

or in decimal form

$-4294967297 \leq \#i < 4294967296$.

If not, the error message '2129 Erroneous operation with #' will be sent by the control.
If the value of the variable is within the these boundaries, the operation will be executed with the 32-bit integer number. If the result of the operation gets to an originally floating-point variable, e.g. to #100, the result will be reconverted into floating-point number by the control.

Negation: #i = NOT #j

The code of the operation: **NOT**

As a result of the operation, at first, the variable #j will be converted into 32-bit fixed-point number. Then, the bitwise negated value of this fixed-point number will be taken at all the 32 bits.

Disjunction: #i = #j OR #k

The code of the operation: **OR**

As a result of the operation, at first the variables #j and #k will be converted into 32-bit fixed-point number.

As a result of the operation, the logic sum of the bitwise values of the variables #j and #k will get to the variable #i, at all the 32 bits. Where there is 0 at the same positional values of the two numbers in both places, at this positional value in the result there will be 0, otherwise 1.

Exclusive or: #i = #j XOR #k

The code of the operation: **XOR**

As a result of the operation, at first the variables #j and #k will be converted into 32-bit fixed-point number.

As a result of the operation, the bitwise values of the variables #j and #k will be summed into the variable #i in such a way so that where there are same numerical values at same positional values, at this positional value in the result there will be 0, but where there are dissimilar numerical values at same positional values, at this positional value in the result there will be 1, at all the 32 bits.

Conjunction: i# = #j AND #k

The code of the operation: **AND**

As a result of the operation, at first the variables #j and #k will be converted into 32-bit fixed-point number.

As a result of the operation, the logic product of the bitwise values of the variables #j and #k will get to the variable #i, at all the 32 bits. Where there is 1 at the same positional values of the two numbers in both places, at this positional value in the result there will be 1, otherwise 0.

Examples of using logic operations

Let us manipulate the 32-bit macro variable #1132 sent to PLC in the following way: the upper 8 bits (31-24) of the variable are to be set in such a way so that the lower bits remain unchanged. To leave the lower 24 bits unchanged and to delete the 8 upper 8 bits, let us have the following bit mask:

The hexadecimal format of the bit mask is:

```
00FFFFFF,
```

and having converted into decimal format, it will be:

```
16777215
```

Let us store the bit mask in the variable #100:

```
#100=16777215
```

Let as store in the variable #100 the value leaving the lower 24 bits of the macro variable #1132 unchanged but deleted at the upper 8 bits of it:

```
#101=#100AND#1132
```

Let us write on the variable #102 the bit mask to be set:

The hexadecimal format of the bit mask is:

```
9B000000,
```

and having converted into decimal format, it will be:

```
-1694498816
```

Let us store the bit mask in the variable #102:

```
#102= -1694498816
```

Then, let us write it on the variable #1132 in such a way so that the lower 24 bits remain unchanged:

```
#1132=#101XOR#102
```

Using brackets accordingly, the following can also be written instead of the four rows above:

```
#1132=[16777215AND#1132]XOR[-1694498816]
```

23.2.4 Functions

Square root: #i = SQRT #j

The code of the function: **SQRT**

As a result of the operation, the variable #i will take on the square root of the variable #j.

```
#101=SQRT[#100+4]
```

The value of the variable #j must be only a positive number or 0:

$\#j \geq 0$

If the argument is negative, the error message '2122 Square root from negative value' will be sent by the control.

Sine: #i = SIN #j

The code of the function: **SIN**

As a result of the operation, the variable #i will take on the sine of the variable #j. ***The value of the #j is to be always interpreted in degree***.

```
G1 X[SIN30-#101]
```

Cosine: #i = COS #j

The code of the function: **COS**

As a result of the operation, the variable #i will take on the cosine of the variable #j. ***The value of the #j is to be always interpreted in degree***.

```
#102=COS[#1+#2]
```

Tangent: #i = TAN #j

The code of the function: **TAN**

As a result of the operation, the variable #i will take on the tangent of the variable #j. ***The value of the #j is to be always interpreted in degree***.

If the value of the argument #j is

$\#j=(2n+1)*90°$, where n=0, ±1, ±2,...

The error message '2116 Absolute value of the argument is not less then 90' will be sent by the control.

```
#1=TAN[#2*15.6]
```

Arc sine: #i = ASIN #j

The code of the function: **ASIN**

As a result of the operation, the variable #i will take on the arc sine of the variable #j. ***The result will be given in degree, the value of the variable #i will be between +90° and -90°***.

```
#101=ASIN[-0.5] (it will be #101=-30)
```

The argument of the variable #j must satisfy the following condition:

$-1 \leq \#j \leq 1$

Otherwise, the error message '2117 Absolute value of the argument is greater then 1' will be sent by the control.

Arc cosine: #i = ACOS #j

The code of the function: **ACOS**

As a result of the operation, the variable #i will take on the arc cosine of the variable #j. ***The result will be given in degree, the value of the variable #i will be between 0° and 180°***.

```
#101=ACOS[-0.5] (it will be #101=120)
```

The argument of the variable #j must satisfy the following condition:

$-1 \leq \#j \leq 1$

Otherwise, the error message '2117 Absolute value of the argument is greater then 1' will be sent by the control.

Arc tangent: #i = ATAN #j

The code of the function: **ATAN**

As a result of the operation, the variable #i will take on the arc tangent of the variable #j. ***The result will be given in degree, the value of the variable #i will be between +90° and -90°***.

```
#101=ATAN[-0.5] (it will be #101=-26.565)
```

Exponential with base e: #i = EXP #j

The code of the function: **EXP**

As a result of the operation, the variable #i will take on the #jth power of the exponential growth constant e.

```
#100=EXP1 (it will be #100=2.71828... i.e."e")
```

Natural logarithm: #i = LN #j

The code of the function: **LN**

As a result of the operation, the variable #i will take on the logarithm to the base of the mathematical constant e of the number #j.

```
#100=LN2.718281828 (it will be #100=0.999... ln(e)=1)
```

If the value of the #j is

$\#j \leq 0$,

the error message '2118 Value of the argument is not positive' will be sent by the control.

23.2.5 Conversion Instruction

Generating the absolute value: #i = ABS #j

The code of the function: **ABS**

As a result of the operation, the variable #i will take on the absolute value of the variable #j.

```
#100=ABS[-3.1] (it will be #100=3.1)
#100=ABS[5.25] (it will be #100=5.25)
#100=ABS[0] (it will be #100=0)
```

Rounding down to the absolute value: #i = FIX #j

The code of the function: **FIX**

The fractional part of the variable #j is discarded by the operation, and the variable #i will have the remaining value.

For example:

```
#130=FIX4.8 (it will be #130=4)
#131=FIX[-6.7] (it will be #131=-6)
```

Rounding up to the absolute value: #i = FUP #j

The code of the function: **FUP**

The fractional part of the variable #j is discarded by the operation, and 1 will be added to the absolute value.

For example:

```
#130=FUP12.1 (it will be #130=13)
#131=FUP[-7.3] (it will be #131=-8)
```

23.2.6 Execution Sequence of Complex Arithmetic Operations

The arithmetic operations and the functions listed above can be combined. The ***execution sequence of operations*** or the order of precedence is as follows:

function – multiplicative arithmetic operations – additive arithmetic operations .

For instance, in the example below the execution sequence of the operations is as follows:

```
#110 = #111 + #112 * COS #113
                     ———1———
               ————————2————————
          ———————————3———————————
```

Modifying the execution sequence of operations

Using brackets[and], the execution sequence of operations can be modified.

An example of a bracketing of three-level depth is illustrated below:

```
#120 = COS [ [ [#121 - #122] * #123 + #125] * #126]
                  ———1———
                  ———————2———————
                  ———————————3———————————
             ————————————————4————————————————
        ————————————————————————5————————————————
```

The numbers indicate the execution sequence of operations. It can be seen that the abovementioned order of precedence apply to execute sequence of operations within the brackets of the same level.

The opening bracket [and the] closing bracket have to be used in pair. If the opening brackets [is less than the closing brackets], the error message ‘2064 Syntax error’ will be sent by the control. If the closing brackets]is less than the opening brackets [, the error message ‘2121Erroneous terminator #’ will be sent by the control.

23.2.7 Conditional Expressions

The conditional expressions known by the program language are the following:

equal to (=):	#i **EQ** #j
not equal to (≠):	#i **NE** #j
greater than (>):	#i **GT** #j
less than (<):	#i **LT** #j

greater than or equal to (≥): #i **GE** #j

smaller than or equal to (≤): #i **LE** #j

The variables on the both sides of a conditional expression can be substituted by a formula too. The conditional expressions above can stand after the instructions IF or WHILE.

23.2.8 Unconditional Branch: GOTOn

Due to the instruction **GOTO**n, execution of the program will resume without any condition in the same program, at the block with the sequence number n. The sequence number n can also be replaced by a variable or a formula. ***The sequence number of the block*** to which jumping is to be intended ***must stand at the beginning of the block***.
If the specified block sequence number is not found, the error message '2125 Block not found Nnnnn' will be sent by the control.
Jump can be made either forward:

```
...
GOTO160 (jumping to the block N160)
...
N160 G0 X100
...
```

or backward. In this case an infinite cycle can also be produced:

```
...
N160 G0 X100
...
#1=100
#2=60
...
GOTO[#1+#2] (jumping to the block N160)
...
```

23.2.9 Conditional Branch: IF[<conditional expression>] GOTOn

If [<conditional expression>] put into brackets compulsorily ***is satisfied***, execution of ***the program will resume in the same program, at the block with the sequence number n.***
If [<conditional expression>] is not satisfied, ***execution of the program will resume at the succeeding block.***
If the IF is not followed by condition check, the error message '2064 Syntax error' or '2121 Erroneous terminator=' will be sent by the control, depending on the type of the error.
For example:

```
...
#100=52.28
#101=16.87
...
IF[#100GT#101] GOTO210 (jumping to the block N210)
...
N210 G0 X0 Y100 Z20
...
```

23.2.10 Conditional Branch: IF[<conditional expression>] (THEN)<instruction>

If [<conditional expression>] is satisfied, the instruction following the THEN will be executed.
If [<conditional expression>] is not satisfied**, **execution of the program will resume at the succeeding block.
The word THEN can be omitted in the instruction, execution of the instruction line

IF[<conditional expression>] instruction

will be the same.

```
...
#100=0
#101=0
#1=20
...
IF[#1EQ20] THEN#100=3.15 (it will be #100=3.15)
IF[#100GT3.15] #101=5 (it will remain #101=0))
...
```

23.2.11 Iteration: WHILE[<conditional expression>] DOm ... ENDm

As long as [<conditional expression>] is satisfied, the blocks from Dom to ENDm will be executed repeatedly. I.e. the control checks whether the condition is satisfied: if yes, the program part between Dom and ENDm will be executed and then, due to the instruction ENDm, the program returns to checking once again the condition following the WHILE.
***If [<conditional expression>] is not satisfied**, **execution of the program will resume at the block following ENDm**.*
If WHILE [<conditional expression>] is omitted, i.e. the cycle is described by the instructions DOm ... ENDm***, the program part between Dom and ENDm will be executed for infinite period of time.***

***The possible values of m: 1, 2, 3**.*

If m>3, the error message ‘2002 DO data out of range’ will be sent by the control.
If not a condition check but an assignment follows the instruction WHILE, the error message ‘2121 Erroneous terminator=’ will be sent by the control.
If, in the same block, the instruction WHILE is not followed by DO, the error message ‘2110 WHILE without DO’ will be sent by the control.
If there is no Dom before the instruction ENDm, the error message ‘2125 Block not found: Dom’ will be sent by the control.

Let us produce a hollow using the following data: depth in the direction Z is 5.4 mm; width in the direction X is 21 mm; the zero point located at the center of the hollow in the direction X and at the top of the hollow in the direction Z. Let depth of cut be 0.2 mm. The task can be solved using cycle organization:

```
#1=0       (Target position in the direction Z)
#2=10           (Target position in the direction X)
G0 X#2          (Motion to the starting point in the
                direction X)
Z[#1+1]         (Motion to the starting point in the
                direction Z)
WHILE[#1GT-5.39] DO1     (Cycle until depth of 5.4 mm is
                         reached in the direction Z)
```

```
#1=#1-0.2 (Depth of cut of 0.2 mm in the direction Z)
#2=-#2          (Reversing the motion direction X)
G1 Z#1 F100     (Infeed in the direction Z)
X#2 F500        (Milling in the direction X)
END1
G0 Z20
...
```

The rules of cycle organization are as follows:

– The instruction Dom has to be specified before the instruction ENDm:

```
:
END1
:
:                       WRONG
:
DO1
```

In the case above, the error message ‘2125 Block not found: Dom’ will be sent by the control at the block END1.

– The instructions DOm and ENDm has to stand in pair:

```
:
DO1
:
DO1                     WRONG
:
END1
:
```

or

```
:
DO1
:
END1                    WRONG
:
END1
:
```

– It is allowed to use the same identification number several times:

```
:
DO1
:
END1
:
:                       RIGHT
:
DO1
:
END1
:
```

– The pairs DOm ... ENDm can be nested into each other up to three levels at most:

```
:
DO1
:
DO2
:
DO3
:
:                       RIGHT
:
END3
:
END2
:
END1
:
```

– The pairs DOm ... ENDm must not overlap each other:

```
:
DO1
:
DO2
:
:                       WRONG
:
END1
:
END2
```

– It is allowed to branch from inside the cycle to outside the cycle:

```
:
DO1
:
GOTO150
:
:                       WRONG
:
END1
:
N150
:
```

– It is allowed to enter the cycle from outside:

```
:
GOTO150
:
DO1
:
:
:
N150
:
END1
:
```

or

```
:
DO1
:
N150
:
:
:
END1
:
GOTO150
:
```

– Calling subprograms or macros from inside the cycle is possible. Inside the subprogram or the custom macros, the cycles can again be nested into each other up to three levels:

```
:
DO1
:
M98...              RIGHT
:
G65...              RIGHT
:
G66...              RIGHT
:
G67...              RIGHT
:
END1
:
```

23.2.12 Indirect Axis Address Specification

The instruction

#i=AXNUM[<axis address>]

makes it possible to query the number of an axis in a given channel on the basis of the address of that axis. If it is intended to query the axis number for a non-existent axis address, the error message ‘2017 Illegal address <axis name>’ will be sent by the control.

Using the instruction

AX[<axis number>]=

a instruction can be issued indirectly, i.e. not by the address of the axis, but by the number of the axis. If there is no the axis of the specified number in the channel, the error message '2018 Bad physical axis number: n' will be sent by the control.
For example, instead of the block

```
G0 X10 Y20 Z30
```

the following instruction line can be written:

```
#101=AXNUM[X]        (Querying the number of the axis X)
#102=AXNUM[Y]        (Querying the number of the axis Y)
#103=AXNUM[Z]        (Querying the number of the axis Z)
G0 AX[#101]=10 AX[#102]=20 AX[#103]=30
```

☞ ***Warning!*** If multi-character axis addresses are used, the axis address will not have to be AX or AXN.

23.2.13 Data Output Instructions

The following data output instructions are known by the control:

POPEN	opening a peripheral
FOPEN	opening a file
BPRNT	binary data output
DPRNT	decimal data output
PCLOS	closing a peripheral
FCLOS	closing a file

These data output instructions can be used to output characters and variable values. Outputting occurs in the memory of the control. They can be used, for example, for storing measurement results, logging etc.

Opening a peripheral: POPENn
Prior to issuing a data output instruction, it is necessary to open the appropriate peripheral through which data output will be executed. The number Number is used for selecting the appropriate peripheral:

n = 31 the memory of the control

On opening the peripheral, one character % is output to the peripheral so each data output begins with one character %.
After the instruction POPEN, a filename has to be output by the instruction DPRNT. If the file already exists, the old one will be overwritten without asking; if it does not exist yet, a new one will be open. The file gets to the folder where the program runs and takes on the extension of the program.
For example:

```
...
#100=4567
POPEN31
DPRNT[O#100[4]]      (The filename will be O4567)
...
```

or

```
...
POPEN31
DPRNT[ABC]           (The filename will be ABC)
...
```

Closing a peripheral: PCLOSn
The peripheral opened by the instruction POPEN has to be closed by the instruction PCLOS. After the instruction PCLOS it is necessary to give the number of the peripheral to be closed. In our case:

```
...
PCLOS31
...
```

On closing, one character % is also output to the peripheral, i.e. each data output is closed with one character %.

Opening a file: FOPENn Ppppp, FOPENn <filename>
Prior to issuing a data output instruction, it is necessary to open a file where the data are to be written.

Opening a file by program number

The instruction

FOPENn P(program number)

opens the file the number of which is given at the address P (program number).
The rules of programming the program number given at the address P are dealt with in the subsection **14.4.1 Identification of Programs in Memory. The Program Number (O)** on page 142. ***The rules of extension and in-memory location of the files opened by program number*** are the same as those of subprograms. See the subsection **14.4.2 Calling a Subprogram (M98)** on page 142.
For example:

```
...
FOPEN1 P12          (The filename will be O0012)
...
```

Opening a file by filename

The instruction

FOPENn <filename>

opens the file the name of which is given between the symbols < and >.
The rules of filename specification are dealt with in the subsection **14.4.1 Identification of Programs in Memory. The Program Number (O)** on page 142. ***The rules of extension and in-memory location of the files opened by filename*** are the same as those of subprograms. See the subsection **14.4.2 Calling a Subprogram (M98)** on page 142.
For example:

```
...
FOPEN1 <data.txt>   (The filename will be data.txt)
...
```

Types of opening a file

The type of opening a file can be given by the number written at the place of 'n':

n = 1: Creating a new file

If the file already exists, the error message '2144 The file (Ooooo) already exists' or '2146 The given file already exists' will be sent by the system, and the program run does not continue.

n = 2: Creating a new file in any case

If the file does not exist, it will be created; if the file exists, it will be open and its content will be deleted. The contents output by the instruction BPRNT or DPRNT will be written from the beginning of the file.

n = 3: Opening a file for attachment

If the file does not exist, the error message '2147 The file (Ooooo) not found' or '2132 The file not found' will be sent by the system, and the program run does not continue. If the file already exists, it will be open and, ***beginning from the end of the file***, the contents output by the instruction BPRNT or DPRNT ***will be attached***.

n = 4: Creating a file or opening it for attachment

If the file ***does not exist*** yet, it ***will be created***; ***if*** the file already ***exists***, it ***will be open*** and, ***beginning from the end of the file***, the contents output by the instruction BPRNT or DPRNT ***will be attached***.

n = 5: Opening a file together with content deletion

If the file does not exist, the error message '2147 The file (Ooooo) not found' or '2132 The file not found' will be sent by the system, and the program run does not continue. If the file already exists, ***it will be open and its content will be deleted***. The contents output by the instruction BPRNT or DPRNT will be written from the beginning of the file.

Closing a file: FCLOS

The opened file can be closed by the instruction FCLOS. For writing, one file can be created or opened simultaneously.

Binary data output: BPRNT[...]

The format of the instruction BPRNT is as follows:

```
BPRNT[ a #b [c] ... ]
       |  |  |
       |  |  └──────── number of digits after the decimal point
       |  └─────────── variable
       └──────────────── character
```

The instruction sends characters in ASCII code, variables in binary format.

– Characters that can be sent are the following:
  alphabetical characters: A, B, ..., Z
  numerical characters: 1, 2, ..., 0
  special characters: *, /, +, –
  Instead of the character * (asterisk), the code of the space will be sent by the control.

– Values of the variables are output by the control at 4 bytes, i.e. at 32 bits, beginning from the most significant byte. After the number of variables, the number of digits after the decimal point has to be given in brackets [ ]. In this case, the control converts the floating-point value of the variable into such fixed-point value in which the number of significant decimal digits will be the value given in brackets [ ]. The possible values of c are: 1, 2, ..., 8. For example, if
  #120 = 258.647673 és [3] ——— 258648=0003F258h will be output.

– A vacant variable will be output with binary code 00000000h

– At the end of data output, one character of carriage return (CR) and one character of line

feed (LF) will automatically be output by the control.

An example:

```
#110=318.49362
#120=0.723415
#112=23.9
BPRNT[C*/X#110[3]Y#120[3]M#112[0]]
```

After rounding and hexadecimal conversion, the values of the variables #110, #120 and #112 will be the following:

#110=318.49362	————	318494=0004DC1Eh
#120=0.723415	————	723=000002D3h
#112=23.9	————	24=00000018h

Characters to be output are the following:

```
7 6 5 4 3 2 1 0

0 1 0 0 0 0 1 1     C
0 0 1 0 0 0 0 0     Space
0 0 1 0 1 1 1 1     /
0 1 0 1 1 0 0 0     X
0 0 0 0 0 0 0 0     00
0 0 0 0 0 1 0 0     04
1 1 0 1 1 1 0 0     DC
0 0 0 1 1 1 1 0     1E
0 1 0 1 1 0 0 1     Y
0 0 0 0 0 0 0 0     00
0 0 0 0 0 0 0 0     00
0 0 0 0 0 0 1 0     02
1 1 0 1 0 0 1 1     D3
0 1 0 0 1 1 0 1     M
0 0 0 0 0 0 0 0     00
0 0 0 0 0 0 0 0     00
0 0 0 0 0 0 0 0     00
0 0 0 1 1 0 0 0     18
0 0 0 0 1 1 0 1     Carriage Return
0 0 0 0 1 0 1 0     Line Feed
```

Decimal data output: DPRNT[...]

The format of the instruction DPRNT is as follows:

```
DPRNT[ a #b [c d] ... ]
       |   |  |  |
       |   |  |  └─ number of digits after the decimal point
       |   |  └──── number of digits before the decimal point
       |   └─────── variable
       └─────────── character
```

All the characters and numbers will be output in ASCII code.

– See the instruction **BPRNT** for the rules applied to outputting the characters.
– For outputting the values of the variables, the number of decimal integers and fractions in which the variable is to be output has to be given. The numeral digits have to be given in brackets [ ].
For specification of numerical digits, the condition $0 < c + d < 16$ has to be specified.
Outputting the numbers begins from their highest place value.
– When number is output, the negative sign (–) will also be output.
– If the decimal point is defined ($d > 0$), all the zeros, including the following zeros too, will be output together with the decimal point (.).
– If d=0, or d is not given, neither decimal point, nor zero will be output by the control.
– If the bit #0 PNT of the parameter N1757 Print Contr is
=0, the code of space will be output in the positions of the sign + and the zeros;
=1, the sign + and the leading will not be output.
– A vacant variable will be output with code 0.
– At the end of data output, one character of carriage return (CR) and one character of line feed (LF) will automatically be output by the control.

An example:

```
#130=35.897421
#500=-150.8
#10=14.8
DPRNT[X#130[53]Y#500[53]T#10[20]
```

After rounding, the values of the variables #130, #500 and #10 will be the following:

#130=35.897421 ———— 35.897
#500=–150.8 ———— 150.8
#10=14.8 ———— 15

Data output at the state #0 PNT=0 of the parameterN1757 Print Contr:

```
7 6 5 4 3 2 1 0

0 1 0 1 1 0 0 0     X
0 0 1 0 0 0 0 0     Space
0 0 1 0 0 0 0 0     Space
0 0 1 0 0 0 0 0     Space
0 0 1 0 0 0 0 0     Space
0 0 1 1 0 0 1 1     3
0 0 1 1 0 1 0 1     5
0 0 1 0 1 1 1 0     Decimal point (.)
0 0 1 1 1 0 0 0     8
0 0 1 1 1 0 0 1     9
0 0 1 1 0 1 1 1     7
0 1 0 1 1 0 0 1     Y
0 0 1 0 1 1 0 1     Negative sign (-)
0 0 1 0 0 0 0 0     Space
0 0 1 0 0 0 0 0     Space
0 0 1 1 0 0 0 1     1
0 0 1 1 0 1 0 1     5
0 0 1 1 0 0 0 0     0
0 0 1 0 1 1 1 0     Decimal point (.)
0 0 1 1 1 0 0 0     8
0 0 1 1 0 0 0 0     0
0 0 1 1 0 0 0 0     0
0 1 0 1 0 1 0 0     T
0 0 1 0 0 0 0 0     Space
0 0 1 1 0 0 0 1     1
0 0 1 1 0 1 0 1     5
0 0 0 0 1 1 0 1     Carriage Return
0 0 0 0 1 0 1 0     Line Feed
```

Data output at the state #0 PNT=1 of the parameterN1757 Print Contr:

```
7 6 5 4 3 2 1 0

0 1 0 1 1 0 0 0     X
0 0 1 1 0 0 1 1     3
0 0 1 1 0 1 0 1     5
0 0 1 0 1 1 1 0     Decimal point (.)
0 0 1 1 1 0 0 0     8
0 0 1 1 1 0 0 1     9
0 0 1 1 0 1 1 1     7
0 1 0 1 1 0 0 1     Y
0 0 1 0 1 1 0 1     Negative sign (-)
0 0 1 1 0 0 0 1     1
0 0 1 1 0 1 0 1     5
0 0 1 1 0 0 0 0     0
0 0 1 0 1 1 1 0     Decimal point (.)
0 0 1 1 1 0 0 0     8
0 0 1 1 0 0 0 0     0
```

```
0 0 1 1 0 0 0 0     0
0 1 0 1 0 1 0 0     T
0 0 1 1 0 0 0 1     1
0 0 1 1 0 1 0 1     5
0 0 0 0 1 1 0 1     Carriage Return
0 0 0 0 1 0 1 0     Line Feed
```

☞ ***Notes***:

– The sequence of data output instructions is fixed: at first, the appropriate file has to be open by the instruction POPEN or FOPEN, then data output can be executed by the instruction BPRNT or DPRNT, and finally, the opened file has to be closed by the instruction PCLOS or FCLOS.

– Opening and closing the file can be given in any point of the program. For example, the file can be open at the beginning of the program and can be closed at the program end, while data can be output in any part of the program between these two instructions.

– The instruction M30 or M2 executed during data output will interrupt the data transfer. To avoid this, prior to execution of the instruction M30 it is necessary to wait during data transfer.

23.3 Calling the Macros, System Macros and System Subprograms

Macro call is similar to subprogram call with the difference that ***macros***, contrary to subprograms, ***can have input variables***, i.e. ***arguments***.
Both macros and subprograms can be called either by their program number or by their filename.

Those macros and subprograms are called ***system macros*** and ***system subprograms*** the call of which is related to an ***address assigned in the parameter*** (for example G, M etc.). ***Specific rules apply to filenames of system macros and system subprograms and to their location in the memory***.

A macro call can be multi-level too. A macro can be called from a subprogram, and a subprogram can be called from a macro. The ***maximum*** aggregated level of calling subprograms and macros is ***16***.

Return from a macro occurs ***by the code***
M99.

One-shot and modal macro calls are distinguished. The modal macro calls can be deleted by the code
G67.

Argument assignment
Arguments can be assigned to a macro program. ***Arguments are specific numerical values assigned to definite addresses that, during macro call, will be stored in the appropriate local variables (#1, #2, ..., #33)***. A macro program (special subprogram) can use these local variables, i.e. a macro call is such a special subprogram call, in which the main program can transfer variables (parameters) to the subprogram.
There are two possible argument assignments:

Argument assignment No.1
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
In the case of argument assignment No.1, the parameters are transferred to the macro by using the letters of the English alphabet. Some of the letters above (***G, P, L***) can be monopolized by certain calls for specific purpose; in that case, these letters cannot be used for parameter transfer. Filling the addresses can be made in arbitrary order; it is not necessary to write them into the calling block in alphabetical order.

Argument assignment No.2
A B C I1 J1 K1 I2 J2 K2 ... I10 J10 K10
In the case of argument assignment No.2, the arguments are transferred to the macro by using the addresses I, J and K, in addition to the addresses A, B and C. At the addresses A, B and C, maximum 10 different argument group can be assigned. If, in a block, several arguments are assigned at the same address, the variables will take on the appropriate value in the order of assignment.

Relations between the local variables and the arguments are as follows:

l v	*1. a a*	*2. a a*
#1	A	A
#2	B	B
#3	C	C
#4	I	I1
#5	J	J1
#6	K	K1
#7	D	I2
#8	E	J2
#9	F	K2
#10	G	I3
#11	H	J3

l v	*1. a a*	*2. a a*
#12	L	K3
#13	M	I4
#14	N	J4
#15	O	K4
#16	P	I5
#17	Q	J5
#18	R	K5
#19	S	I6
#20	T	J6
#21	U	K6
#22	V	I7

l v	*1. a a*	*2. a a*
#23	W	J7
#24	X	K7
#25	Y	I8
#26	Z	J8
#27	–	K8
#28	–	I9
#29	–	J9
#30	–	K9
#31	–	I10
#32	–	J10
#33	–	K10

where: *lv*: local variable
1.a a: argument assignment No.1,
2.a a: argument assignment No.2.
The indexes following the addresses I, J and K show the sequence of argument assignment. Decimal point and sign can also be transferred at the addresses.

Handling the address N
The ***first address N*** written in the block is interpreted as ***block number***, the ***second address N***, however, ***will be recorded*** in the local variable #14 as ***argument***:

```
/N130 X12.3 Y32.6 N250
      N130: block number
      #24=12.3
      #25=32.6
      #14=250 (argument)
```

If the address N had already been recorded as argument, the succeeding reference to the address N will result in the error message ‘2017Illegal address N’.

Mixed argument assignment
The argument assignment No.1 and the argument assignment No.2 can exist together within a block, the control accepts it. Error message will be sent in the case when a variable of a given number is referred twice. For example:

```
A2.12 B3.213 J36.9 J-12 E129.73 P2200
```

The local variables take on the address values in the following sequence:

```
A: #1=2.12
B: #2=3.213
J: #5=36.9 (first J)
J: #8=-12 (second J)
E: #8=ERROR, the variable #8 had already been filled
```

In this example, the second address J (with the value –12) assigned a value to the variable #8. Since the value of the address E will also be taken on by the variable #8, the error message

‘2017Illegal address E’ will be sent by the control.
But, if the sequence of assigning the second address J and the address E is interchanged,

```
A2.12 B3.213 J36.9 E129.73 J-12 P2200
```

error message will not be sent, and the value of the address J will be written in the variable #11 of the succeeding J:

```
A: #1=2.12
B: #2=3.213
J: #5=36.9 (first J)
E: #8=129.73
J: #11=-12 (it gets to the variable of the third J
     because the variable #8 is already occupied)
```

☞ ***Warning! Only one-character addresses can be used for argument assignment***, it means that ,C ,R ,A and multi-character axis addresses and spindle addresses not!

Argumentum levels
A macro call, like a subprogram call, ***can also be multi-level***, but the maximum aggregated level of calling subprograms and macros can be 16.
Therefore, the local variables #1 ... #33 are multi-level too. The level of the local variables belonging to the main program is 0 (zero level), then macro calls fill the levels 1, 2 etc. in sequence, up to the maximum level 16.
A subprogram or calling the system subprogram does not change the level of local variables.
After return from macro call, the local variables of the given level will be eliminated, they will be deleted to empty by the #0. ***The local variables of the main program*** will be eliminated ***at the end of the program***.

Macro call by program number

By programming

P(program number)

the macro program with the number defined at the address P (program number) will be called. ***The rules of programming the program number*** given at the address P are dealt with in the subsection **14.4.1 Identification of Programs in Memory. The Program Number (O)** on page 142. ***The rules of extension and in-memory location of the macros defined by program number*** are the same as those of subprograms. See the subsection **14.4.2 Calling a Subprogram (M98)** on page 142.

Macro call by filename

☞ ***Warning! Only G65-type*** non-modal ***macros can be called*** by filename, G66-type and G66.1-type modal macros not!

By programming

<filename>

the macro program with the filename given between the symbols < and > will be called. ***The rules of filename specification*** are dealt with in the subsection **14.4.1 Identification of Programs in Memory. The Program Number (O)** on page 142. ***The rules of extension and in-memory location of the macros defined by filename*** are the same as those of subprograms. See the subsection **14.4.2 Calling a Subprogram (M98)** on page 142.

Calling the system macros and subprograms and their in-memory location

System macros and subprograms ***can only be called at the address specified in parameter***. The rules applied to the storage of system macros and subprograms in memory are stricter than those of macros and subprograms.

In the memory, the system macros and subprograms have to be stored ***in separate folder for each channel***, namely, ***in the folder SystemMacros of the directory Programs***:

..\Programs\SystemMacros\Channel1\
..\Programs\SystemMacros\Channel2\

..\Programs\SystemMacros\Channel8\

Their filename has to begin with the letter O which has to be followed by ***four decimal digits. The extension of the filename has to be .nct.*** For example***:***

O9010.nct

23.3.1 Simple Macro Call (G65)

Due to the instructions

G65 P(program number) L(repetition number) <argument assignment>
G65 <filename> L(repetition number) <argument assignment>

the macro program with the number defined at the address P (program number) or with the filename given between the symbols < and > will be called sequentially and repeatedly; the repetition number is given at the address L.

It is not a modal call.

Rules of argument assignment in the case of call G65

Both forms of argument assignment can be used.

In the case of the call G65, the addresses G, P and L cannot be used for argument transfer, but the values specified will be written in the appropriate local variable.

Multiple call G65

If macro is called from macro again, the level of local variables will grow in accordance with the level of macro.

```
Main program        Macro     Macro     Macro     Macro
Level 0             Level 1   Level 2   Level 3   Level 4
                    O_____    O_____    O_____    O_____


G65 P               G65 P     G65 P     G65 P


                    M99       M99       M99       M99

Local variables
Level 0             Level 1   Level 2   Level 3   Level 4
#1                  #1        #1        #1        #1
                    :         :         :         :
#33                 #33       #33       #33       #33
```

When the first macro is called, the local variables of the main program from #1 to #33 will be stored, and, at the level 1, the local variables will take on the argument values specified on

call. In the case of a further macro call from the level 1, the local variables of the level 1 from #1 to #33 will be stored, and, at the level 2, the local variables will take on the argument values specified on call. In the case of multiple call, the local variables of the previous level will be stored, and, at the succeeding level, the local variables will take on the argument values specified on call. In the case of M99, when return from the called macro to the calling program occurs, the stored local variables of the previous level will be reset to the state in which they were stored on call.

A hole pattern is to be produced using call G65. The code of drilling is to be specified at the address 'A', the other addresses have to be filled in the way used in the case of drilling cycles.

Let the ***main program*** be the following:

```
G54 G17 X0 Y0 Z20
...
G65 P300 A81 Z-2 R2 F300 S500 M3 (center punching by G81)
G65 P300 A83 Z-30 R2 Q6 E1 F100 S1000 (drilling by G83)
G65 P300 A84.2 Z-30 R2 S1000 F1000 (thread tapping by
G84.2)
G0 Z20
...
```

In the macro O0300, the arguments of the call will be received by the local variables, i.e. data related to the spindle:

speed S#19
direction of rotation M#13;

input parameters of drilling:

drilling cycle code A#1
hole depth Z#26
coordinate of the point R R#18
feedrate F#9
infeed Q#17
distance to the point of approaching E#8.

Then, listing the coordinates X, Y of the hole follows.

The body of the ***macro O0300*** executing the drilling is as follows:

```
S#19 M#13                        (Starting the spindle)
G#1 X0 Y0 Z#26 R#18 F#9 Q#17 E#8 (Setting the drilling
cycle)
X100
Y100
X0
X50 Y50
G80
M99
```

23.3.2 Macro Modal Call after Each Motion Command (G66)

Due to the instruction

G66 P(program number) L(repetition number) <argument assignment>

the ***macro program*** with the number defined at the address P (program number) ***will be called***

sequentially and repeatedly ***after execution of each motion command***; the repetition number is given at the address L.
It is a modal call.
The assigned macro will be called until the instruction

G67

is issued to cancel the modal call.

Rules of argument assignment in the case of call G66

Both forms of argument assignment can be used.
In the case of the call G66, the addresses G, P and L cannot be used for argument transfer, but the values specified will be written in the appropriate local variable.

An example:
In a given segment of the part program, a hole has to be produced after each motion:
Main program

```
...
G66 P1250 Z-100 R-1 X2 F130 (Z: the bottom point of the
     hole, R: the point R of the hole,  X: dwell time, F:
     feedrate)
N100 G91 G0 X100
N110 Y30
...
N180 X150
G67
```

From the block N110 up to and including the block N180, the macro program O1250 will be called at the end of positioning, and the drilling operation written there will be executed with the input parameters specified in the block G66.

Macro program

```
O1250
G0 Z#18          (Rapid traverse positioning in the
                 direction Z to the position -1 given at the
                 address R)
G1 Z#26 F#9      (Drilling to the bottom point -100 given at
                 the address Z, at the feedrate 130 mm/min
                 given at the address F)
G4 P#24          (Dwell at the bottom of the hole for 2 s
                 given at the address X)
G0 Z-[#18+#26](Retracting the tool to the initial point)
M99              (Return to the calling program)
```

The state G91 will be inherited from the block N100 of the main program by the macro program O1250, for this reason, each positioning will be executed incrementally.

Multiple call G66

In the case of multiple call of G66-type macros, after execution of each motion block, ***at first the macro that was called last will be called, and the macros that were called previously will be called from this macro, in reverse order***. Let us see the example below:

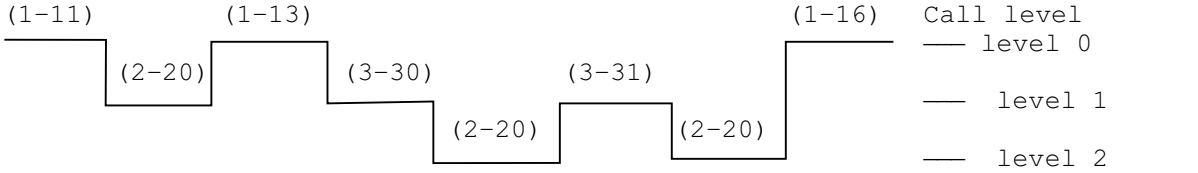
```
O0001
...
N10 G66 P2
```

```
N11 G1 G91 Z10      (1-11)
N12 G66 P3
N13 Z20             (1-13)
N14 G67             (G66 P3 call deletion)
N15 G67             (G66 P2 call deletion)
N16 Z-5             (1-16)
...

O0002
N20 X4              (2-20)
N21 M99

O0003
N30 Z2              (3-30)
N31 Z3              (3-31)
N32 M99
```

Taking only the blocks containing motion into account, the execution order will be as follows:



The first of the numbers in parentheses means the number of the program being under execution, and the second one means the number of the block being under execution.

Each call G66 must be deleted by separate instruction G67. The first instruction G67 deletes the call G66 that was called last, and then, the other calls will be deleted by a succeeding instruction G67, in reverse order. The instruction G67 given in the block N14 deletes the macro called in the block N12 (O0003), and the instruction G67 given in the block N15 deletes the macro called in the block N10 (O0002).

23.3.3 Macro Modal Call from Each Block (G66.1)

Due to the instruction

G66.1 P(program number) L(repetition number) <argument assignment>

all the blocks following it will be interpreted as an argument assignment, the ***macro program*** with the number defined at the address P (program number) ***will be called*** sequentially and repeatedly ***for each block***; the repetition number is given at the address L.

The effect produced by the instruction is the same as if each block was a macro call G65:

```
G66.1 P L
X Y Z = G65 P L X Y Z
M S = G65 P L M S
X = G65 P L X
G67
```

It is a modal call.

The assigned macro will be called until the instruction

G67

is issued to cancel the modal call.

Rules of argument assignment

1. ***In the block executing start*** (where G66.1 P L is programmed):
   In the call G66.1, the addresses G, P and L cannot be used for argument transfer, but the values specified will be written in the appropriate local variable.
2. ***In the blocks following the instruction G66.1***:
   The addresses G, P and L can also be used.
   The address G (G: #10) ***can be used with the restriction*** that reference to ***only one address G in a block*** is accepted by the control; if several addresses G are programmed, the error message '2017 Illegal address G' will be sent by the control.

Rules of argument assignment in the case of G66.1

The assigned macro will already be called from that block where the code G66.1 was given, taking the rules of argument assignment in the item 1 into account.
From the block following the code G66.1 to the block containing the code G67, each NC block will resulted in macro call in accordance with the rules of argument assignment in the item 2. Macro will not be called in the case of vacant block, e.g. N1240 where there is only a reference to an address N, or it will not be called from the block containing macro instruction.

Multiple call G66.1

In the case of multiple call of G66.1-type macros, ***at first the macro called last will be called on read-in of each block***, handling the addresses of this block as argument; and then, reading in and handling the blocks of this macro, the macro specified one-ahead will be called.
Each call G66.1 must be deleted by separate instruction G67. The first instruction G67 deletes the call G66.1 that was called last, and then, the other calls will be deleted by a succeeding instruction G67, in reverse order.

The following program part is written for the plane XY, at vertical position of the machine head.

```
...
G1 X40 Y30 F1000
G3 X0 Y50 I-40 J-30
G1 Y0
...
```

In the meantime, the machine head had been mounted into horizontal position. In order that the program written for the plane XY has not to be written again, the axes Y and Z have to be inverted, and direction of the axis X has to be reversed so that the coordinate system remains right-handed. By calling the macro O0400, the direction of motion along X will be reversed and the axes Y and Z will be inverted:

Main program:

```
...
G66.1 P400                (Call for inverting macro)
G18 X0 Y0 Z-5             (Plane designation, positioning)
G1 X40 Y30 F1000          (O0400 call)
G3 X0 Y50 I-40 J-30       (O0400 call)
G1 Y0                     (O0400 call)
G67                       (Deletion of modal call)
...
```

Macro:

```
IF[#10EQ66.1] GOTO10 (avoiding recursive call)
G#10 X-#24 Y#26 Z#25 I-#4 K#5 R#18 F#9  (changes)
N10 M99
```

The macro O0400 will already be called by the block G66.1.P400. In this case, the code 66.1 will be received at the variable #10. By the first call, return will immediately be executed. The second block of the macro will receive and process the arguments: It will invert the axes Y and Z, and reverse the direction X.

23.3.4 System Macro Call by Codes G Given in Parameter

For macro call, single codes G or arrays of codes G given in parameter can be assigned for each channel.

Assigning 10 single codes G for system macro call

In the parameter field, ***for each channel, maximum 10 different codes G can be assigned,*** for which macro call can be initiated. In this case, instead of the instruction line

Nn G65 Pp <argument assignment>

the instruction line

Nn **Gg** <argument assignment>

has to be written. In the parameter field it is necessary to set which calling G code has to call which program number. The codes G65, G66, G66.1 and G67cannot be assigned for this purpose.

These parameters are the following:

N1704 G(9010): the code G calling the program with the name O9010.nct
N1705 G(9011): the code G calling the program with the name O9011.nct
:
N1713 G(9019): the code G calling the program with the name O9019.nct

If 0 is written in the parameter, the macro with the given program number will not be called.
If ***macro call*** is to be initiated ***by G0***, ***1000*** has to be written in the parameter.

Assigning an array of codes G for system macro call

By using the parameters below, ***an array of codes G can be assigned*** for macro call, for each channel.

In the parameter N1714 Start G Macro, the initial number of the array of codes G is specified by decimal integer number.
In the parameter N1715 Start Prg No, the program number of the macro belonging to the code G given in the parameter Start G Macro is specified.
In the parameter N1716 No. of G Codes, the number of elements of the array of codes G is specified.

If No. of G Codes=0, for these codes G macro will not be called.

An example:
Let, for example, the initial code of the array be G2000. Therefore, Start G Macro=2000 has to be set.
If the code G2000 calls the program O3400, then Start Prg No=3400.
In the parameter No. of G Codes, the number of codes G belonging to the array can be specified. If there is 50 codes in the array, then No. of G Codes=50.

So, relation between the codes G and the program numbers is as follows:

Code G 1	G2000 ➜	Program number 1	O3400
Code G 2	G2001 ➜	Program number 2	O3401
Code G 3	G2002 ➜	Program number 3	O3402
.....			
Code G 50	G2049 ➜	Program number 50	O3449

Assigning an array of codes G with decimal point for system macro call

By using the parameters below, such an array of codes G can be assigned for macro call, for each channel, the code of which contains decimal point and one decimal place.

In the parameter N1717 Start Dec G Macro, the initial number of the array of codes G is specified with decimal point and one decimal place.

In the parameter N1718 Start Prg No. Dec G, the program number of the macro belonging to the code G given in the parameter Start Dec G Macro is specified.

In the parameter N1719 No. of Dec G Codes, the number of elements of the array of codes G containing decimal point is specified.

If No. of Dec G Codes=0, for these codes G macro will not be called.

An example:

If the starting code of the group is, for example, G310.5, the parameter has to be filled as follows: Start Dec G Macro=310.5.

If the code G310.5 calls the program O4000, then Start Prg No. Dec G=4000.

In the parameter No. of Dec G Codes, the number of G codes belonging to the group can be given. If there are 10 codes, then No. of Dec G Codes=10.

So, relation between the codes G and the program numbers is as follows:

Code G 1	G310.5	➜	Program number 1	O4000
Code G 2	G310.6	➜	Program number 2	O4001
Code G 3	G310.7	➜	Program number 3	O4002
.....				
Code G 10	G311.4	➜	Program number 10	O4009

Making calls for code G modal

If ***negative value is assigned*** to single codes G in the case of single calls, and to the ***parameters*** N1714 Start G Macro or N1717 Start Dec G Macro in the case of array calls, ***modal call will be generated by the assigned code G or group of codes G.*** For example, if G(9011)=–120, the instruction G120 will result modal call in the program. The type of the call will be determined by the following parameter state :

N1755 Macro Contr #0 MEQ=0: the type of the call G is G66;

N1755 Macro Contr #0 MEQ=1: the type of the call G is G66.1.

If the value of the is 0, the macro will be called at the end of each motion block. If the value of the parameter 1, the macro will be called by each block.

Referring to the same code G in the body of the macro G

If a ***standard code G*** (for example G01) ***is designated as user call*** and it is referred again ***in the body of the macro***, this reference will not result in an additional macro call, but ***it will be interpreted*** and executed ***as a usual code G*** by the control.

If a ***non-standard code G*** (for example G101) ***is designated as user call*** and ***the calling code G*** (in this case G101) is referred again ***in the body of the macro***, ***the error message*** ‘2123 Not

implemented function: G<number>' ***will be sent by the control***.

Referring to M, S ... in the body of a macro G, referring to G in the body of an M, S ... call

– Calling a system subprogram/ macro M, S, T, A, B, C, ASCII from calling a system macro G, and

– calling a system macro G from calling a system subprogram/ macro M, S, T, A, B, C, ASCII is enabled depending on the following parameter state:

N1755 Macro Contr #1 ENC=0: call will not be initiated (they will be executed as usual code M, S, ...G);

N1755 Macro Contr #1 ENC=1: call will be initiated.

The argument set of the system macros G

The argument set can be defined as follows:

– if the type of the code is G65 or G66, it will be the argument set assigned to the G65, as well as P and L;

– if the type of the code is G66.1, the subject-matter mentioned at the code G66.1will apply to its argument set.

The modal call can be deleted by instruction G67.

23.3.5 System Macro Call by Codes M Given in Parameter

For macro call, single codes M or arrays of codes M given in parameter can be assigned for each channel.

Assigning 10 single codes M for system macro call

In the parameter field, for each channel, maximum 10 different codes M can be assigned, for which macro call can be initiated. In this case, instead of the instruction line

Nn G65 Pp <argument assignment>

the instruction line

Nn **Mm** <argument assignment>

has to written. In this case, ***the code Mm will not be transferred to the PLC***, but the macro with appropriate program number will be called.

In the parameter field it is necessary to set which calling code M has to call which program number. These parameters are the following:

N1733 M(9020): the code M calling the program with the name O9020.nct

N1734 M(9021): the code M calling the program with the name O9021.nct

:

N1742 M(9029): the code M calling the program with the name O9029.nct

If 0 is written in the parameter, the macro with the given program number will not be called.

Assigning an array of codes M for system macro call

By using the parameters below, ***an array of codes M can be assigned*** for macro call, for each channel.

In the parameter N1743 Start M Macro, the initial number of the array of codes M is specified by decimal integer number.

In the parameter N1744 Start Prg No. M Macro, the program number of the macro

belonging to the code M given in the parameter Start M Macro is specified.
In the parameter N1745 No. of M Macro Codes , the number of elements of the array of codes G is specified.
If No. of M Macro Codes=0, for these codes M macro will not be called.

An example:
Let, for example, the initial code of the array be M500. Therefore, Start M Macro=500 has to be set.
If the code M500 calls the program O3500, then No. M Macro=3500.
In the parameter No. of M Macro Codes , the number of codes M belonging to the array can be specified. If there is 20 codes in the array, then No. of M Macro Codes=20.
So, relation between the codes M and the program numbers is as follows:

Code M 1	M500	➜	Program number 1	O3500
Code M 2	M501	➜	Program number 2	O3501
Code M3	M502	➜	Program number 3	O3502
.....				
Code M 20	M520	➜	Program number 20	O3519

Position of the code M starting a macro call in the block
In the block, the code M assigned in the parameter field and initiating a macro call can be preceded only by the symbol / and the address N (block number, the input argument can only follow then.

The macro called by code M is not modal.
By using code M, always G65-type, so ***non-modal*** call can be specified.

Referring to the same code M in the body of the macro M
If, ***in the body of the macro***, ***the same code M is referred*** again, the macro will not be called again, but ***the code M will be transferred to the PLC***.

Referring to G, S ... in the body of a macro M, referring to M in the body of an G, S ... call
– Calling a system subprogram/ macro G, S, T, A, B, C, M, ASCII from macro call initiated by a code M, and
– macro call initiated by a code M from calling a system subprogram/ macro G, S, T, A, B, C, ASCII
is enabled depending on the following parameter state:
N1755 Macro Contr #1 ENC=0: call will not be initiated (they will be executed as usual code M, S, ...G);
N1755 Macro Contr #1 ENC=1: call will be initiated.

The argument set of the codes M starting a macro call
In the block containing ***macro call started by code M***, ***both argument sets***, i.e. either the set 1 or the set 2, can be used.
Following the specification of the calling code M, the second code M will be interpreted as argument and transferred to the variable #13 by the control.

23.3.6 System Subprogram Call by Codes M Given in Parameter

For system subprogram call, single codes M or an array of codes M given in parameter can be assigned for each channel.
The difference between the code M calling a subprogram and the code M calling a macro is that ***the code M calling a subprogram does not transfer any argument to the subprogram***, but the code calling a macro does it.

Assigning 10 single codes M for system subprogram call

In the parameter field, for each channel, maximum 10 different codes M can be assigned, for which subprogram call can be initiated. In this case, ***the code Mm will not be transferred to the PLC***, but the macro with appropriate program number will be called, i.e. instead of the instruction

Nn Gg Xx Yy M98 Pp

the instruction

Nn Gg Xx Yy **Mm**

has to written..
In the parameter field it is necessary to set which calling code M has to call which program number. These parameters are the following:

N1720 M(9000): the code M calling the program with the number O9000
N1721 M(9001): the code M calling the program with the number O9001
:
N1729 M(9009): the code M calling the the program with the number O9009

If 0 is written in the parameter, the subprogram with the given program number will not be called.

Assigning an array of codes M for system subprogram call

By using the parameters below, an array of codes M can be assigned for subprogram call, for each channel.

In the parameter N1730 Start M SubP, the initial number of the array of codes M is specified by decimal integer number.
In the parameter N1731 Start Prg No. M SubP , the program number of the subprogram belonging to the code M given in the parameter Start M SubP is specified.
In the parameter N1732 No. of M Codes, the number of elements of the array of codes M is specified.

If No. of M Codes=0, for these codes M subprogram will not be called.

An example:
Let, for example, the initial code of the array be M600. Therefore, Start M SubP=600 has to be set.
If the code M600 calls the subprogram O3600, then Start Prg No. M SubP=3600.
In the parameter No. of M Codes , the number of codes M belonging to the array can be specified. If there is 30 codes in the array, then No. of M Codes=30.
So, relation between the codes M and the program numbers is as follows:

Code M 1	M600	➜	Program number 1	O3600
Code M 2	M601	➜	Program number 2	O3601
Code M3	M602	➜	Program number 3	O3602

.....
Code M 20 M630 ➜ Program number 20 O3629

Position of the code M starting a subprogram call in the block
In the block, the code M can be written in arbitrary position.

Referring to the same code M in the body of the macro M
If, ***in the subprogram***, ***the same code M is referred*** again, the subprogram will not be called again, but ***the code M will be transferred to the PLC***.

Referring to G, S ... in the body of a subprogram M, referring to M in the body of an G, S ... call
– Calling a system subprogram/ macro G, S, T, A, B, C, M, ASCII from subprogram call initiated by a code M, and
– subprogram call initiated by a code M from calling a system subprogram/ macro G, S, T, A, B, C, ASCII
is enabled depending on the following parameter state:
N1755 Macro Contr #1 ENC=0: call will not be initiated (they will be executed as usual code M, S, ...G);
N1755 Macro Contr #1 ENC=1: call will be initiated.

23.3.7 System Subprogram Call by Codes A, B, C, S, T Enabled in Parameter

Having been enabled, a subprogram can be called by codes A, B, C, S and T, for each channel.

Assigning codes A, B, C, S, T for subprogram call

At several bit positions of the parameter N1746 ABCST, the subprogram calls are as follows:

when #0 AM=1, the subprogram O9030.nct will be called by the code A;
when #1 BM=1, the subprogram O9031.nct will be called by the code B;
when #2 CM=1, the subprogram O9032.nct will be called by the code C;
when #3 SM=1, the subprogram O9033.nct will be called by the code S;
when #4 TM=1, the subprogram O9034.nct will be called by the code T.

In the case of addresses designated for calling subprograms, the values A, B, C, S and T will not be transferred either to the interpolator (if the address A, B or C is assigned to axis), or to the PLC, but the subprograms above will be initiated by the codes A, B, C, S and T.
If, for example, the code T is assigned to call a subprogram, the block

Gg Xx Yy **Tt**

will be equivalent to the following two blocks:

#199=t
Gg Xx Yy M98 P9034

Transferring the argument of the codes A, B, C, S and T to the subprogram

The values assigned to the addresses A, B, C, S and T, as arguments, will be written in the following ***global variables***:

Code A ➜ #195
Code B ➜ #196
Code C ➜ #197
Code S ➜ #198
Code T ➜ #199

Then, these variables can be used by the subprogram.

Position of the codes A, B, C, S and T starting a subprogram call in the block

In the block, the codes can be written in arbitrary position.

Referring to the calling code in the body of the subprogram A, B, C, S and T

If, in the subprogram being called to ***the address A, B, C, S and T***, the calling address is referred again, the subprogram will not be called again, but ***the code A, B, C, S and T will be transferred either to the interpolator or to the PLC.***

Referring to G, S ... in the body of a subprogram A, B, C, S and T, referring to A, B, C, S, T in the body of an G, S ... call

– Calling system subprogram/macro G, M, S, T, A, B, C, ASCII from subprogram call initiated by the code A, B, C, S, T (if the call initiated not from the calling address), and
– calling a subprogram initiated by the code A, B, C, S, T from system subprogram/macro G, M, S, T, A, B, C, ASCII call (if the call initiated not from the calling address)

is enabled depending on the following parameter state:

N1755 Macro Contr #1 ENC=0: call will not be initiated (they will be executed as usual code M, S, ...G);
N1755 Macro Contr #1 ENC=1: call will be initiated.

23.3.8 System Subprogram Call by Codes ASCII Given in Parameter

Subprogram can be called by four codes ASCII given in parameter, for each channel. ***The letters of the English alphabet*** can be selected from ***the codes ASCII.***

Assigning the code ASCII for subprogram call

In the parameters

N1747 ASCII Code SubP1
N1748 ASCII Code SubP2
N1749 ASCII Code SubP3
N1750 ASCII Code SubP4

4 different codes (letters of the English alphabet) can be set. Then, by these parameters, the control will call subprograms with the number (Onnnn) specified in the following parameters:

N1751 Prg No. ASCII Call1
N1752 Prg No. ASCII Call2
N1753 Prg No. ASCII Call3
N1754 Prg No. ASCII Call4

Transferring the argument of the code ASCII to the subprogram

The values assigned to the address ASCII, as argument, will be written in the following ***global variables***:

Code ASCII 1 ➜ #191
Code ASCII 2 ➜ #192
Code ASCII 3 ➜ #193
Code ASCII 4 ➜ #194

Then, these variables can be used by the subprogram.

Position of the code ASCII starting a subprogram call in the block

In the block, the codes can be written in arbitrary position.

Referring to the calling code in the body of the subprogram being called by the code ASCII

If, ***in the subprogram being called by the code ASCII, the calling address is referred*** again, the subprogram will not be called again, but ***the code ASCII will be transferred either to the interpolator or to the PLC***.

Referring to G, S ... in the body of the subprogram ASCII, referring to ASCII in the body of an G, S ... call

– Calling system subprogram/macro G, M, S, T, A, B, C, ASCII from subprogram call initiated by the code ASCII (if the call initiated not from the calling address), and
– calling the subprogram initiated by the code ASCII from system subprogram/macro G, M, S, T, A, B, C, ASCII call (if the call initiated not from the calling address)

is enabled depending on the following parameter state:

N1755 Macro Contr #1 ENC=0: call will not be initiated (they will be executed as usual code M, S, ...G);
N1755 Macro Contr #1 ENC=1: call will be initiated.

23.3.9 Displaying the Blocks of Macros and Subprograms in Automatic Mode

Basically, the blocks of the macros and subprograms being under execution are listed by the control. However, it is possible to disable listing the blocks. In this case, the control will consider the whole macro or subprogram a block, and, in the block-by-block mode, it will not stop at the inner blocks.

Listing can be enabled or disabled by the bits #0 MD8 and #1 MD9 of the parameter N1756 List Contr. Listing the macros and subprograms numbered from 8000 to 8999 is controlled by the bit MD8, but listing those numbered from 9000 to 9999 is controlled by the bit MD9.
If the value of the bit MD8 or MD9 is 0, the blocks of the macro and the subprogram will not be listed during execution of the macros and the subprograms from 8000 to 8999 and from 9000 to 9999, respectively; execution will not stop at the inner blocks in the block-by-block mode.
If the value of the bit MD8 or MD9 is 1, the blocks of those macros and subprograms will also be listed, and the control will also stop at the inner blocks in the block-by-block mode.

23.4 Interruption-type Macro

In the course of running the part program, it is possible to suspend the actual program, to call another program, and then, after execution of this program, to continue the suspended one. The events causing interruption can be, for example, power failure, tool break or data gathering done between times.
The program being called during suspension is named ***interruption-type macro***.
Calling the interruption-type macro is started by the signal CP_MINT ***from the PLC program***.
In the part program, ***calling the interruption-type macro has to be enabled***, otherwise the PLC signal will be ineffective.
In addition, the program to be run due to the signal has to be designated.
The instruction

M96 P(program number)

or

M96 <filename>

enables the interruption signal CP_MINT from the PLC program ***to influence***. Due to the interruption signal, ***the program with the number given*** at the address P (program number) will be called.

The rules of programming the program number given at the address P are dealt with in the subsection **14.4.1 Identification of Programs in Memory. The Program Number (O)** on page 142. ***The rules of extension and in-memory location of the interruption-type macros given by the program number*** are the same as those of subprograms. See the subsection **14.4.2 Calling a Subprogram (M98)** on page 142.

Location of the interruption-type macro in the memory

The bit state #5 SYM of the parameter N1758 Intrrt Contr determines where in the memory the interruption-type macro has to be searched by the control:

– in the case of #5 SYM=0: always in the folder, in which the main program is there, even if the interruption-type macro was enabled not in the main program by the instruction M96P;
– in the case of #5 SYM=1: always in the directory of the folder SystemMacros, which directory belongs to the appropriate channel.

Return from the interruption-type macro occurs ***by the code M99***.

The instruction

M97

disables calling the interruption-type macro, i.e. the interruption-type macro will not be called any more even if the signal is received. Interruption will also be disabled by reset and program end.

The interruption-type macro can exclusively be used
in ***automatic*** or ***manual data input*** mode,
when ***start*** state is effective.

Enabling the function to work

In the control, the bit state #0 USD=1 of the ***parameter*** N1758 Intrrt Contr ***enables the interruption-type macro to work***.
If the value of the parameter is #0 USD=0, the interruption-type macro will not work, and the PLC can use the codes M96 and M97 as function.

Enabling interruption by other codes

In the bit state #4 MCD=0 of the parameter N1758 Intrrt Contr, ***enabling/disabling*** the interruption-type macro can be done by using the code pair ***M96/M97***.
In the bit state #4 MCD=1 of the parameter N1758 Intrrt Contr, enabling/disabling the interruption-type macro can be done by using a code pair different from the M96/M97.
In this case, ***the code M enabling the interruption*** can be given in the ***parameter*** N1759 M Code MI On, but ***the code M disabling the interruption*** can be given in the parameter N1760 M Code MI Off.
It can be useful in the case, when the code M96 or M97 is already used by the PLC for other function.

Subprogram-type or macro-type interruption

In the bit state #1 STP=1 of the parameter N1758 Intrrt Contr, interruption will be ***subprogram-type***, i.e. the level of local variables will not be higher by the call, the values of the local variables in the called program and in the interrupted program will be the same.
On the other hand, in the bit state #1 STP=0, interruption will be ***macro-type***, the level of local variables will be higher in the called program, i.e. the values of the local variables in the calling program and in the interrupted program will be different.

Edge-controlled and level-controlled interruption

Interruption is ***edge-controlled*** in the case, when the interruption-type macro is called by the ***rising edge*** of the PLC signal CP_MINT. If the signal remains switched on after returning from the interruption-type macro, the macro will be called again only after switching the signal off and then switching it on again.
On the contrary, in the case of ***level-controlled*** interruption, if the signal CP_MINT remains switched on after returning from the interruption-type macro, the interruption-type macro ***will be called again*** until the signal CP_MINT will be switched off.

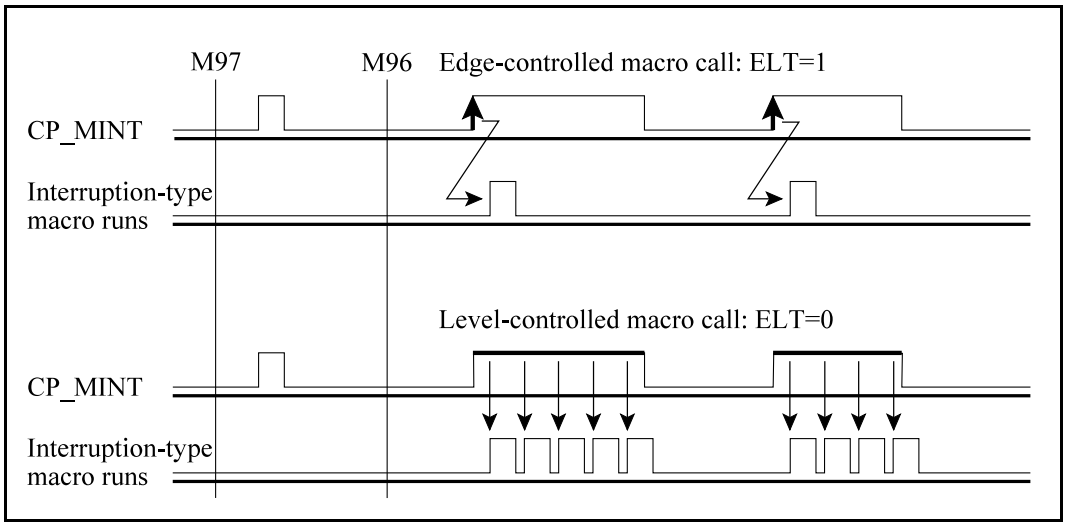


Fig. 23.4-1

The bit #3 ELT of the parameter N1758 Intrrt Contr determines the type of the macro call. If #3 ELT=1, the macro call will be edge-controlled, if #3 ELT=0, the macro call will be level-controlled.

Interruption during block execution or at the end of the block

The user can decide whether the interruption-type macro begins to run ***at the moment of receiving the signal***, or it waits until motion will be completed in the actual block and begins to run ***only at the end of the block***.
In the bit state #2 TPI=0 of the parameter N1758 Intrrt Contr , the interruption-type macro will be called immediatel, but in the bit state #2 TPI=1 it will be called only at the end of the block.

Modal information in the interruption-type macro

Calling the interruption-type macro differs from a normal subprogram call.
In the case of a subprogram call (M98Pp), the subprogram inherits from the calling program
the modal information#4001...#4130 valid in the previous block, and
the modal information#4201...#4330 valid in the block being executed.
Certainly, these two kinds of information are different, because, due to buffering, execution lags behind preprocessing the blocks.
In the case of the interruption-type macro, the situation is different.
Interruption occurs during execution of the program, therefore ***the interruption-type macro (M96Pp) can read out the modal information valid in the block being executed, i.e. in the interrupted block, in the variables #4401...#4530***.
The modal information #4001...#4130 and #4201...#4330 will be initialized after entering the interruption-type macro.
During execution of the interruption-type macro, the variables #4001...#4130 and #4201...#4330 work in the usual way, but the variables #4401...#4530 retain the modal information valid at the moment of interruption.

Return from the interruption-type macro by M99

The control ***stores the modal information valid at the moment of interruption***, and then, ***after return by M99, it restores the saved modal information***.
If the program is interrupted during interpolation, the program will finish interpolation after return; if the program is interrupted at the end of a block, the succeeding block will be taken and executed.
Therefore, if motion has to also be programmed in the interruption-type macro, it will be advisable to store the block-end position (#5001 ...) after entering the interruption-type macro, and then, prior to return, to move back to this position.

Return from the interruption-type macro by M99Pp

If return from the interruption-type macro is executed by the instruction M99Pp, the machining will be continued from that block of the interrupted program the number of which is given at the address P.
After return by the M99Pp, the saved modal information will not be restored by the control, but the program will be continued using the modal information generated in the interruption-type macro.

23.5 NC and Macro Instructions. Execution of Macro Blocks

NC and macro blocks

In the program language, ***NC and macro blocks*** can be distinguished.
The blocks written with traditional codes G, M etc. are considered to be ***NC blocks***, even if the values of the addresses are not only numbers but variables or formula as well.
The following blocks are considered to be ***macro instructions***:
– the blocks containing assignment statement: #i=#j;
– the blocks containing conditional or cycle organizing instruction: IF, WHILE;
– the blocks containing control instructions: GOTO, DO, END;
– the blocks containing macro call: G65, G66, G66.1, G67 or those codes G or M initiating macro call;
– the subprogram call (M98P or the subprogram initiated by the A, B, C, S, T and M);
– the code of return from a subprogram or a macro (M99).

Synchronization of instructions (G53)

In the bit state #6 MBM=1 of the parameter N1301 DefaultG2 (multibuffer mode), the block preparator reads both the NC and macro blocks ahead, processes them, and then, places them in the buffer memory. The executors, the interpolator and the PLC, take the blocks from the buffer memory during execution of the program. It is necessary in order that the interpolator will be able to move the axes continuously, and will not have to wait for processing the succeeding block.
As a result of block processing in advance and buffering, the executor lags behind the block preparator by hundreds of blocks. For example, the block preparator is already processing the block 1500 while the executor is executing the block 1000 yet.
This is the reason of distinguishing the variables #4001...#4130 (#_BUFx) and the variables #4201...#4330 (#_ACTx). While the former provides information about the blocks placed in the buffer memory last, the latter gives information about the blocks being executed.
In some cases, preprocessing and buffering the blocks has to be suspended; for example, when it is necessary to wait for a given position of an axis during program execution.
The instruction

G53

written in a separate block suspends reading the blocks ahead. The control waits until the block buffer will be empty and only after that begins to read in and process the succeeding block.

Execution of macro blocks

The control can execute the macro blocks in parallel with execution of NC blocks or following the execution of them. It is the parameter N1755 Macro Contr #2 SBM that controls execution of NC and macro blocks. If the value of the parameter is:

=0: the NC and macro blocks will be executed in the sequence written in the program;
=1: macro instructions will be executed during execution of the NC blocks.

An example:

SBM=0

```
%O1000
...
N10 #100=50
N20 #101=100
N30 G1 X#100 Y#101
N40 #100=60 (assignment statement
after N30)
N50 #101=120 (assignment statement
after N30)
N60 G1 X#100 Y#101
```

Assignment statement written in the blocks N40 and N50 will be carried out after execution of the block N30.

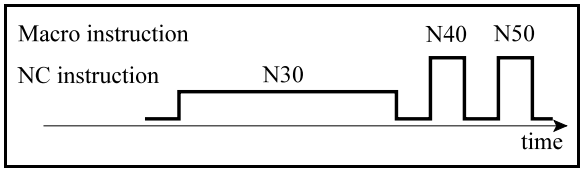


Fig. 23.5-1

☞ *Consequences:*
– *slower program execution,*
– *if execution of the block N30 is interrupted and then machining is started again, and since the variables of the block N30 are not rewritten by the blocks N40 and N50 yet, machining can be continued in a usual way.*

SBM=1

```
%O1000
...
N10 #100=50
N20 #101=100
N30 G1 X#100 Y#101
N40 #100=60 (assignment statement
during N30)
N50 #101=120 (assignment statement
during N30)
N60 G1 X#100 Y#101
```

Assignment statement written in the blocks N40 and N50 will be carried out during execution of the block N30.

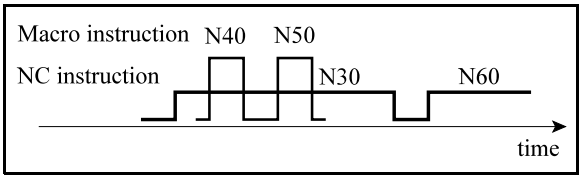


Fig. 23.5-2

☞ *Consequences:*
– *faster program execution,*
– *if execution of the block N30 is interrupted and then machining is started again, and since the variables of the block N30 are rewritten by the blocks N40 and N50 already, machining can be continued only in the case, when search is started for the block N30.*

23.6 Pocket Milling Macro Cycle

The instruction

G65 P9999 X Y Z I J K R F D E Q M S T

starts a pocket milling cycle.

Prior to calling the cycle, the tool must be positioned over the geometric center of the pocket in the selected plane, at a safety distance from the workpiece surface. At the end of the cycle the tool will be positioned back to the same point.

Interpretation of the addresses of the block is as follows:

X: the size of the pocket in the direction X;

Y: the size of the pocket in the direction Y;

Z: the size of the pocket in the direction Z.

It is determined by the instructions G17, G18 and G19, which of the three coordinates will be the length, width and depth of the pocket. For example, in the case of G17, Z will be the depth of the pocket, the longer one of X and Y will be the length of the pocket and the shorter one will be the width thereof.

These values have to be entered in absolute values as positive numbers.

R: the radius of the corners of the pocket.

Rounding (if any) of the pocket corners has to be specified at the address R. If the address R is not filled, the rounding of the pocket corners will be equal to the tool radius.

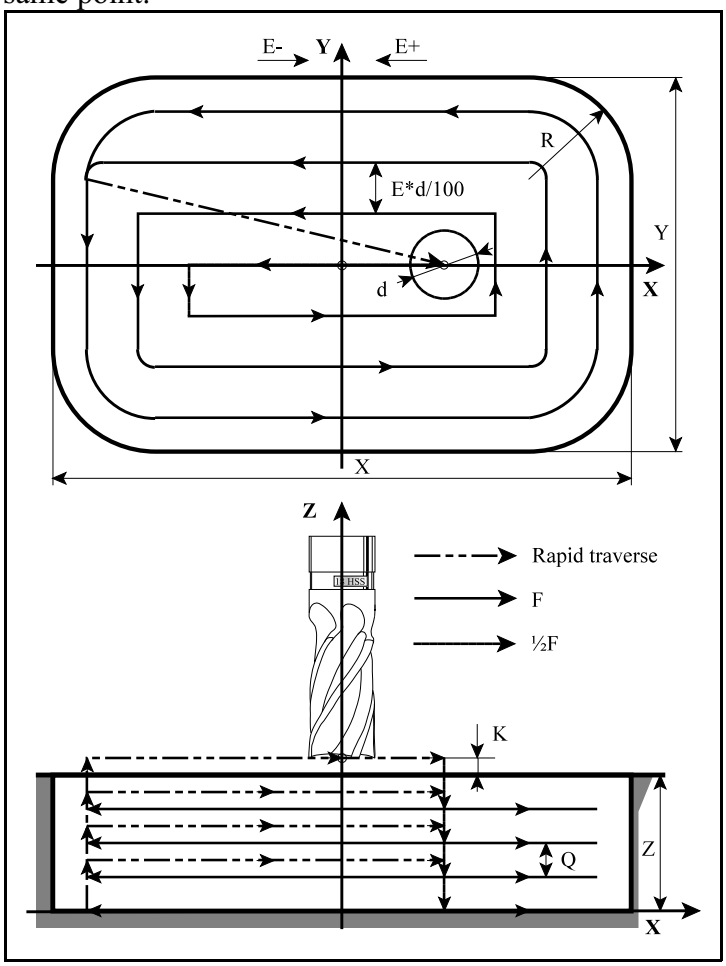


Fig. 23.6-1

I: the safety distance in the direction of depth in the case of G19.

J: the safety distance in the direction of depth in the case of G18.

K: the safety distance in the direction of depth in the case of G17.

Depending on the plane selected, the safety allowance in the direction of the tool has to be specified at the addresses I (G19), J (G18) or K (G17) in the block. When the cycle is started, it is assumed by the control, that the tip of the tool is located at such a distance from the workpiece surface. In the course of milling the pocket, when removing a level of material is completed, the tool will be retracted by such a distance so that it can be positioned to the start point for milling the next level.

D: the address of the cell containing the tool radius compensation.

The radius compensation cell number of the tool used in the program has to be specified mandatorily at the address D. Otherwise, milling a pocket has to be carried out in the state

G40.
E: the width of cutting in percent of the milling tool diameter:
with the sign +: counter clockwise machining;
with the sign –: clockwise machining.
At the address E, the types of information can be given to the control. The value of E determines how great the width of cutting has to be in percent of the milling tool diameter. If it is not given, its value is automatically assumed by the control to be +83%. Depending on the width of the pocket, the data given at the address E can be specified by the control in such a way so that the value of infeed will be constant in the course of milling a level. However, modification has to be decrease only. The sign of the address E shows the direction of milling. In the case of E+, i.e. when it is positive, machining is being performed in the counter clockwise direction; in the case of E–, i.e. when it is negative, machining is being performed in the clockwise direction.
Q: depth of cut
At the address Q, the depth of cut can be given in the unit system used, i.e. in mm or in inch. Depending on the depth of the pocket, the programmed value can be revoked by the control in order to obtain constant division of cut. However, modification has to be decrease only.
F: feed
At the address F, the value of the feed used in the course of cycle can be given. If no value is given at the address F, the modal value of F will be taken into account by the control. 50% of the value of F will be applied in the following cases:
– When machining a level is started and drilling to the depth Q is executed in the direction of the tool.
– During longitudinal milling, as long as the tool is loaded on both sides.
M S T: function
In the block calling the pocket milling, one function M and functions S and T can be given, which will be executed prior to starting the milling.

Degenerate cases of pocket milling

If the width of the pocket is not given, the radius of the pocket corners will be taken on twice and it will be the width of the pocket.

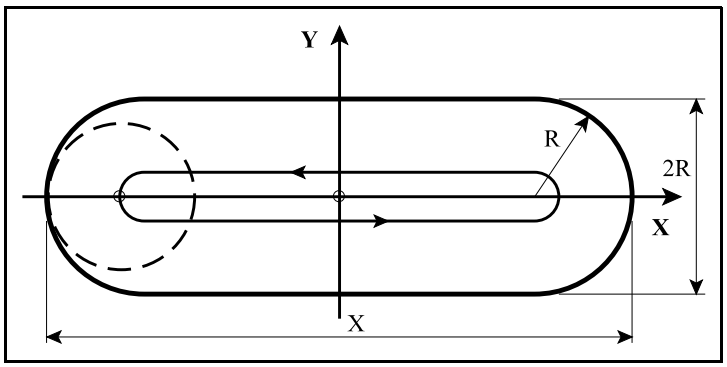


Fig. 23.6-2

If neither the width of the pocket nor the radius of the corner is given, the diameter of the tool applied will be taken as the with of the pocket (groove).

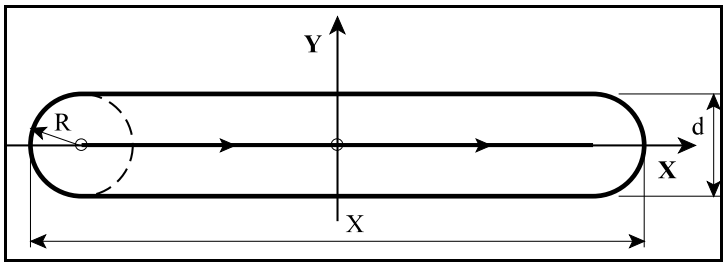


Fig. 23.6-3

If neither the length nor the width of the pocket is given, and only the address R is programmed, a circular pocket will be milled.

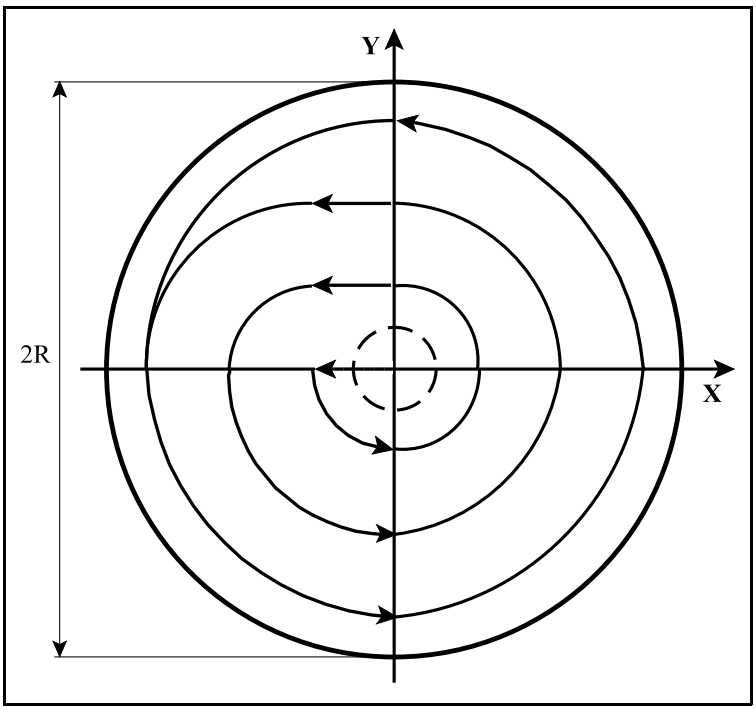


Fig. 23.6-4

If neither the length nor the with and the radius is given, the cycle will degenerate into drilling.

Error messages which are possible during execution of pocket milling:

MACRO ERROR 1: wrong block specification. Possible causes:
– The depth of the pocket is not specified.
– The tool radius is not specified.
– The depth of cut is not specified.

MACRO ERROR 2: wrong dimensional specification. Possible causes:
– The dimension given as the length or the width of the pocket smaller than the double of the pocket radius.

– The value of the length and the width of the pocket smaller than tool diameter called at the address D.
– The value of the width of cutting is 0, or the tool diameter called is 0.
– The value of the depth of cut is 0, i.e. 0 is programmed at the address Q.

24 Writing and Reading the Parameters

The parameters are used to set up operation of the control according to the demands. The parameters are stored in the non-volatile memory of the control. The parameters of the control can be rewritten and read in from the part program.
Each of the parameters has an ***identifier number*** of maximum ***four digits***, which identifies the parameter in the memory. ***One value*** (in the case of global parameters) or ***several different values*** (indexed per channel, per axis, per spindle) ***can belong to these identifier numbers***.
The parameters can be classified in accordance with ***number representation***, ***effect range*** and ***case of taking into account***.

Classification of parameters in accordance with number representation

Bit-type parameters: their value range can be
0 or 1.

Integer-type parameters (DWORD): their value range can be
without sign: 0,...4294967295 or
with sign: -2147483648... +2147483647.

Floating point-type parameters (double): their value range can be
in the case of negative numbers: -1.7×10^{308} ... -5.0×10^{-324}
in the case of positive: 5.0×10^{-324} ... 1.7×10^{308}.

Classification of parameters in accordance with effect range

Global parameters: Those parameters are named global which are valid in every channel of the control. For example, the parameters N2201 Waiting M Codes Min and N2202 Waiting Codes Max are such ones, which designate the M codes group controlling synchronization between channels and are valid for each of the channels.
In the case of global parameters, one parameter value belongs to each identifier number.

Parameters that can be specified per channel: These are the parameters, in which values different per channel can be written. For example, the group N1500 Return Val G73 is such a parameter, in which the distance of retraction can be set up in the drilling cycle G73 and values different per cannel can be written.
In the case of parameters that can be specified per channel, ***maximum 8*** parameter values can belong to each identifier number.

Parameters that can be specified per axis: These are the parameters, in which values different per axis can be written. For example, the bit #0 DIA of the group N0106 Axis properties is such a parameter, in which it can be set per axis whether the the given axis shoukd be programmed in radius or in diameter.
In the case of parameters that can be specified per axis, ***maximum 32*** parameter values can belong to each identifier number.

Parameters that can be specified per spindle: These are the parameters, in which values different per spindle can be written. For example, the group N0608 Spindle Encoder Counts is such a parameter, in which the number of pulses of the encoder mounted on the given axis can be specified per spindle.
In the case of parameters that can be specified per spindle, ***maximum 16*** parameter values can belong to each identifier number.

Classification of parameters in accordance with case of taking into account

Parameters taken into account during run-time: These parameters will be taken into account by the control immediately after rewriting them.

Parameters taken into account after restart: Rewriting these parameters will be taken into account by the control only after restart (after turn-off and then turn-on).

Parameters that can be rewritten in the case of emergency stop state: Rewriting these parameters is enabled by the control only in the case of emergency stop state.

Detailed description of the parameters can be found in the manual 'NCT3xx Machine Tool Controls Parameters'.

24.1 Writing the Parameters from Part Program (G10 L52)

The command

G10 L52 (writing the parameter on)

written in separate row turns on the function of writing the parameter. Then, the serial number and the value of each parameter has to be specified in separate row in accordance with the following way:

N_ (Q_) R_ (writing global parameter)
N_ P_ (Q_) R_ (writing channel/axis/spindle parameter)
...
...

It is the command

G11 (end of writing the parameter)

written in separate row that closes the end of writing the parameter.
Interpretation of the addresses N, P, Q and R:

N: the identifier number of the parameter (0-9999). The leading zeros can be neglected.

☞ ***Warning!*** *Only such parameter having a given number can be rewritten by the use of the command G10 L52, which does not require restart for being taken into account or does not require emergency stop state for being rewritten!*

P: the number of the channel (1-8), the axis (1-32) or the spindle (1-16). The value of P that can be assigned in the specific cases are given in brackets.

☞ ***Warning!*** *If a parameter, which can be specified per channel, is being written and the address P is not filled in, the command will rewrite the parameter of the channel, in which the program is running.*

Q: in the case of the bit-type parameter, it is the number of the bit to be written from 0 to 7.

R: the value of the parameter. For the value of R, the incremental operator I is accepted; it increases the actual value by the value specified at the address RI. In the case of floating-point data, decimal point (.) can be used.

Parameters modified by the command G10 L52 will be saved so modification will be valid

during next turn-on too.

If the addresses N, P, Q or R addresses are filled in incorrectly, the control will send the message ‘2002 <N, P, Q or R> data out of range‘

to the address N: if there is no parameter having such identifier number;
to the address P: if there is no channel/axis/spindle having such number;
to the address Q: if the value of Q is less than 0 or greater than 7;
to the address R: if a value being out of the value range permitted for the parameter having identifier N is written on R.

If any of the data N, P, Q and R is missing, the control will send the error message ‘2004 <N, P, Q, R> data missing’.
If such parameter should be rewritten, which requires restart for being taken into account or requires emergency stop state for being rewritten, the control will send the error message ‘2193The parameter Nnnnn cannot be modified by the command G10’.
An example of writing the parameters:

```
...
G10 L52                 (Writing the parameters on)
N107 P4 Q0 R1           (N0107 RollOver Control A4- REN)
N1339 (P1) R0.5         (N1339 Radius Diff)
N1746 (P1) Q1 R1        (N1746 ABCST BM=1)
G11                     (Writing the parameters off)
...
```

24.2 Reading the Parameters from Part Program (PRM)

Using the assignment statements

#a=PRM[#b,#c]

and

#a=PRM[#b,#c] / [#d]

the value of ***any parameter*** of the control ***can be read out without any constraint***. The meaning of the arguments is as follows:

#a: it is a writable macro variable.
#b: it is the identifier number of the parameter. It can be given indirectly on a macro variable or directly using a number value.
#c: it is the bit number of the parameter in the case of reading a bit-type parameter. It can be given indirectly on a macro variable or directly using a number value.
#d: it is the number of the channel (1-8), the axis (1-32) or the spindle (1-16). The value that can be assigned on #d in the specific cases are given in brackets. It can be given indirectly on a macro variable or directly using a number value.

☞ ***Warning!***

*If a parameter which can be specified per channel is being read and the part **/ [#d]** is neglected from the command, the command will read in the parameter of the channel, in which the program is running.*
If the arguments of the command PRM are specified on macro variables and the value of the macro variable is #0 (empty), it would produce such an effect as if that argument would not have been specified.
Arguments could also be results of macro expressions.

If the command PRM stands on the right side of a command, which is not an assignment statement, the control will send the error message ‘2017 Illegal address PRM’.
If the argument #b of the command PRM, i.e. the number of the parameter id missing, the control will send the error message ‘2064 Syntax error’.
If the bit number #c or the argument /[#d] of the command PRM is missing, the control will send the error message ‘2194 PRM function param <2., 3.> missing’.

An example of reading the parameters:

```
...
#101=PRM[107,0]/[4]       (N0107 RollOver Control A4- REN)
#102=PRM[1339](/[1]       (N1339 Radius Diff)
#103=PRM[1746,1](/[1])    (N1746 ABCST BM=1)
#104=PRM[2201]            (N2201 Waiting M Codes Min)
...
```

25 5-axis Machining

Functions of the 5-axis machining valid for the machine tools on which ***the main axes*** X, Y and Z form a ***right-handed orthogonal coordinate system*** and, in addition to the main axes, there are ***2 rotary axes*** as well.
Standard naming of the rotary axes is illustrated in the picture.
5-axis machine tools managed by the control are described in the following sections and subsections of this chapter.

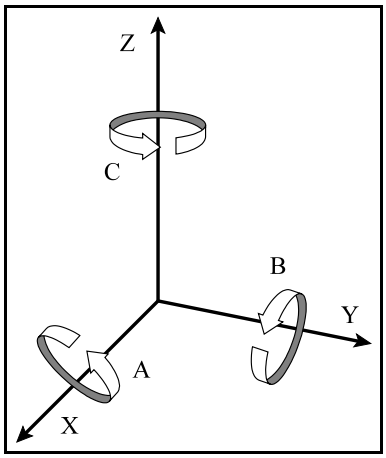


Fig. 24.2-1

25.1 Construction of 5-axis Machine Tools

Types of 5-axis machine tools

The following three main groups of the 5-axis machine tools managed by the control can be distinguished:

machine tools of head-head configuration: the two rotary axes rotate the tool;
machine tools of table-table configuration: the two rotary axes rotate the workpiece;
machine tools of head-table configuration: one of the rotary axes rotates the tool, the other one rotates the workpiece.

The machine type can be set in the parameter N3200 Mechanical Type.

Direction of the tool axis

The next important characteristic of a 5-axis machine tool is the following: what is the main axis (X, Y or Z) with which ***the axis of rotation of the tool*** is parallel in normal situation, i.e. when the machine position of the two rotary axes is 0°. It can be defined in the parameter N3201 Tool Axis Direction.

First and second rotary axis

From the rotary axes available on the machine tool, ***two rotary axes*** that will be used in the course of 5-axis machining have to be selected. Then, it has to be decided which one will be the first rotary axis and which one will be the second rotary axis:

the ***first rotary axis*** bears, mechanically, the second rotary axis;
the ***second rotary axis*** is mounted on the first rotary axis.

☞ ***Note***: *Correct designation of the first rotary axis and the second rotary axis is essential, because this setting will determine the sequence of rotation transformations. Wrong setting will result in faulty operation.*

The first rotary axis and the second rotary axis can be specified in the parameters N3204 No. of the First Rot. Ax. and N3208 No. of the Second Rot. Ax..
In the further parts of this manual, in the case of concrete referring a rotary axis, for example in a sample program, the addresses will be used.
In the further parts of this manual, when the first rotary axis and the second rotary axis is generally referred,

the address Θ, θ (theta) will mean reference to the first rotary axis; and

the address Ψ, ψ (pszi) will mean reference to the second rotary axis.

Direction of the rotary axes

It is necessary to define, which one of the first and the second rotary axes rotates around which axis of the coordinate system X, Y, Z. Direction of the first and the second rotary axes given by the parameters N3205 Direction of the First Rot. Ax. and N3209 Direction of the Second Rot. Ax..

In accordance with standard naming, the axis A rotates around the axis X, the axis B rotates around the axis Y and the axis C rotates around the axis Z; however, different naming can also be used.

Hereafter, the standard axis names are used in this manual.

If the axis of rotation of the rotary axes is parallel with the X, Y and Z axes, the directions of the axes can be set by the use of the following parameter values:

=1: it rotates about the X axis (the basic axis 1);

=2: it rotates about the Y axis (the basic axis 2);

=3: it rotates about the Z axis (the basic axis 3).

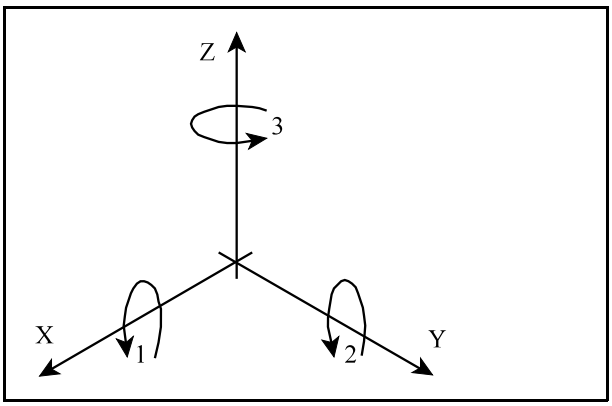


Fig. 25.1-1

In the case of the machines, where the axis of rotation of the rotary axis is not parallel with any one of the linear axes but makes an angle with it, the directions of the axes can be set by the use of the following parameter values:

=4: in the XY plane, inclining with respect to the X axis;

=5: in the YZ plane, inclining with respect to the Y axis;

=6: in the ZX plane, inclining with respect to the Z axis.

In this case, the inclination angle of the given axis can be specified by the parameters N3206 Inclination Angle of the First Rot. Ax. and N3210 Inclination Angle of the Second Rot. Ax., az it is illusztrated in the figure.

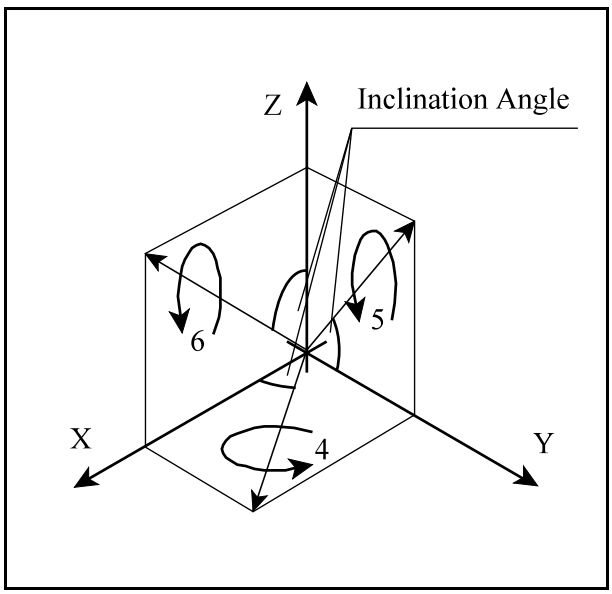


Fig. 25.1-2

Direction of rotation transformation and direction of rotation of the rotary axes

It is necessary to distinguished direction of rotation transformations made around the main axes from direction of rotation of the rotary axes.

The positive direction of the rotation transformations made around the main axes and used in the control ***always follows the right-hand rule***, as it is illustrated in the figure.

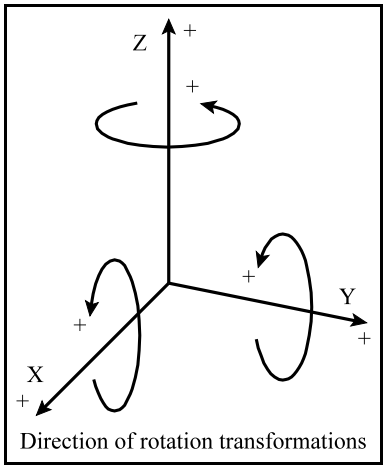


Fig. 25.1-3

Direction of rotation of the rotary axes can differ from this.

Let the axis C rotate around the axis Z.

On a ***machine tool of head-head configuration***, the positive direction of rotation of the axis C will be ***according to the right-hand rule***, because the tool tip will rotate in the positive direction relative to the workpiece only in this case.

On a ***machine tool of table-table configuration***, the situation is often reversed. The positive direction of rotation of the axis C is ***opposite to the right-hand rule***, because the tool tip will rotate in the positive direction relative to the workpiece only in this case, since the workpiece will rotate under the tool.

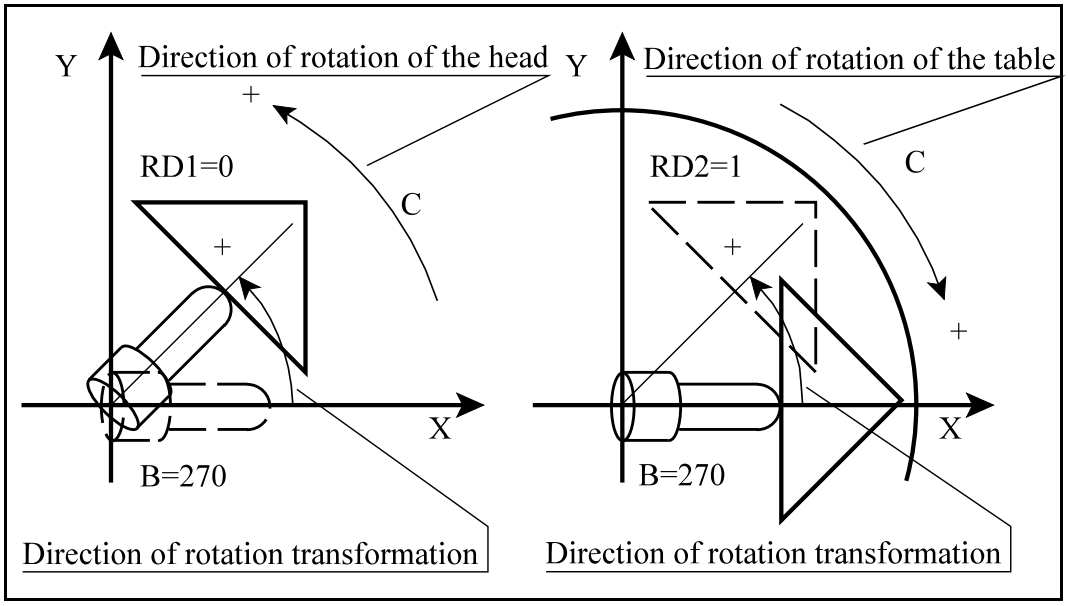


Fig. 25.1-4

It can be set in the parameter N3212 5D Control whether direction of rotation of a rotary axis participating in a 5-axis machining corresponds with direction of the rotation transformation or is opposite to it. If direction of rotation of the first or the second rotary axes corresponds with direction of rotation transformation, the value of the parameter bit RD1 or RD2 will be 0; in the case of non-correspondence , the value of the parameter bit RD1 or RD2 will be 1.

Machine tools of head-head configuration

Those kind of machines are called machine tools of head-head configuration, on which both rotary axes rotate the tool.

If the axis of rotation of the tool is parallel with the axis Z, the following arrangements can occur:

The first axis is C, the second axis is A

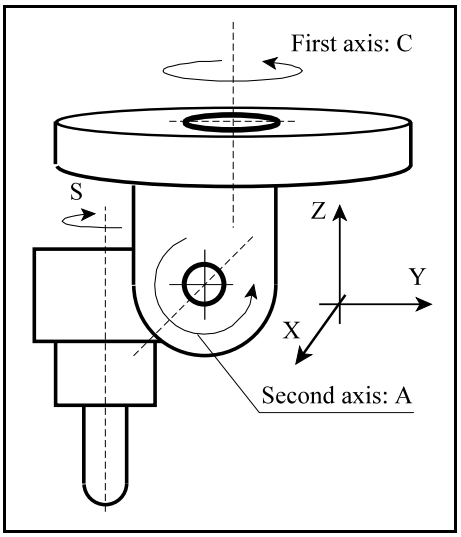


Fig. 25.1-5

Mechanically, the axis C bears the axis A.

The first axis is C, the second axis is B

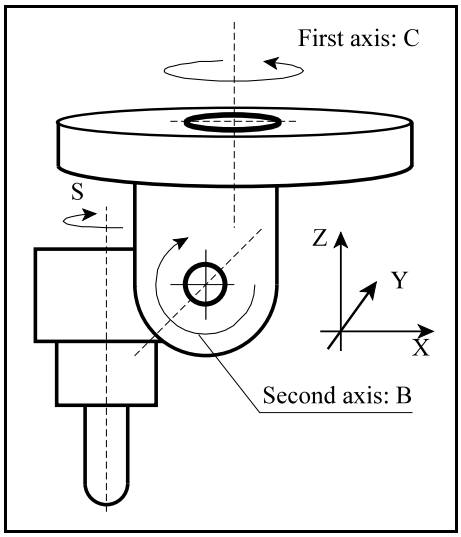


Fig. 25.1-6

Mechanically, the axis C bears the axis B.

The first axis is A, the second axis is B

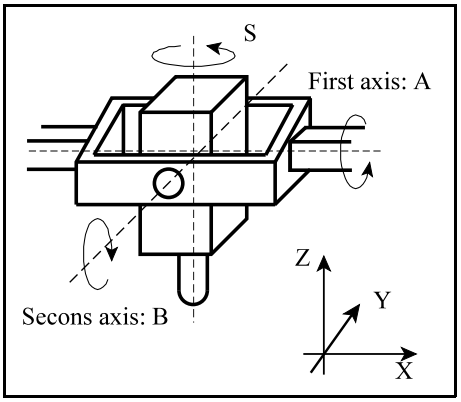


Fig. 25.1-7

Mechanically, the axis A bears the axis B.

The first axis is B, the second axis is A

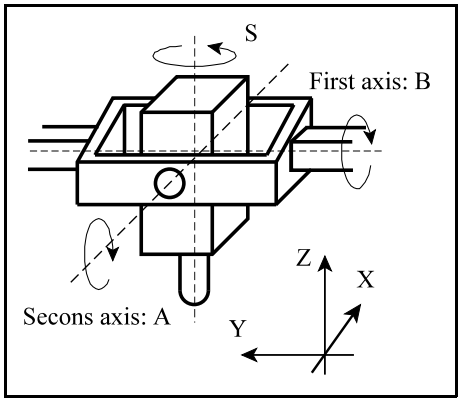


Fig. 25.1-8

Mechanically, the axis B bears the axis A.

The figure illustrates a machine tool of head-head configuration equipped with inclined axis of rotation. The axis of rotation of the rotary axis 1 (B) is parallel with the Y axis, therefore the value of the parameter N3205 Direction of the First Rot. Ax. is 2. The axis of rotation of the rotary axis 2 (C) is inclined with respect to the Y axis in the YZ plane, therefore the value of the parameter N3209 Direction of the Second Rot. Ax. is 5.
The axis of rotation of the C axis inclines at the angle 45° with respect to the Y axis, therefore the value of the parameter N3209 Direction of the Second Rot. Ax. is 45.

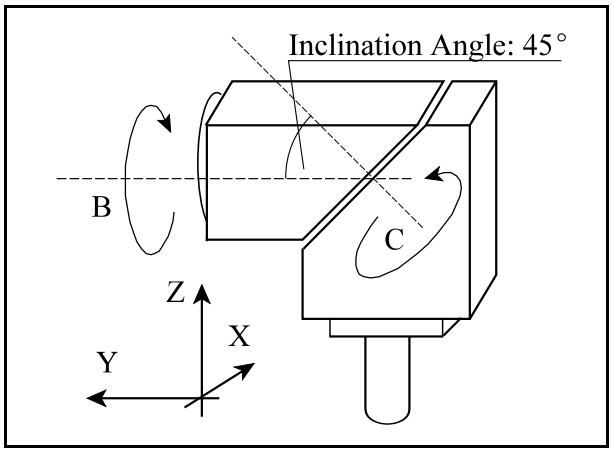


Fig. 25.1-9

The first and the second rotary axes do not have to intersect neither each other nor the axis of rotation of the tool (the spindle).

Starting from ***the tool holding point***, ***the distance to the axis of rotation of the second rotary axis*** can be specified using the following parameters:
N3219 Offs. between Tool Holder and 2nd Rot. Ax. X
N3220 Offs. between Tool Holder and 2nd Rot. Ax. Y
N3221 Offs. between Tool Holder and 2nd Rot. Ax. Z

The distance between the axes of rotations of the second and the first rotary axes are specified by the following parameters:
N3222 Offs. between the 2nd and 1st Rot. Ax. X
N3223 Offs. between the 2nd and 1st Rot. Ax. Y
N3224 Offs. between the 2nd and 1st Rot. Ax. Z

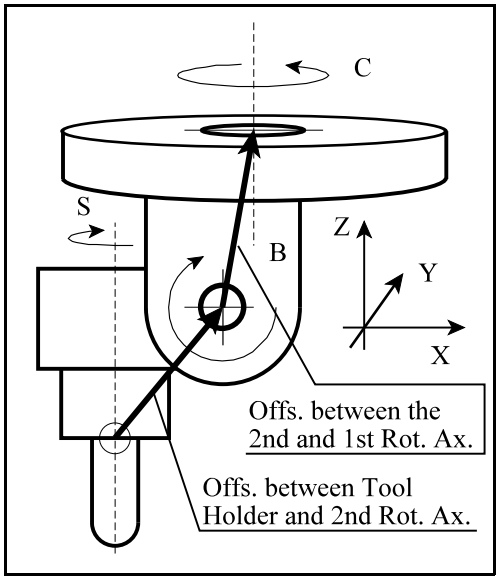


Fig. 25.1-10

The distances mentioned above are measured by the builder of the machine tool.

Machine tools of table-table configuration

Those kind of machines are called machine tools of table-table configuration, on which both rotary axes rotate the workpiece.

If the axis of rotation of the tool is parallel with the axis Z, the following arrangements can occur:

The first axis is A, the second axis is C

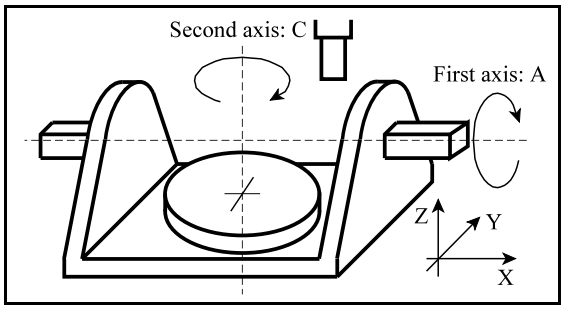


Fig. 25.1-11

Mechanically, the axis A bears the axis C.

The first axis is B, the second axis is C

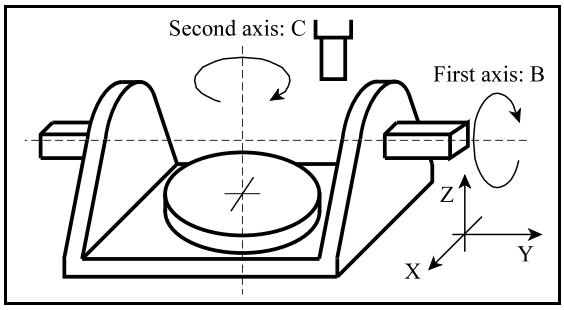


Fig. 25.1-12

Mechanically, the axis B bears the axis C.

The first axis is A, the second axis is B

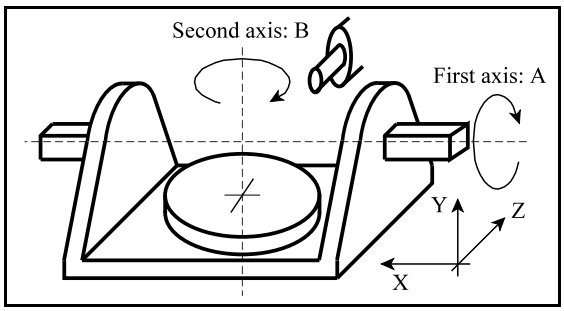


Fig. 25.1-13

Mechanically, the axis A bears the axis B.

The figure illustrates a machine tool of table-table configuration equipped with inclined axis of rotation.
The axis of rotation of the rotary axis 1 (B) is inclined with respect to the Y axis in the YZ plane, therefore the value of the parameter N3205 Direction of the First Rot. Ax. is 5.
The axis of rotation of the rotary axis 2 (C) is parallel with the Z axis, therefore the value of the parameter N3209 Direction of the Second Rot. Ax. is 3.
The axis of rotation of the B axis inclines at the angle −45° with respect to the Y axis, therefore the value of the parameter N3206 Inclination Angle of the First Rot. Ax. is −45.

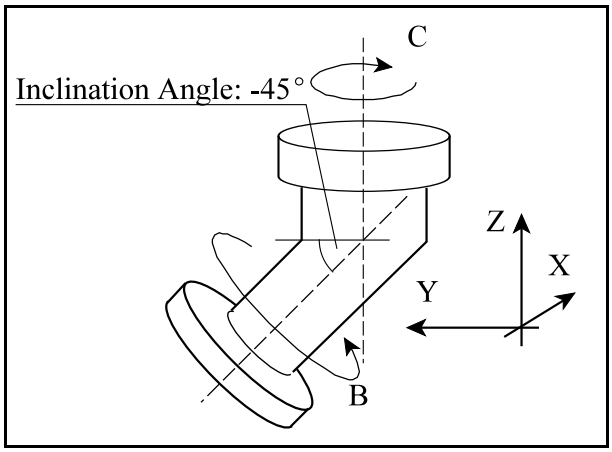


Fig. 25.1-14

The axes of rotation of the first and the second rotary axes do not have to intersect each other.

The position of the axis of rotation of the first rotary axis relative to the origin of the machine coordinate system is specified by the following parameters:
N3213 Rotary Table Pos. X
N3214 Rotary Table Pos. Y
N3215 Rotary Table Pos. Z

The distance between the axes of rotation of the first and the second rotary axes is specified by the following parameters:
N3216 Offset of 2nd Rotary Table X
N3217 Offset of 2nd Rotary Table Y
N3218 Offset of 2nd Rotary Table Z

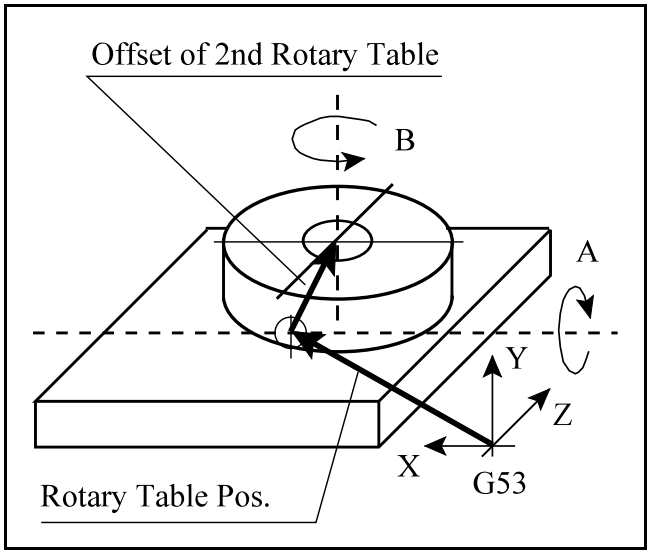


Fig. 25.1-15

The distances mentioned above are generally measured by the builder of the machine tool.

Machine tools of head-table configuration

Those kind of machines are called machine tools of head-table configuration, on which the first rotary axis rotates the tool, the second rotary axis rotates the workpiece.

☞ ***Note!*** *On the machine tools of head-table configuration, the parameters have to be set in such a way so that the axis rotating the tool will be the first rotary axis, and the axis rotating the workpiece will be the second rotary axis!*

If the axis of rotation of the tool is parallel with the axis Z, the following arrangements can occur:

The first axis is A, the second axis is C

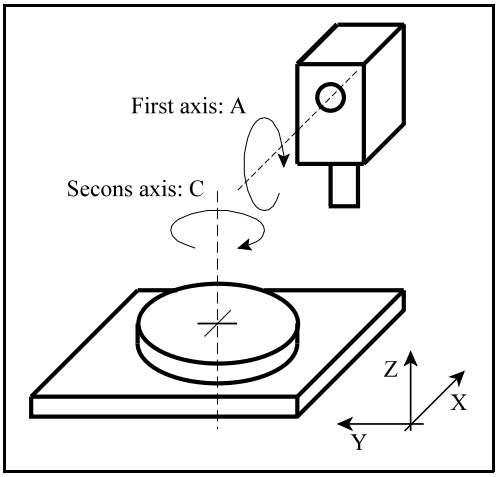


Fig. 25.1-16

The axis A rotates the tool, the axis C rotates the workpiece.

The first axis is B, the second axis is C

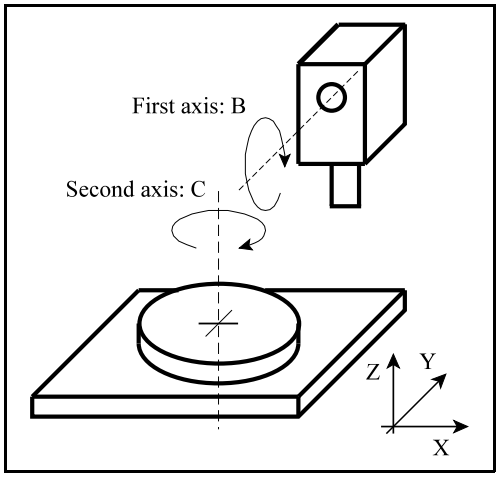


Fig. 25.1-17

The axis B rotates the tool, the axis C rotates the workpiece.

The first axis is A, the second axis is B

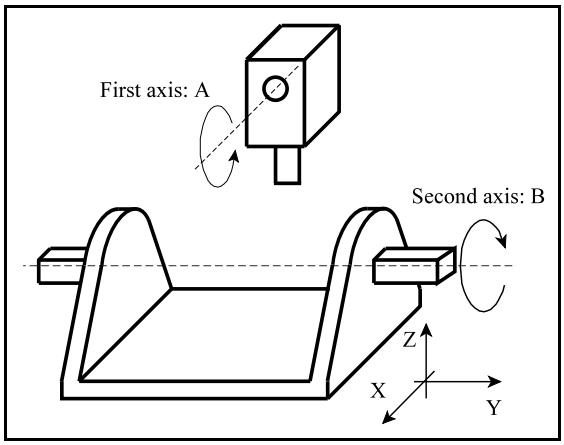


Fig. 25.1-18

The axis A rotates the tool, the axis B rotates the workpiece.

The first axis is B, the second axis is A

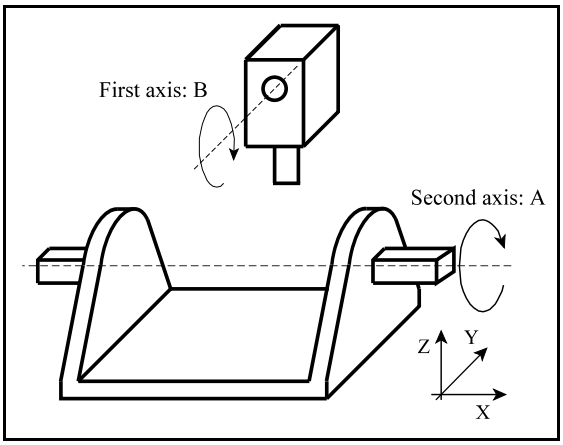


Fig. 25.1-19

The axis B rotates the tool, the axis A rotates the workpiece.

The figure illustrates a machine tool of head-table configuration equipped with inclined axis of rotation.
The axis of rotation of the rotary axis 1 (B) is inclined with respect to the Y axis in the YZ plane, therefore the value of the parameter N3205 Direction of the First Rot. Ax. is 5.
The axis of rotation of the rotary axis 2 (C) is parallel with the Z axis, therefore the value of the parameter N3209 Direction of the Second Rot. Ax. is 3.
The axis of rotation of the B axis inclines at the angle 45° with respect to the Y axis, therefore the value of the parameter N3206 Inclination Angle of the First Rot. Ax. is 45.

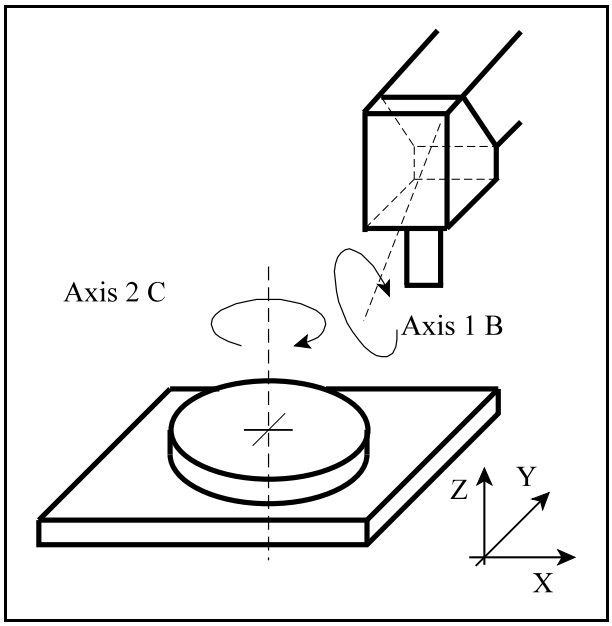


Fig. 25.1-20

On the machine tools of head-table configuration, the first rotary axis rotates the tool. Starting from ***the tool holding point***, ***the distance to the axis of rotation of the first rotary axis*** can be specified using the following parameters:
N3219 Offs. between Tool Holder and 2nd Rot. Ax. X
N3220 Offs. between Tool Holder and 2nd Rot. Ax. Y
N3221 Offs. between Tool Holder and 2nd Rot. Ax. Z

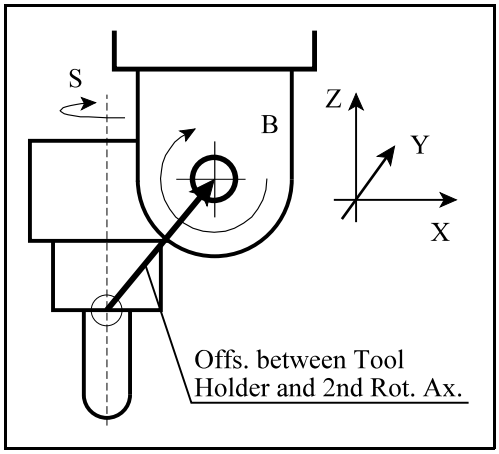


Fig. 25.1-21

On the machine tools of head-table configuration, the first rotary axis rotates the table. ***The position of the axis of rotation of the second rotary axis relative to the origin of the machine coordinate system*** is specified by the following parameters:
N3213 Rotary Table Pos. X
N3214 Rotary Table Pos. Y
N3215 Rotary Table Pos. Z

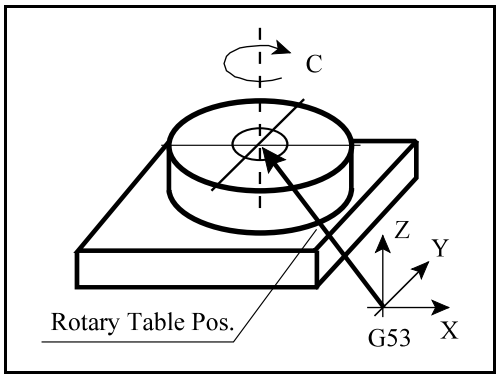


Fig. 25.1-22

The distances mentioned above are measured by the builder of the machine tool.

25.2 Length Compensation in the Tool Axis Direction (G43.1)

If, on a 5-axis machine tool, the tool is rotated by both rotary axes or one of them, the length compensation has to be taken into account in the tool axis direction. Such machines are the machine tools of head-head or head-table configuration.
If, when this function is switched on, one of the rotary axes or both of them are being moved, the tool tip will move in the space X, Y, Z. In order that the tool tip will be at the same position at the end of the rotation where it was at start, the axes X, Y and Z have to be moved too.
The instruction

G43.1 Hn

switches on the length compensation in the tool axis direction.
The instruction

G49

switches off the length compensation in the tool axis direction.
Depending on the bit #7 FOS of the parameter N3212 5D Control, there are two ways to take tool-directional length compensation into account. If the value of the bit #7 FOS:

=0: always with motion, i.e. the linear axes will move even if motion is programmed only for the rotary axes; in other words, whenever the compensation changes due to ***the change of the direction of the compensation vector*** (***rotation of the rotary axes***), this will always be ***followed by motion along the axes X, Y and Z***.

=1: with transformation, i.e. if displacement is programmed only for the rotary axes, the change of the position of the rotary axes will not be followed by motion along the axes X, Y and Z.

The following examples are related to such a machine tool, on which, in the zero degree position of the first rotary axis C and the second rotary axis A, the direction of the tool is Z. For the explanation, the parameter setting #7 FOS=0 is assumed.
Example 1:

```
G54 G49 G0 G90 X0 Y0 Z100 A0 C0
G43.1 Z100 H1  (A=0, C=0: The length compensation is taken
               into account along the axis Z, the end
               position is X0, Y0, Z100)
A45       (Tilting the tool by 45 degrees, motion along
          the axes Y and Z, the end position is X0, Y0,
          Z100, A45, C0)
```

Example 2:

```
G54 G49 G0 G90 X0 Y0 Z100 A0 C0
G43.1 A30 C60 H1    (All the axes X, Y and Z are moving,
                    the end position is X0, Y0, Z100, A30,
                    C60)
A45       (Tilting the tool by 45 degrees, motion along
          the axes X, Y and Z, the end position is X0, Y0,
          Z100, A45 C60)
```

Example 3:

```
G54 G49 G0 G90 X0 Y0 Z100 A0 C0
G43.1 H1  (A=0, C=0: The length compensation is taken into
          account along the axis Z, the end position is
          X0, Y0, Z[100-H1] where H1 is the value of
          compensation cell H1)
```

If, in the state G43.1, positioning, linear or circular interpolation is programmed, for example

G0 G43.1 H X Y Z Θ Ψ

G1 X Y Z Θ Ψ,

the control point will be moved along the appropriate path, and not the tool center point.

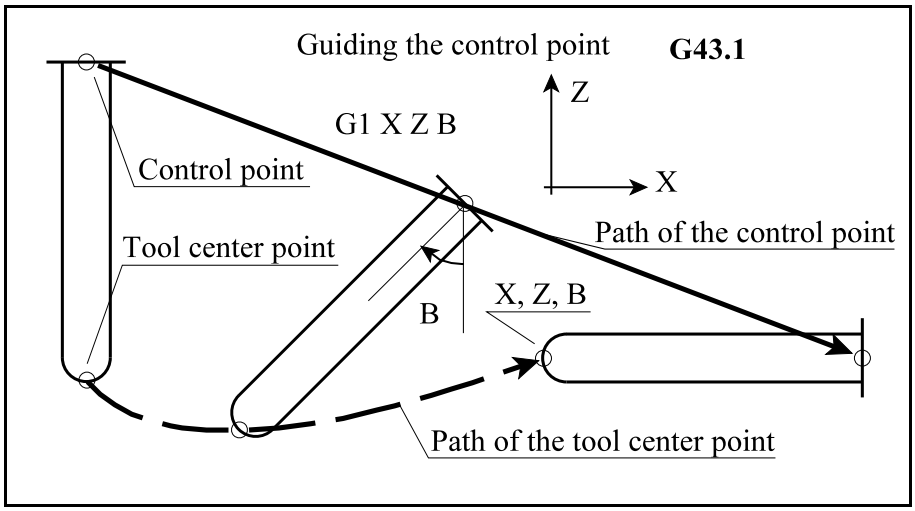


Fig. 25.2-1

The following figure shows such a machine tool of head-head configuration, on which the first rotary axis C rotates the tool around the axis Z, and the second rotary axis B tilts the tool around the axis Y.

In the zero degree machine position of the axes B and C, which is illustrated on the left side of the figure, the direction of the length compensation vector (**V**) is Z, and its magnitude is equal to the distance between the tool center point (tool tip) and the tool holding point. In this case, the control point coincides with the tool holding point.

Let us ***tilt the axis B to the 15 degree machine position***. This situation is illustrated on the right side of the figure. In this case, the length compensation vector (**V**) turned from the direction Z and its length has changed. The control point and the tool holding point move away each other. The magnitude of moving away depends on the value of the distance between the tool holding point and the second rotary axis, and the value of the distance between the second and the first rotary axes, given in parameter.
By the end of the motion of the axis B, the position of the tool center point measured in the workpiece coordinate system will be the same as it was prior to rotation of the axis B.

In the course of tilting the tool, the tool center point (the tool tip) moves.

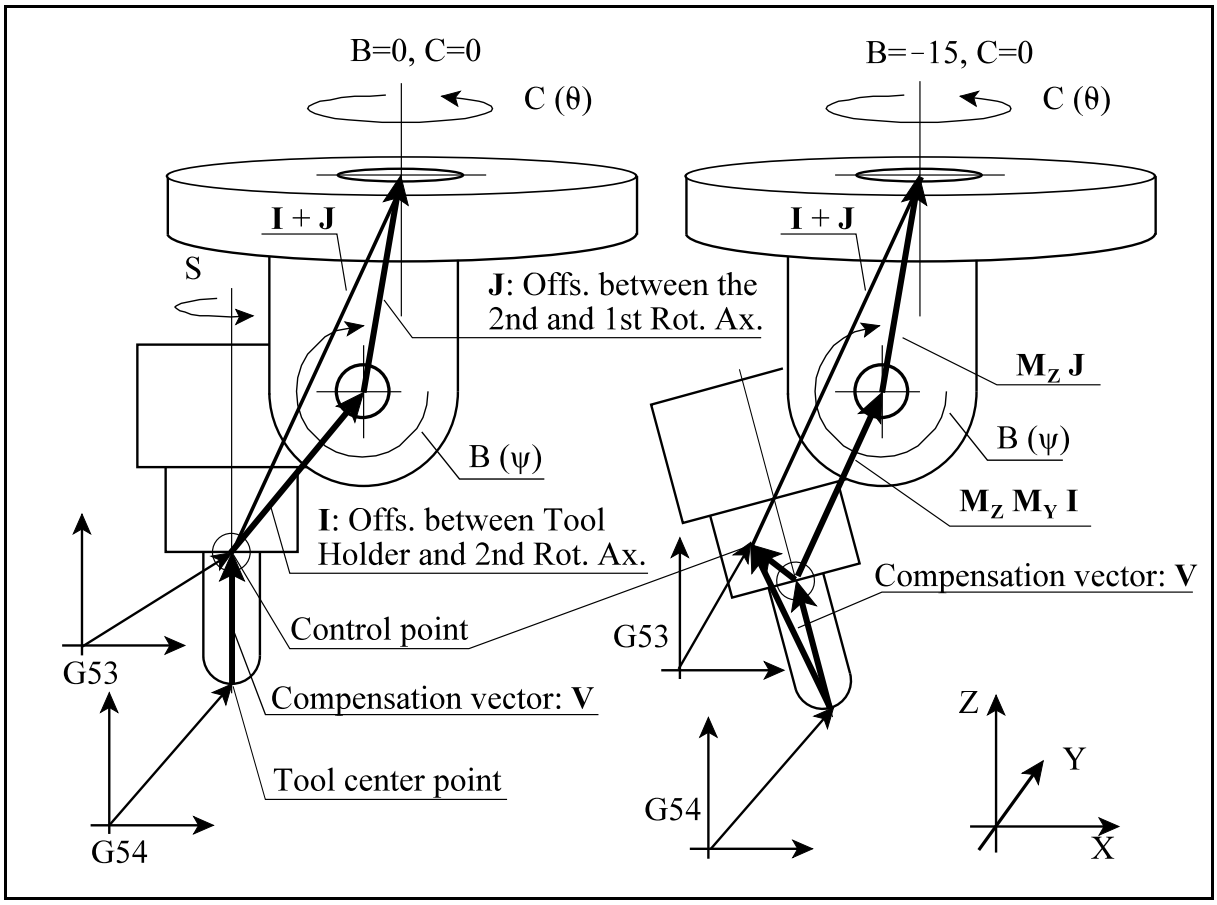


Fig. 25.2-2

Equations of length compensation calculation

The compensation vector **V** is:

$$\mathbf{V} = \mathbf{M_1}(\theta)\,\mathbf{M_2}(\psi)\,\mathbf{H} + \mathbf{M_1}(\theta)\,\mathbf{M_2}(\psi)\,\mathbf{I} + \mathbf{M_1}(\theta)\,\mathbf{J} - [\,\mathbf{I} + \mathbf{J}\,]$$

where

I=[I_X, I_Y, I_Z]: vector specified in the parameters N3219...3221 Offs. between Tool Holder and 2nd Rot. Ax.;

J=[J_X, J_Y, J_Z]: vector specified in the parameters N3222...3224 Offs. between the 2nd and 1st Rot. Ax.;

H=[0, 0, H_Z]: compensation from the compensation table taken into account in accordance with the parameter N3201 Tool Axis Direction;

V=[V_X, V_Y, V_Z]: length compensation vector;

θ: angular position of the first rotary axis;

ψ: angular position of the second rotary axis;

$\mathbf{M_1}$: rotation matrix related to the first rotary axis;

$\mathbf{M_2}$: rotation matrix related to the second rotary axis.

Depending of rotation by the first and the second rotary axes are realized around which longitudinal axis, the transformation matrices $\mathbf{M_1}$ and $\mathbf{M_2}$ will be the following:

$$M_X = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi & -\sin\varphi \\ 0 & \sin\varphi & \cos\varphi \end{bmatrix} \quad M_Y = \begin{bmatrix} \cos\varphi & 0 & \sin\varphi \\ 0 & 1 & 0 \\ -\sin\varphi & 0 & \cos\varphi \end{bmatrix} \quad M_Z = \begin{bmatrix} \cos\varphi & -\sin\varphi & 0 \\ \sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

25.3 Three-dimensional Coordinate Transformation (G68.1)

Three-dimensional coordinate transformation is available for shifting the workpiece coordinate system and rotating the coordinate system around a given axis. It is necessary in the case, when machining is executed not in a plane parallel with a main plane but in an arbitrary plane in space.
In this case, the part program is written in usual way, for example in the plane XY, and then, the program written in such a way will be shifted and rotated in space using the three-dimensional coordinate transformation.

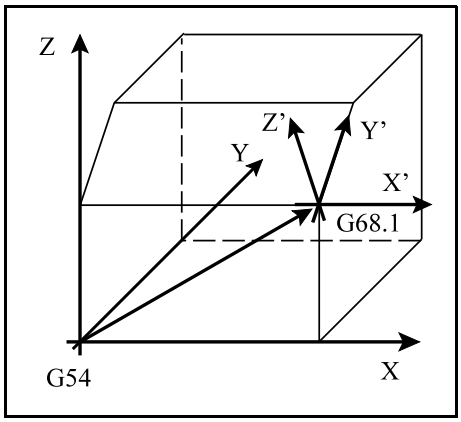


Fig. 25.3-1

The instruction

G68.1 X Y Z I J K R

switches on the three-dimensional coordinate transformation, where

X, Y, Z: coordinates of shifting the coordinate system in the workpiece coordinate system, given as absolute value;

I, J, K: assigning the axis around which rotation is to be done. The address I, J and K assigns rotation around the axis X, Y and Z, respectively. Only one among I, J and K can have the value 1, the others have the value 0. For example, if rotation is to be done around the axis Y, it will be I0 J1 K0.

R: angle of rotation, given in degree. The sign of rotation has to be specified in accordance with the right-hand rule.

The instruction

G69.1

switches off the three-dimensional coordinate transformation.
In the part program, not more than 2 three-dimensional coordinate transformations can be given:

```
...
N1 G68.1 Xx1 Yy1 Zz1 Ii1 Jj1 Kk1 Rr1
N2 G68.1 Xx2 Yy2 Zz2 Ii2 Jj2 Kk2 Rr2
...
G69.1
...
```

In the figure, ***X, Y and Z mark the workpiece coordinate system***.
The transformation defined in the block N1 of the example above shifts the origin by the vector $\mathbf{v_1}[x_1, y_1, z_1]$ and rotates the coordinate system around the axis Z' by the angle r_1. In such a way, ***the coordinate system X', Y', Z'*** is produced. It was assumed, that I0 J0 K1 was programmed in the block N1.
If, in the block N1, shifting is not programmed at the addresses X, Y and Z, the origin of the coordinate system X', Y', Z' and the origin of the coordinate system X, Y, Z will be the same.
Data of the block N2 will already be effective in the coordinate system X', Y', Z' transformed in the block N1.

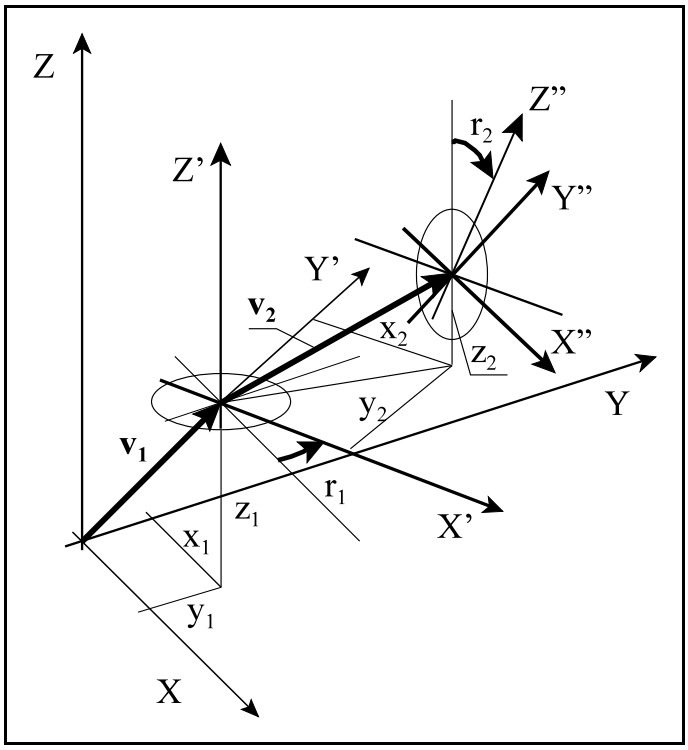


Fig. 25.3-2

The transformation defined in the block N2 of the example above shifts the origin of the coordinate system X', Y', Z' by the vector $\mathbf{v_2}[x_2, y_2, z_2]$ and rotates the coordinate system around the axis Y" by the angle r_2.In such a way, ***the coordinate system X", Y", Z"*** is produced. It was assumed, that I0 J1 K0 was programmed in the block N2.
If, in the blockN2, shifting is not programmed at the addresses X, Y and Z, the origin of the coordinate system X", Y", Z" and the origin of the coordinate system X', Y', Z' will be the same.
Then, the programmer coordinates X, Y and Z written in the program will already be executed in the coordinate system X", Y", Z".

Equations of the three-dimensional coordinate transformation

In the case of one rotation, the equation is:

$$\mathbf{V} = \mathbf{M_1}(\alpha)\ \mathbf{v} + \mathbf{v_1}$$

In the case of two rotations, the equation is:

$$\mathbf{V} = \mathbf{M_1}(\alpha)\ \mathbf{M_2}(\beta)\ \mathbf{v} + \mathbf{M_1}(\alpha)\ \mathbf{v_2} + \mathbf{v_1}$$

where

$\mathbf{v}[x, y, z]$: position given in the programmer coordinate system;
$\mathbf{v_1}[x_1, y_1, z_1]$: coordinates of shifting given in the first instruction G68.1;
$\mathbf{v_2}[x_2, y_2, z_2]$: coordinates of shifting given in the second instruction G68.1;
α: angle of rotation given in the first instruction G68.1;
β: angle of rotation given in the second instruction G68.1;
$\mathbf{M_1}$: rotation matrix related to the axis given in the first instruction G68.1;
$\mathbf{M_2}$: rotation matrix related to the axis given in the second instruction G68.1;
$\mathbf{V}[X, Y, Z]$: coordinates produced in the workpiece coordinate system after transformation.

The rotation matrices are the following:

$$M_X = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi & -\sin\varphi \\ 0 & \sin\varphi & \cos\varphi \end{bmatrix} \quad M_Y = \begin{bmatrix} \cos\varphi & 0 & \sin\varphi \\ 0 & 1 & 0 \\ -\sin\varphi & 0 & \cos\varphi \end{bmatrix} \quad M_Z = \begin{bmatrix} \cos\varphi & -\sin\varphi & 0 \\ \sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

where

$\mathbf{M_X}$: matrix of rotation around the axis X (when I1 J0 K0 is programmed);
$\mathbf{M_Y}$: matrix of rotation around the axis Y (when I0 J1 K0 is programmed);
$\mathbf{M_Z}$: matrix of rotation around the axis Z (when I0 J0 K1 is programmed);
φ: angle of rotation.

The multiplication sequence of rotation matrices is determined by ***the specification sequence of two succeeding rotations G68.1 executed around different axes***. Since the sequence of matrix multiplications cannot be interchanged (the matrix multiplication is not commutative), changing the sequence will result in different outcome.

25.4 Rotary Table Dynamic Zero Point Offset (G54.2 P)

If, on a machine tool of table-table or head-table configuration, the zero points have been measured relative to the axis of rotation of the table(s) in an arbitrary position of the table(s), the programmer will not have to care for change of zero point vectors during rotation of the table(s), because it will be calculated automatically by the control. The program is always written by the programmer for the axes parallel with the machine coordinate system. This function is called rotary table zero point offset.

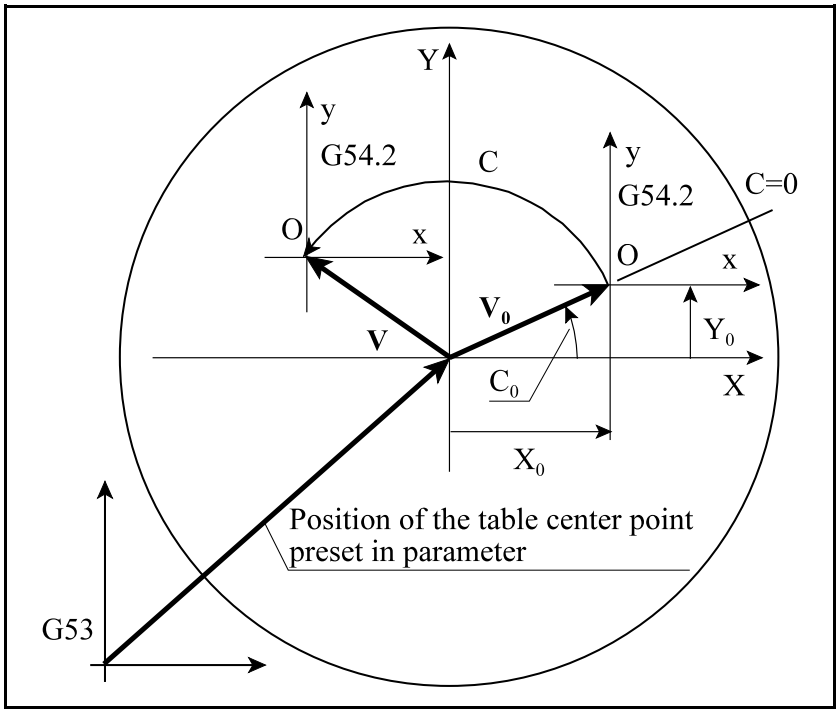


Fig. 25.4-1

The appropriate dynamic zero point can be called from the part program by the instruction

G54.2 Pp

where

p=1, ..., 8

This instruction, at the same time, switches on the function of dynamic zero point offset, which means, that ***if the position of the first and the second rotary axes specified in parameter changes, the value of the zero point vector on the main axes will be recalculated***. The G54.2 belongs to the #14 group of codes G.

The instruction G54.2 can be used or the table of dynamic zero point offset will seen in a channel if the value of the parameter N3200 Mechanical Type is 2 (***machine tool of table-table configuration***) or 3 (***machine tool of head-table configuration***).
The linear axes participating in dynamic zero point offset are the linear main axes defined in the parameter N0103 Axis to Plane (=1 X, =2 Y, =3 Z).
If the value of the parameter N3200 Mechanical Type is 2 (machine tool of table-table configuration), two rotary axes will participate in dynamic zero point offset, the number of which can be specified in the parameters N3204 No. of the First Rot. Ax. and N3208 No. of

the Second Rot. Ax..
If the value of the parameter N3200 Mechanical Type is 3 (machine tool of head-table configuration), one rotary axis will participate in dynamic zero point offset, which can be specified in the parameter N3208 No. of the Second Rot. Ax..
Eight different dynamic zero point offset can be set in the control.

By the instruction
G54.2 Pp X Y Z Θ Ψ
the pth dynamic zero point will be changed, the zero point related to the endpoint position of the first and second rotary axes (Θ, Ψ) will be calculated, and the axes X, Y and Z will already be positioned in the converted coordinate system.

When dynamic zero point offset is called, i.e. when G54.2 part program is programmed,
Θ Ψ
G54.2 Pp
and , in the state G54.2, when rotary axes involved in dynamic zero point offset are moved,
G54.2 Pp
Θ Ψ
if the value of the bit #7 FOS of the parameter N3212 5D Control:
=0: ***the linear axes*** involved in dynamic zero point offset will always move ***to the new dynamic position taking the endpoint position of the rotary axes into account;***
=1: ***the linear axes*** involved in dynamic zero point offset ***will not move,*** and the instruction will result only in ***zero point change***. ***The linear axes moves to the dynamic position only after a positioning block***.
If the parameter is set for zero point change by motion and not the interpolation code G0 or G1 is valid when G54.2 is called, the control will execute motion using the code G0 temporarily.

If, in the switch-on state of the dynamic zero point offset, linear interpolation together with table rotation is programmed, the path of the tool center point will coincide with the straight line fixed to the table only at the beginning and at the end of the bloc, during motion it will deviate from the line.
In the figure, a situation of a horizontal machine tool is illustrated, when. In the state G54.2, motions X, Y and B are programmed in one block. Together with rotation B, the workpiece fixed to the table will also rotate. Interpolation occurs not in a coordinate system fixed

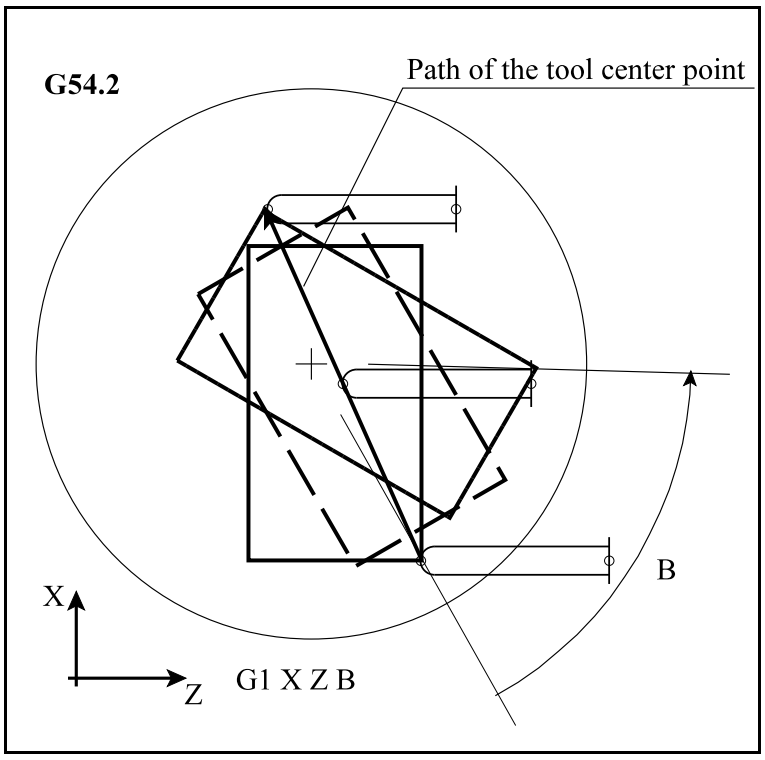


Fig. 25.4-2

to the table and rotating continuously, so the tool center point will coincide with the straight line only at the beginning and at the end of the bloc.

The codes G28, G30 and G53 can suspend managing the dynamic zero point offset .
In the case of the codes above, motion will always executed to the appropriate machine position.

Calculation of the origin of the dynamic zero point offset

Let the dynamic zero point offset be as follows:

$\mathbf{V_0}[X_0, Y_0, Z_0]$: the values of the dynamic zero point offset for the linear axes;

Θ_0, Ψ_0: the values of the dynamic zero point offset for the first and the second rotary axes.

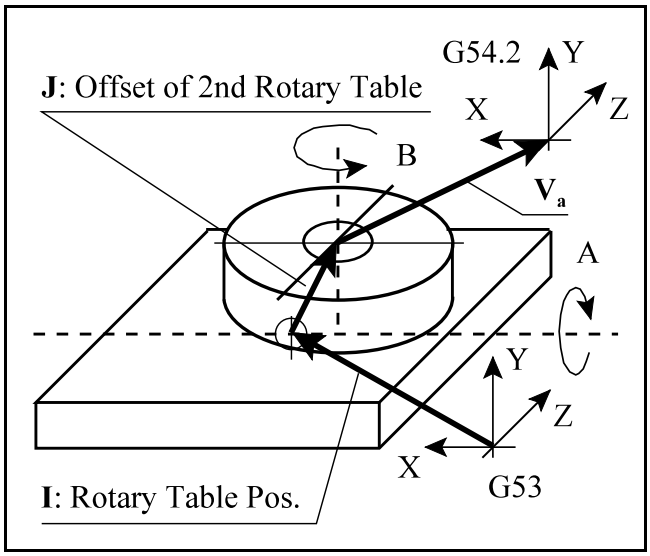


Fig. 25.4-3

The first step is to calculate the offset vector

$\mathbf{V_a}[X_a, Y_a, Z_a]$

related to the machine position angles $\boldsymbol{\Theta_a=0}$ and $\boldsymbol{\Psi_a=0}$ from the following equation:

$$\mathbf{V_a} = \mathbf{M_2}(-\Psi_0)\ \mathbf{M_1}(-\Theta_0)\ \mathbf{V_0}$$

where

$\mathbf{M_1}$: transformation matrix related to the first rotary axis;
$\mathbf{M_2}$: transformation matrix related to the second rotary axis.

The second step is to calculate the offset vector
V
related to the new angles θ és ψ of the rotary axes given in the programmer coordinate system. Its values in the machine coordinate system are:

$$\mathbf{V} = \mathbf{M_1}(\theta + \Theta_0)\ \mathbf{M_2}(\psi + \Psi_0)\ \mathbf{V_a} + \mathbf{M_1}(\theta + \Theta_0)\ \mathbf{J} + \mathbf{I}$$

where $\mathbf{V}[X, Y, Z]$: the offset vector related to the angles θ és ψ;
$\mathbf{I}[I_X, I_Y, I_Z]$: the values of the parameters Rotary Table Pos;
$\mathbf{J}[J_X, J_Y, J_Z]$: the values of the parameters Offset of 2nd Rotary Table;
$\mathbf{M_1}$: transformation matrix related to the first rotary axis;
$\mathbf{M_2}$: transformation matrix related to the second rotary axis.

Depending of rotation by the first and the second rotary axes are realized around which longitudinal axis, the transformation matrices M_1 and M_2 will be the following:

$$M_X = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi & -\sin\varphi \\ 0 & \sin\varphi & \cos\varphi \end{bmatrix} \quad M_Y = \begin{bmatrix} \cos\varphi & 0 & \sin\varphi \\ 0 & 1 & 0 \\ -\sin\varphi & 0 & \cos\varphi \end{bmatrix} \quad M_Z = \begin{bmatrix} \cos\varphi & -\sin\varphi & 0 \\ \sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Managing the dynamic zero point offset on the machine tools of head-table configuration is similar, with this difference that only one rotary axis will participate in creating the zero point.

In managing the dynamic zero point, the positive direction of rotation transformation is according to the right-handed rule. In many cases, the positive direction of rotation of the table is opposite to the right-handed rule. ***The dynamic zero point offset will work correctly in the case if the bits RD1 and RD2 of the parameter N3212 5D Control is set in accordance with the direction of rotation of the table***. See the section 25.1 **Construction of 5-axis Machine Tools** on page 356.

25.4.1 Setting the Dynamic Zero Point Offsets (G10 L21)

Setting can be done by the instruction

G10 L21 P v

where

P = 1...8 selection of the dynamic workpiece coordinate system **G54.2 P1, G54.2 P2, ..., G54.2 P8;**

v (X, Y, Z, A ...): the value of offset for each axis.

The linear axis offsets are always read in as orthogonal values.

In absolute data specification instruction state **G90**, the value written at the coordinate addresses gets to the appropriate offset register.

In incremental data specification instruction state **G91** or when operator I is used, the value written at the addresses will be added to the content of the appropriate offset register.

25.5 Examples of Application of the Functions G43.1, G68.1 and G54.2

Let us machine the three upper faces of a tetrahedron with edges of 100 mm. (Tetrahedron is a body all four faces of which are equilateral triangles.)
Let the workpiece zero point be at the top corner of the tetrahedron. The tool has to go round the triangle and the circle inscribed in the triangle, and drill a hole into the center of the triangle.

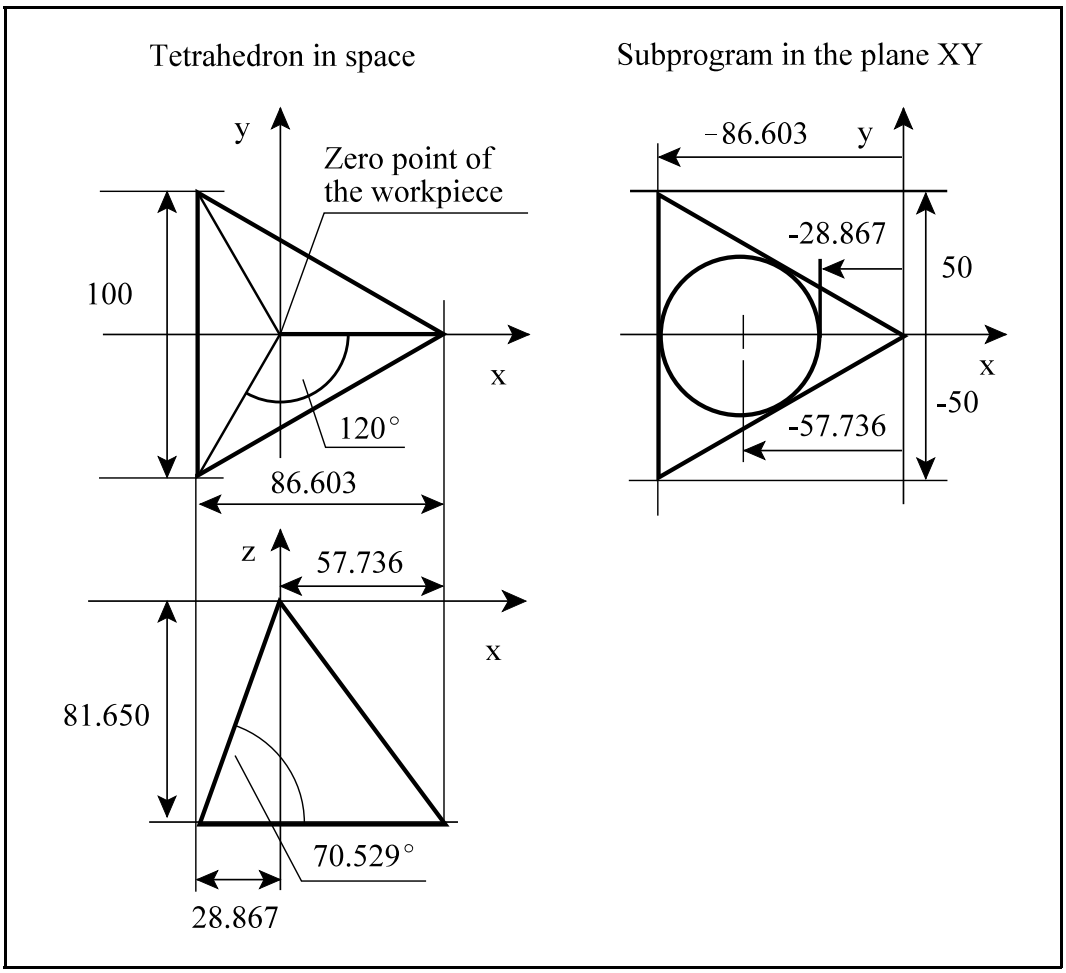


Fig. 25.5-1

Let the program machining the equilateral triangle be the subprogram O1000. The triangle spread on the plane XY is shown in the figure above. The subprogram O1000 is written according to this figure:

```
O1000 (machining one face of the tetrahedron)
N10 X20 Y0 Z-5
N20 G42 G1 X0 D1 F600 (going the triangle round)
N30 X-86.603 Y50
N40 Y-50
N50 X0 Y0
N60 G40 X20
N70 Z-2.5
```

```
N80 G42 X-28.867
N90 G3 I-28.867 (the circle inscribed in the triangle)
N100 G40 G0 X20 Z20
N110 G81 X-57.736 Y0 R2 Z-50 (drilling the hole)
N120 G80
N130 M99
```

It will be presented below how the main program has to be written.

The case of a machine tool of head-head configuration

Let the first rotary axis of the machine tool of head-head configuration be the axis C rotating the tool around the axis Z, and the second rotary axis be the axis B rotating the tool around the axis Y.
The faces will be machined in the sequence of numbers (1, 2 and 3) shown in the figure.
At first, for machining the face 1, the tool has to be tilted the degree of −70.529 around the axis Y using the axis B in order that the tool will be perpendicular to the plane of the triangle 1.
For machining the faces 2 and 3, the tool has to be tilted the degrees 120 and 240, respectively around the axis Z using the axis C.

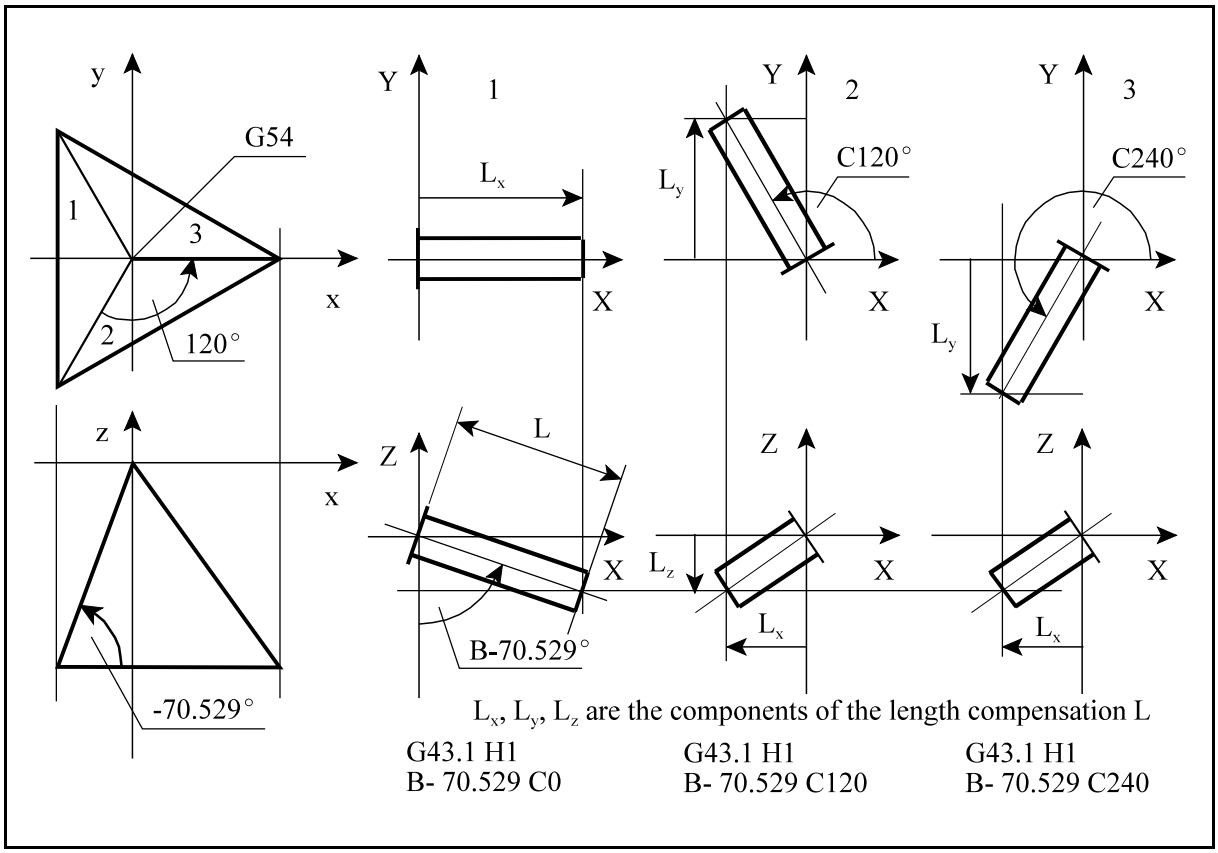


Fig. 25.5-2

Since it was the tool that has been rotated relative to the direction Z, this rotation has to be followed by the transformation G68.1 in order that the program written in the plane XY will be three-dimensional.

The main program is as follows:

```
(Tetrahedron)
N10 G0 G54 G17 G90 G49 Z100
N20 X0 Y0
N30 B-70.529 C0 (tool rotation to the plane 1)
N40 M3 S1000
N50 G43.1 H1 (length compensation in tool direction on)
N60 G68.1 X0 Y0 Z0 I0 J1 K0 R-70.529 (-70.529°rotation
     around the axis Y)
N70 (machining the triangle 1)
N80 M98 P1000
N90 G69.1 (three-dimensional rotation off)
N100 Z100
N110 X0 Y0
N120 C120 (tool rotation to the plane 2)
N130 G68.1 X0 Y0 Z0 I0 J0 K1 R120 (120°rotation around the
     axis Z)
N140 G68.1 X0 Y0 Z0 I0 J1 K0 R-70.529 (-70.529°rotation
     around the axis Y)
N150 (machining the triangle 2)
N160 M98 P1000
N170 G69.1 (three-dimensional rotation off)
N180 Z100
N190 X0 Y0
N200 C240 (tool rotation to the plane 3)
N210 G68.1 X0 Y0 Z0 I0 J0 K1 R240 ((240°rotation around
     the axis Z)
N220 G68.1 X0 Y0 Z0 I0 J1 K0 R-70.529 (-70.529°rotation
     around the axis Y)
N230 (machining the triangle 3)
N240 M98 P1000
N250 G69.1 (three-dimensional rotation off)
N260 Z100
N270 X0 Y0
N280 G49 (length compensation in tool direction off)
N290 Z100
N300 X0 Y0
N310 B0 C0
N320 M30
```

The case of a machine tool of table-table configuration

Let the first rotary axis of the machine tool of table-table configuration be the axis A rotating the workpiece around the axis X, and the second rotary axis be the axis C rotating the workpiece around the axis Z. ***The positive direction of rotation of the axes A and C is according to the left-handed rule, so parameter values RD1=RD2=1 has to be used.*** The faces will be machined in the sequence of numbers (1, 2 and 3) shown in the figure.

In the initial position of the workpiece the edge of the tetrahedron is parallel with the axis Y (it is marked with dashed line in the figure).
Rotation is possible around the axis X only, so the tetrahedron has to be rotated -90° using the axis C, an then it has to be tilted 70.529° using the axis A in order that the plane 1 will be perpendicular to the axis Z (to the tool).
The altitude line of of the face of the tetrahedron rotated in such a way will be parallel with the axis Y. The subprogram O1000 has been written for such a triangle where the altitude line is parallel with the axis X. So, 90° rotation around the axis Z has to be programmed using transformation G68.1.

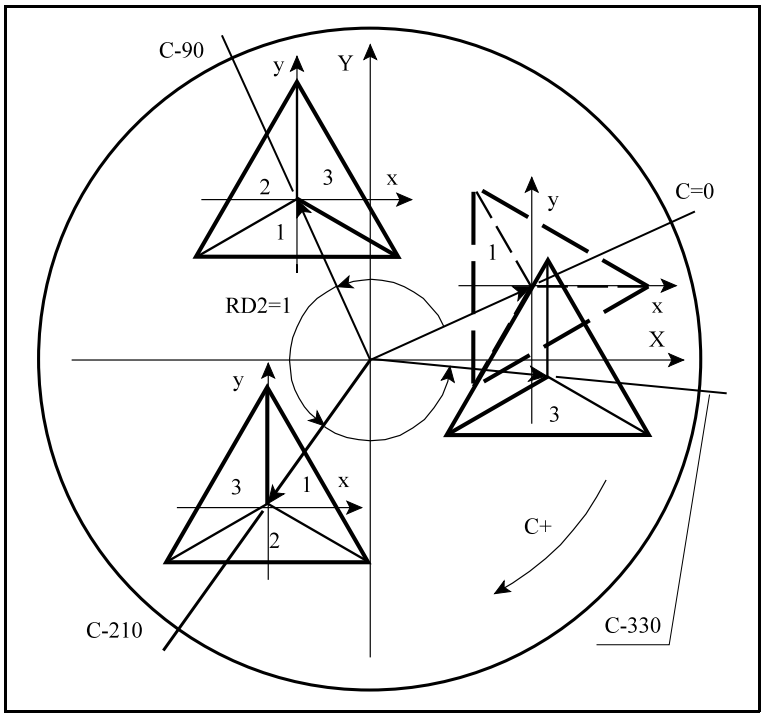


Fig. 25.5-3

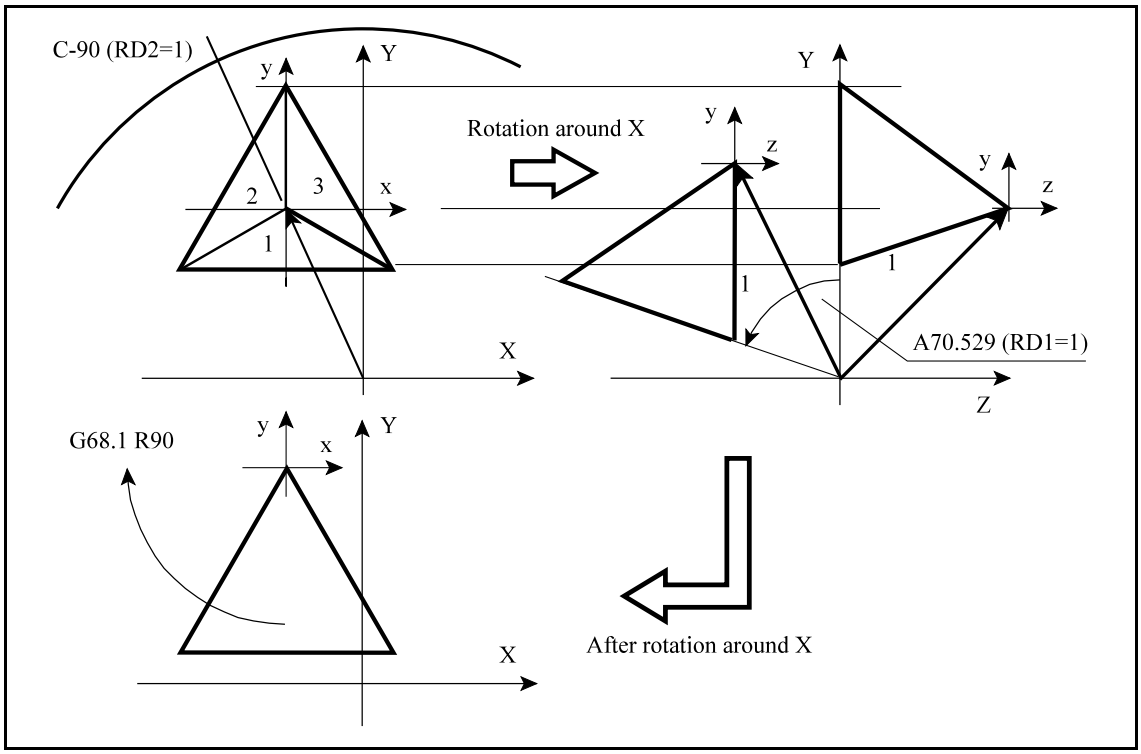


Fig. 25.5-4

Since it is the workpiece that is rotated, dynamic zero point is used in order not to deal with the position of the zero point vector rotating around the axes X and Z in space; it has to be calculated by the control automatically.

The main program is as follows:

```
(Tetrahedron, RD1=RD2=1)
N10 G0 G54 G17 G90
N20 A0 C0
N30 G49 Z100
N40 T1 M6
N50 G43 Z100 H1 (length compensation on)
N60 X0 Y0
N70 M3 S1000
N80 G54.2 P1 (dynamic zero point management on)
N90 A70.529 C-90 (tilting the cradle and rotating the
     table for machining the plane 1)
N100 G68.1 X0 Y0 Z0 I0 J0 K1 R90 (rotation around Z)
N110 M98 P1000
N120 Z100
N130 X0 Y0
N140 C-210 (rotating the table for machining the plane 2)
N150 M98 P1000 (machining)
N160 Z100
N170 X0 Y0
N180 C-330 (rotating the table for machining the plane 3)
N190 M98 P1000 (machining)
N200 Z100
```

```
N210 X0 Y0
N220 G69.1 (rotation off)
N230 G54 (dynamic zero point management off)
N240 G49 (length compensation off)
N250 Z100
N260 X-100 Y0
N270 A0 C0
N280 M30
```

The case of a machine tool of head-table configuration

Let the first rotary axis of the machine tool of head-table configuration be the axis B tilting the tool around the axis Y, and the second rotary axis be the axis C rotating the workpiece around the axis Z. ***The positive direction of rotation of the axis C is according to the left-handed rule, so parameter value RD2=1 has to be used.*** The faces will be machined in the sequence of numbers (1, 2 and 3) shown in the figure.

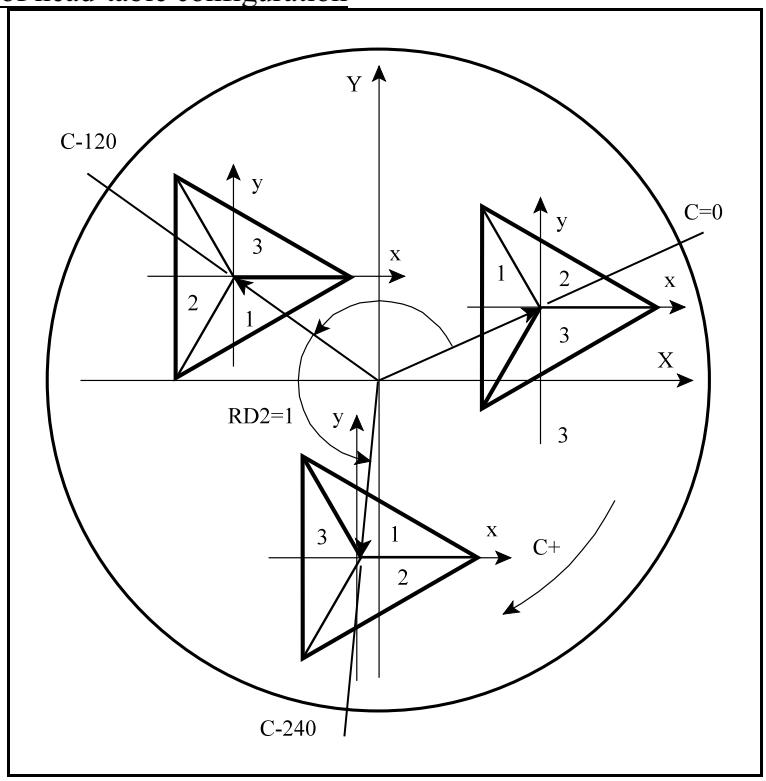


Fig. 25.5-5

At first, for machining the face 1, the tool has to be tilted the degree of −70.529 around the axis Y using the axis B in order that the tool will be perpendicular to the plane of the triangle 1.

For machining the faces 2 and 3, the workpiece has to be rotated the degrees −120 and −240, respectively around the axis Z using the axis C.

Since it was the tool that has been rotated relative to the direction Z, this rotation has to be followed by the transformation G68.1 in order that the program written in the plane XY will be three-dimensional.

Since the workpiece has to also be rotated in order that machining all the three faces of it will be possible, dynamic zero point G54.2 is used in order not to deal with the position of the zero point vector rotating around the axis Z in space; it has to be calculated by the control automatically.

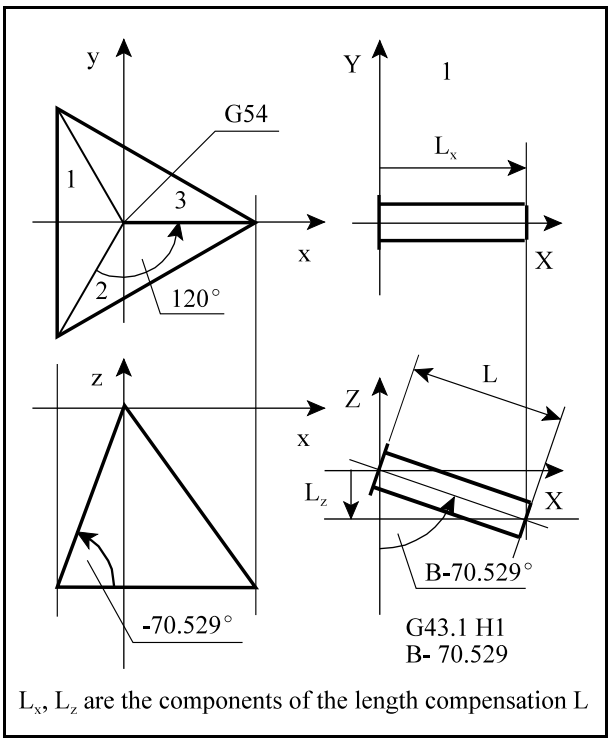


Fig. 25.5-6

The main program is as follows:

```
(Tetrahedron, RD2=1)
N10 G0 G54 G17 G90 G49 Z100
N20 X0 Y0
N30 B-70.529 (tilting the tool to the face of the tetrahedron)
N40 M3 S1000
N50 G43.1 H1 (length compensation in tool direction on)
N60 G54.2 P1 (dynamic zero point management on)
N70 C0 (rotating the table for machining the plane 1)
N80 G68.1 X0 Y0 Z0 I0 J1 K0 R-70.529 (rotation around the axis
Y)
N90 M98 P1000 (machining)
N100 Z100
N110 X0 Y0
N120 C-120 (rotating the table for machining the plane 2)
N130 M98 P1000 (machining)
N140 Z100
N150 X0 Y0
N160 C-240 (rotating the table for machining the plane 3)
N170 M98 P1000 (machining)
N180 G69.1 (three-dimensional rotation off)
N190 G54 (dynamic zero point management off)
N200 G49
N210 Z100
N220 X0 Y0
N230 B0 C0
N240 M30
```

25.6 Defining a Plane in Space by Giving Eulerian Angles (G68.2)

Using this instruction, a plane with arbitrary incline in space and a zero point offset can be defined in the workpiece coordinate system. Then, the program can be written in one of the main planes, e.g. in the plane XY in the usual way, provided that the direction of the tool is Z.

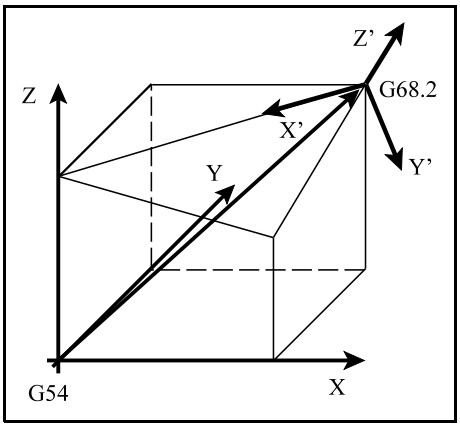


Fig. 25.6-1

By using the instruction

G68.2 Xx Yy Zz Iα Jβ Kγ

an arbitrary inclined plane can be defined in space, where

x, y, z: orthogonal coordinates of the origin of the coordinate system assigned to the plane, in the actual workpiece coordinate system;

α, β, γ: the Eulerian angles of the plane, in degree.

Interpretation of the Eulerian angles is as follows:

First rotation: rotation of the original coordinate system ***X, Y, Z around the axis Z by an angle α*** results in the coordinate system X', Y', Z.

Second rotation: rotation of the coordinate system X', Y', Z ***around the axis X' by an angle β*** results in the coordinate system X', Y", Z".

Third rotation: rotation of the coordinate system X', Y'', Z'' ***around the axis Z'' by an angle γ*** results in the final coordinate system $\mathbf{X_E}$, $\mathbf{Y_E}$, $\mathbf{Z_E}$.

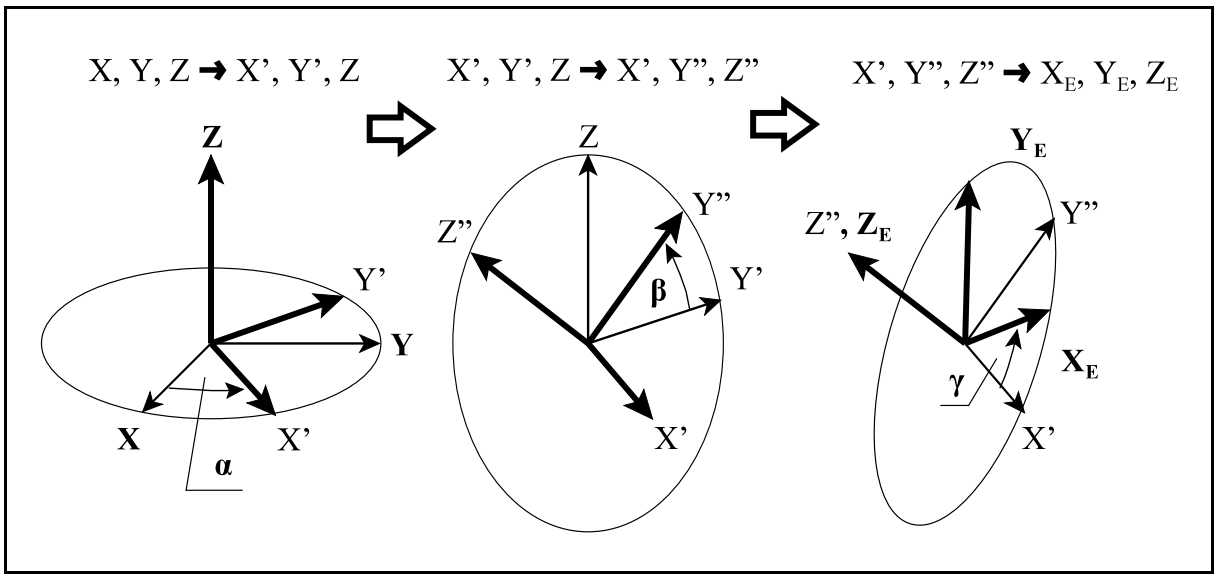


Fig. 25.6-2

Such Eulerian angles have to be specified so that the plane defined by them will be perpendicular to the axis Z after rotation. Then, the program can be written in the plane XY. ***The instruction G53.1*** moves the rotary axes available on the machine automatically to such position so that the plane will be perpendicular to the axis of the tool.

The instruction

G69.1

cancels defining a plane in space.

Equations of defining a plane in space using Eulerian angles

The instruction G68.2 is actually a three-dimensional coordinate transformation creating a new programmer coordinate system. The equation of transformation is the following:

$$\mathbf{V} = \mathbf{M_Z}(\alpha)\ \mathbf{M_X}(\beta)\ \mathbf{M_Z}(\gamma)\ \mathbf{v} + \mathbf{v_0}$$

where

v[x, y, z]: position specified in the coordinate system G68.2;
$\mathbf{v_0}[x_0, y_0, z_0]$: coordinates of the offset specified in the instruction G68.2;
α: angle of rotation around the axis Z specified at the address I;
β: angle of rotation around the axis X' specified at the address J;
γ: angle of rotation around the axis Z" specified at the address K;
$\mathbf{M_X}$: matrix of rotation related to the axis X;
$\mathbf{M_Z}$: matrix of rotation related to the axis Z;
V[X, Y, Z]: coordinates resulted in the workpiece coordinate system after transformation.

The matrices of rotation are the following:

$$M_X = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi & -\sin\varphi \\ 0 & \sin\varphi & \cos\varphi \end{bmatrix} \quad M_Z = \begin{bmatrix} \cos\varphi & -\sin\varphi & 0 \\ \sin\varphi & \cos\varphi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

An example for the application of the Eulerian angles:

Based on the figure, determine the coordinate system X ', Y', Z' by specifying the Euler angles:
The origin of the coordinate system X ', Y', Z'given in the coordinate system X, Y, Z:
(200; 0; 50)
The angle of the first rotation around the axis Z:
90°.
The angle of the second rotation around the axis X':
30°.
The angle of the third rotation around the axis Z':
0°
The program is as follows:

...
G68.2 X200 Y0 Z50 I90 J30 K0
G53.1 (or G53.6)
...

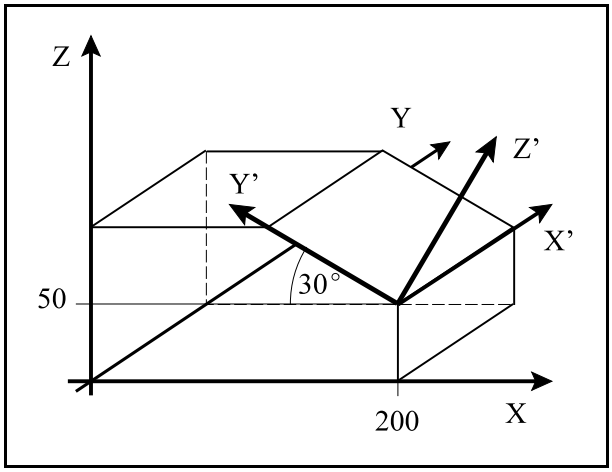


Fig. 25.6-3

25.7 Defining a Plane in Space by Rotation around Axes (G68.2 P1)

Using this instruction, a plane with arbitrary inclination in space and a zero point offset can be defined in the workpiece coordinate system. Then, the program can be written in one of the main planes, e.g. in the plane XY in the usual way, provided that the direction of the tool is Z.

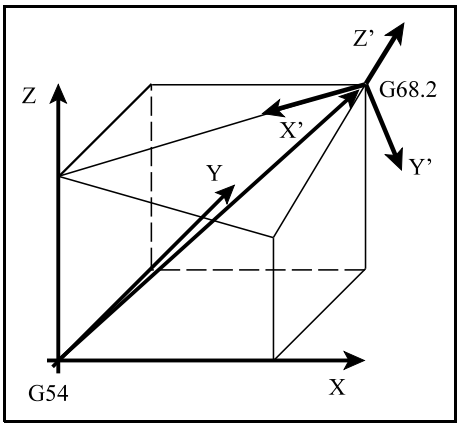


Fig. 25.7-1

By using the instruction

G68.2 P1 Qq Xx Yy Zz Iα Jβ Kγ

a plane with arbitrary inclination can be defined in space,

where

x, y, z: orthogonal coordinates of the origin of the coordinate system assigned to the plane, in the actual workpiece coordinate system;

α: the angle of rotation around the axis X (roll);

β: the angle of rotation around the axis Y (pitch);

γ: the angle of rotation around the axis Z (yaw), in degree.

The order of rotation can be specified at the address **Q**:

Value of Q	Axis of the rotation 1	Axis of the rotation 2	Axis of the rotation 3	Multiplication order of the rotation matrices
123	Axis X	Axis Y	Axis Z	$\mathbf{M_Z}(\gamma)\ \mathbf{M_Y}(\beta)\ \mathbf{M_X}(\alpha)\ \mathbf{v}$
132	Axis X	Axis Z	Axis Y	$\mathbf{M_Y}(\beta)\ \mathbf{M_Z}(\gamma)\ \mathbf{M_X}(\alpha)\ \mathbf{v}$
213	Axis Y	Axis X	Axis Z	$\mathbf{M_Z}(\gamma)\ \mathbf{M_X}(\alpha)\ \mathbf{M_Y}(\beta)\ \mathbf{v}$
231	Axis Y	Axis Z	Axis X	$\mathbf{M_X}(\alpha)\ \mathbf{M_Z}(\gamma)\ \mathbf{M_Y}(\beta)\ \mathbf{v}$
312	Axis Z	Axis X	Axis Y	$\mathbf{M_Y}(\beta)\ \mathbf{M_X}(\alpha)\ \mathbf{M_Z}(\gamma)\ \mathbf{v}$
321	Axis Z	Axis Y	Axis X	$\mathbf{M_X}(\alpha)\ \mathbf{M_Y}(\beta)\ \mathbf{M_Z}(\gamma)\ \mathbf{v}$

where

$$M_X = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix} \quad M_Y = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \quad M_Z = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

If any other than the 6 above is specified at the address Q, the controll will send the message ‘2039 Q definition error’.

The rotations take place always around the X, Y, Z axes of the original coordinate system.

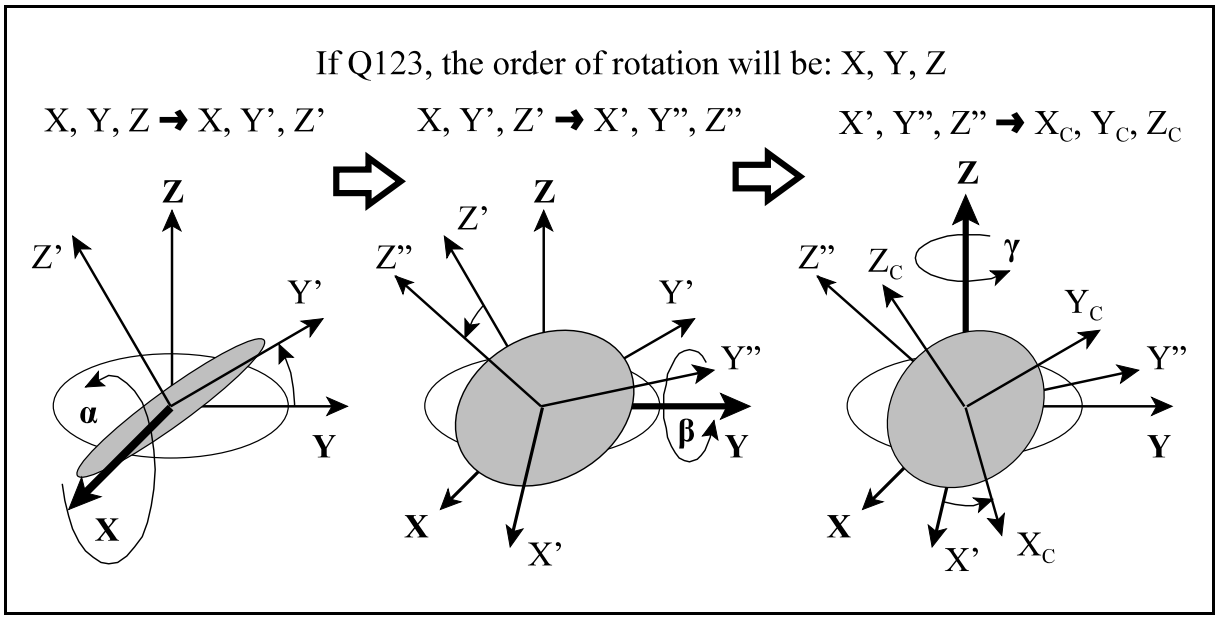


Fig. 25.7-2

The instruction ***G53.1 or*** the instruction ***G53.6*** automatically bring the rotary axes available on the machine to such position so that the plane is perpendicular to the tool axis.

The instruction

G69.1

switches the designation of the plane in space off.

An example for the application of rotation around axes:

Based on the figure, determine the coordinate system X ', Y', Z' by rotation around axes:

The origin of the coordinate system X ', Y', Z'given in the coordinate system X, Y, Z:

(200; 0; 50)

Let the order of rotation:

X; Y; Z

The angle of the first rotation around the axis X:

30°.

The angle of the second rotation around the axisY:

0°

The angle of the third rotation around the axis Z:

90°

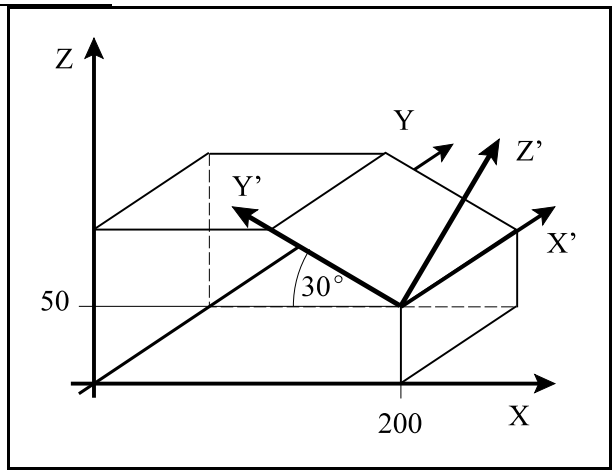


Fig. 25.7-3

The program is as follows:

```
...
G68.2 P1 Q123 X200 Y0 Z50 I30 J0 K90
G53.1 (or G53.6)
...
```

25.8 Defining a Plane in Space by its 3 Points (G68.2 P2)

A plane with arbitrary inclination in space can be defined by specification of its 3 points in the workpiece coordinate system. Then, the program can be written in one of the main planes, e.g. in the plane XY in the usual way, provided that the normal position of the tool is of direction Z.

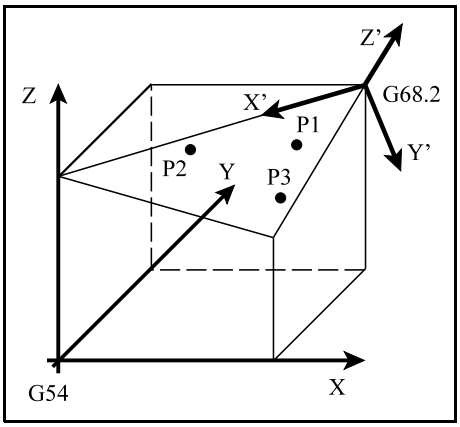


Fig. 25.8-1

The 3 points of the plane can be specified by using the following 3 blocks:

G68.2 P2 Q1 Xx$_1$ Yy$_1$ Zz$_1$
G68.2 P2 Q2 Xx$_2$ Yy$_2$ Zz$_2$
G68.2 P2 Q3 Xx$_3$ Yy$_3$ Zz$_3$

where

Q1 Xx$_1$ Yy$_1$ Zz$_1$: the point P1 of the plane and, at the same time, the origin of the coordinate system X' Y' Z';
Q2 Xx$_2$ Yy$_2$ Zz$_2$: the point P2 of the plane;
Q3 Xx$_3$ Yy$_3$ Zz$_3$: the point P3 of the plane.

The points P1, P2 and P3 must always be given with the addresses Q1, Q2 and Q3. ***The points must be given always in the workpiece coordinate system XYZ***.

It is the P1 specified at the address Q1 that will be the origin of the coordinate system X'Y'Z'.
The direction of the axis X' is determined by ***the vector pointing from P1 to P2***.
The axis Z1 is perpendicular to the plane defined by the points P1, P2 and P3, and is formed ***by rotation of the vector P1-P2 onto the P1-P3 applying the shorter angular displacement, in accordance with the right-hand rule***.
The direction of the axis Y' can be given ***by rotation of the axis Z' onto the axis X' in accordance with the right-hand rule***.

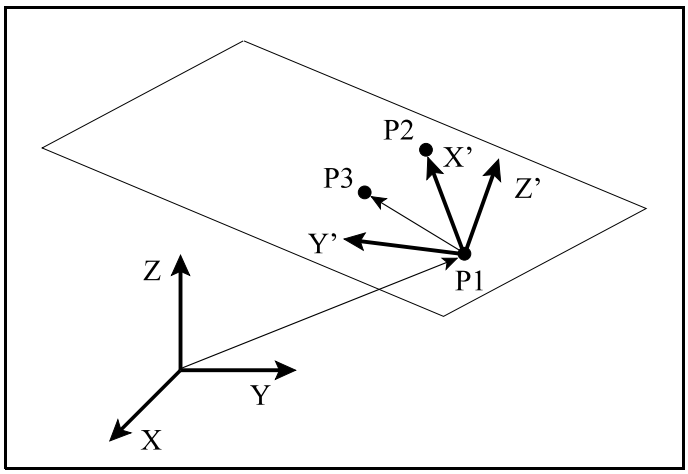


Fig. 25.8-2

The coordinate system X'Y'Z' can optionally be even translated and rotated around the axis Z" by an angle γ:

G68.2 P2 Q0 Xx$_0$ Yy$_0$ Zz$_0$ Rγ

where

Q0 Xx$_0$ Yy$_0$ Zz$_0$: ***the translation given in the coordinate system X'Y'Z'***, and
Rγ: the rotation around the axis Z", in degree.

It is not mandatory to give the point P0 and the angle γ with the address Q0.

The origin of the coordinate system X'Y'Z' is translated by the vector $\mathbf{v_0}$ (x_0, y_0, z_0) given in the same coordinate system and then is rotated around the axis Z" by the angle γ. Then, the program has to be written in the coordinate system X"Y"Z".

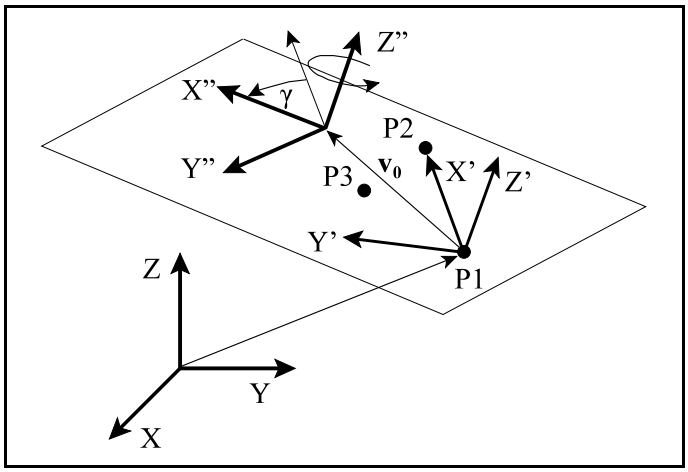


Fig. 25.8-3

If the value of Q is not 0, 1, 2 or 3, the control will send the message '2039 Q definition error'.
If any one of the points P1, P2 and P3 is not given at the addresses Q1, Q2 and Q3, the control will send the message '2064 Syntax error'.
If two or three of the points given at the addresses Q1, Q2 and Q3 are the same, or the given points lie approximately on one straight line, i.e. the plane is not defined uniquely, the control will send the error message '2088 Illegal plane selection'.

The instruction G53.1 or the instruction G53.6 brings the rotary axes available on the machine to such position so that the plane is perpendicular to the tool axis.

The instruction

G69.1

switches the designation of the plane in space off.

An example for the application of giving 3 points

Based on the figure, determine the coordinate system X ', Y', Z' by giving 3 points of the plane.
The coordinates of point 1, which is the origin of the coordinate system X'Y'Z' at the same time, given in the coordinate system XYZ:

(200; 0; 50)

The coordinates of point 2:

(200; 100; 50)

The coordinates of point 3:

(113.398; 0; 100)

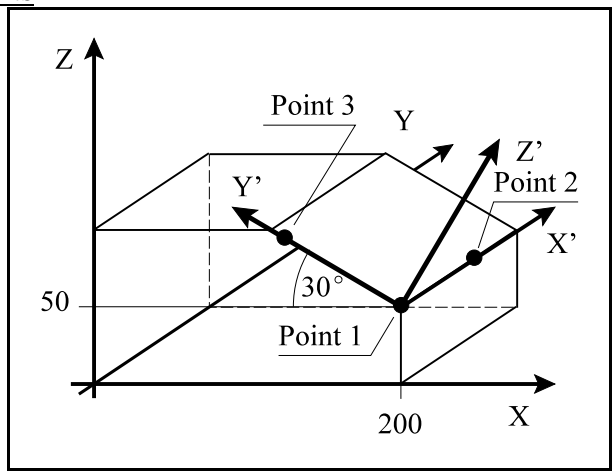


Fig. 25.8-4

The program is as follows:

```
...
G68.2 P2 Q1 X200 Y0 Z50
G68.2 P2 Q2 X200 Y100 Z50
G68.2 P2 Q3 X113.398 Y0 Z100
G53.1 (or G53.6)
...
```

25.9 Defining a Plane in Space by Giving 2 Vectors (G68.2 P3)

A plane with arbitrary inclination in space can be defined by giving 2 vectors. In addition, by using this instruction, the workpiece coordinate system can also be translated.

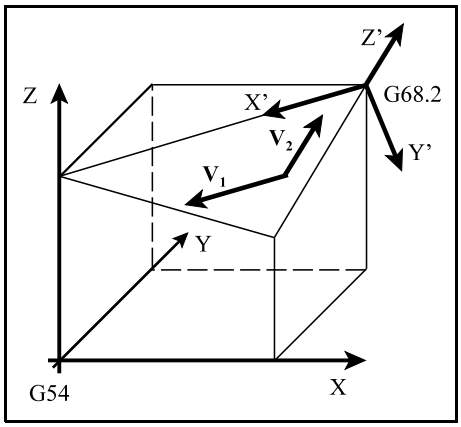


Fig. 25.9-1

The 2 vectors defining the plane and the zero point offset are specified by the following 2 blocks:

G68.2 P3 Q1 Xx Yy Zz Ii_1 Jj_1 Kk_1
G68.2 P3 Q2 Ii_2 Jj_2 Kk_2

where

Q1 Xx Yy Zz: the zero point offset, the origin of the coordinate system X'Y'Z' ***given in the workpiece coordinate systemXYZ***; filling it is optional;

Q1 Ii_1 Jj_1 Kk_1: the vector $\mathbf{V_1}$, the direction X' of the plane to be designated;

Q2 Ii_2 Jj_2 Kk_2: the vector $\mathbf{V_2}$, the direction Z' of the plane to be designated or the vector lying in the plane Z'X'**.**

Both vectors must be given in the workpiece coordinate system XYZ.

The vector **v**(x, y, z) translates the origin of the coordinate system XYZ.
The direction of the axis X' is defined by the vector $\mathbf{V_1}(i_1, j_1, k_1)$.
The vector $\mathbf{V_2}(i_2, j_2, k_2)$ either points in the direction of the axis Z', or it is the vectro lying in the plane Z'X'.
The axis Y' is perpendicular to the plane defined by the vectors V_1 and V_2 and its direction is formed ***by rotation of the vector V_2 onto the V_1 applying the shorter angular displacement, in accordance with the right-hand rule***.

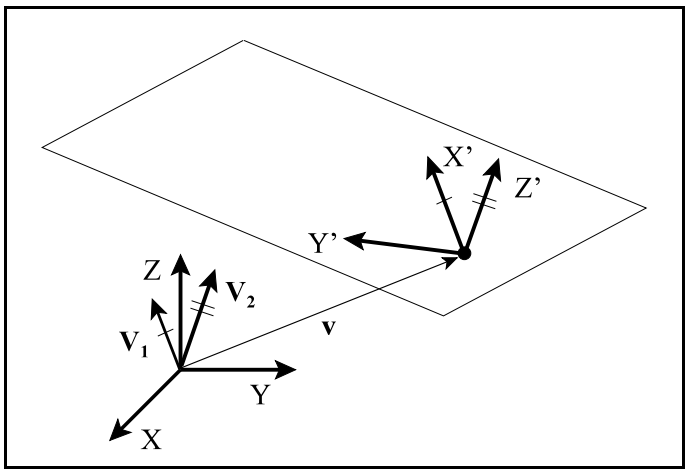


Fig. 25.9-2

The direction of the axis Z' can be given ***by rotation of the axis X' onto the axis Y', in accordance with the right-hand rule***.

The first and second vectors do not have to be perpendicular to each other. The statement that the vector $\mathbf{V_2}$ is the direction Z' of the plane to be designated or is the vector lying in the plane Z'X', is illustrated in the Figure.

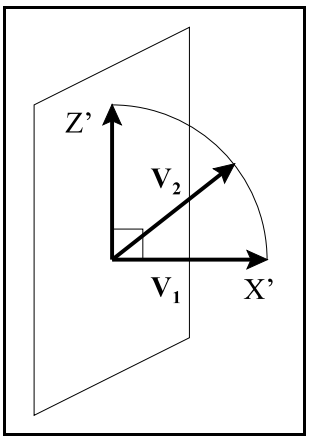


Fig. 25.9-3

If the value of Q is not 1 or 2, the control will send the message ‘2039 Q definition error’.
If any one of the vectors $\mathbf{V_1}$ and $\mathbf{V_2}$ is not given at the addresses Q1 or Q2, the control will send the message ‘2064 Syntax error’.
If the vectors $\mathbf{V_1}$ and $\mathbf{V_2}$ given at the addresses Q1 and Q2 point approximately in the same direction, i.e. the angle between them is small, or the length of any of the vectors is approximately 0, the plane will not be defined uniquely, and the control will send the error message ‘2088 Illegal plane selection'.

The instruction G53.1 or ***the instruction G53.6*** brings the rotary axes available on the machine to such position so that the plane is perpendicular to the tool axis.

The instruction

G69.1

switches the designation of the plane in space off.

An example for the application of giving 2 vectors

Based on the figure, determine the coordinate system X ', Y', Z' by giving 2 vectors:
The origin of the coordinate system X ', Y', Z' given in the coordinate system X, Y, Z:

(200; 0; 50)

The direction of the first vector, the vector $\mathbf{V_1}$, at the same time the direction of the axis X’:

(0; 100; 0)

The direction of the second vector, the vector $\mathbf{V_2}$, at the same time the direction of the axis Z’:

(50; 0; 86.603)

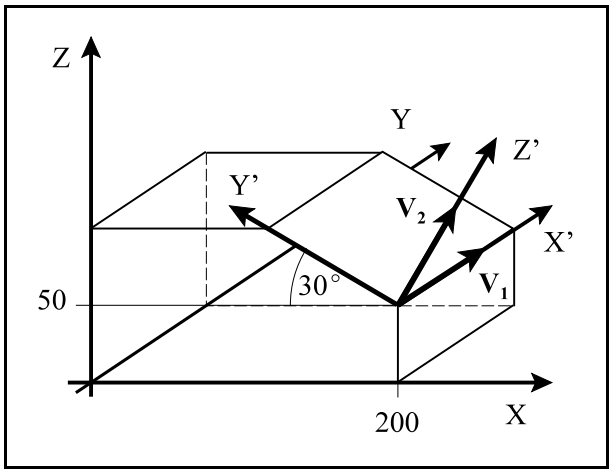


Fig. 25.9-4

The program is as follows:

```
...
G68.2 P3 Q1 X200 Y0 Z50 I0 J100 K0
G68.2 P3 Q2 I50 J0 K86.603
G53.1 (or G53.6)
...
```

25.10 Defining a Plane in Space by Giving Projection Angles (G68.2 P4)

A plane with arbitrary inclination in space can be defined by giving the projection angles of the plane.
The two vectors ($\mathbf{V_1}$ and $\mathbf{V_2}$) defining the plane are formed in the following way:
The vector $\mathbf{V_1}$ is obtained by rotation of a vector directed along the axis X around the axis Y by the angle α.
The vector $\mathbf{V_2}$ is obtained by rotation of a vector directed along the axis Y around the axis X by the angle β.

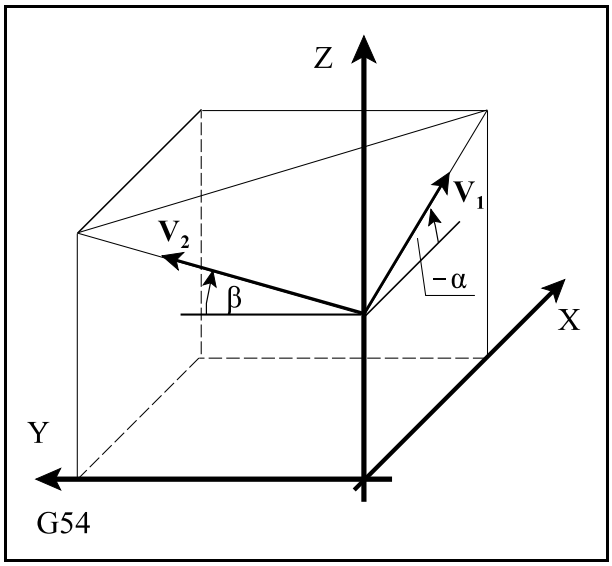


Fig. 25.10-1

The projection angles, the rotation and the zero point offset defining the plane can be specified by the following block:

G68.2 P4 Xx Yy Zz Iα Jβ Kγ

where

Xx Yy Zz: the zero point offset, the origin of the coordinate system X'Y'Z' ***given in the coordinate system XYZ***. Filling it is optional;

Iα: the angle by which the vector directed along the axis X is rotated around the axis Y;

Jβ: the angle by which the vector directed along the axis Y is rotated around the axis X;

Kγ: the angle by which the vector $\mathbf{V_1}$ should be rotated around the axis Z' to obtain the direction of the axis X'. Filling it is optional.

The vector **v**(x, y, z) translates the origin of the coordinate system XYZ.
The vector V_1 is obtained by rotation of a vector directed along the axis X around the axis Y by the angle α.
The vector V_2 is obtained by rotation of a vector directed along the axis Y around the axis X by the angle β.
The axis Z' is perpendicular to the plane defined by the vectors V_1 and V_2 and its direction is formed ***by rotation of the vector V_1 onto the V_2 applying the shorter angular displacement, in accordance with the right-hand rule.***

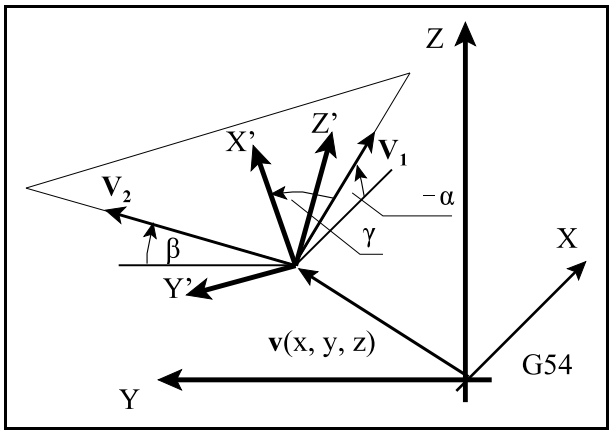


Fig. 25.10-2

The axis X' is obtained by rotation of the vector V_1 around the axis Z' by the angle γ.
The axis Y' is perpendicular to the axex Z' and X' and its direction is formed ***by rotation of***

the axis Z' onto the axis X' applying the shorter angular displacement, in accordance with the right-hand rule.

If the vectors $\mathbf{V_1}$ and $\mathbf{V_2}$ point approximately in the same direction, i.e. the angle between them is small, the plane will not be defined uniquely, and the control will send the error message '2088 Illegal plane selection'.

The instruction G53.1 or ***the instruction G53.6*** brings the rotary axes available on the machine to such position so that the plane is perpendicular to the tool axis.

The instruction

G69.1

switches the designation of the plane in space off.

An example for the application of giving projection angles

Based on the figure, determine the coordinate system X ', Y', Z' by giving projection angles:
The origin of the coordinate system X ', Y', Z' given in the coordinate system X, Y, Z:

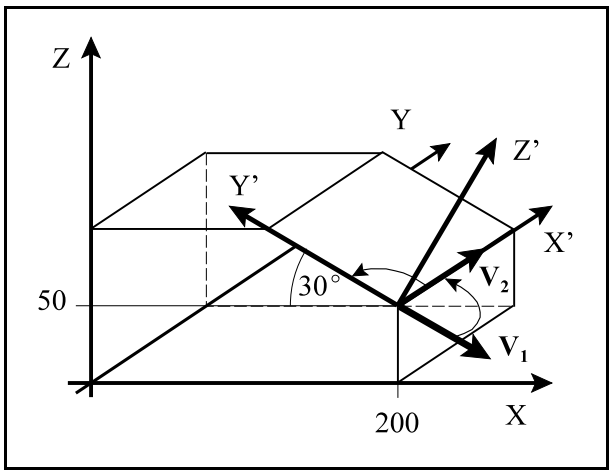
(200; 0; 50)

The first vector, the vector $\mathbf{V_1}$ is obtained by rotation of a vector directed along the axis X around the axis Y by the angle

30°.

The second vector, the vector $\mathbf{V_2}$ is obtained by rotation of a vector directed along the axis Y around the axis X by the angle

0°.



25.10**Fig.**

Then, by rotation of the vectors $\mathbf{V_1}$ and $\mathbf{V_2}$ around the axis Z'by the angle

90°,

the coordinate system X', Y', Z' will be obtained.
The program is as follows:

```
...
G68.2 P4 X200 Y0 Z50 I30 J0 K90
G53.1 (or G53.6)
...
```

25.11 Setting Rotary Axes in the Tool Direction (G53.1)

Using the instruction G68.2, a plane with arbitrary incline in space can be defined. The control is capable of calculating position where the rotary axes involved in the 5-axis machining have to be moved in order that the plane specified in G68.2 will be perpendicular to the axis of the tool.
It is the instruction

G53.1

that realizes the function above.
The instruction G53.1 has to be preceded by the instruction G68.1 defining the plane.
The instruction G53.1 has to be given ***in separate block***, ***programming other address together with it is not allowed***.
This is a ***one-shot*** and non-modal instruction.

Positioning along the axes X, Y, Z and the rotary axes assigned to 5-axis machining is executed by the use of ***linear interpolation***.

Depending on the bit #7 FOS of the parameter N3212 5D Control, the instruction G53.1 can be executed in two ways. If the value of the bit #7 FOS
=0: ***the linear axes (X, Y, Z) move simultaneously with the rotary axes***;
=1: ***only the rotary axes move***.

The following apply to the bit value #7 FOS=0 of the parameter N3212 5D Control. ***At the beginning and the end of motion G53.1, the position of the tool centre point relative to the workpiece will be the same***; however, since the 5-axis motion is executed using linear interpolation, ***the position of the tool centre point relative to the workpiece changes in the course of motion***, as it can be seen in the figure.

The axis Z+ of the coordinate system defined in the instruction G68.2 is considered to be the tool direction.
The instruction can be used for all the three configuration (head-head, table-table and head-table) of machine tool.

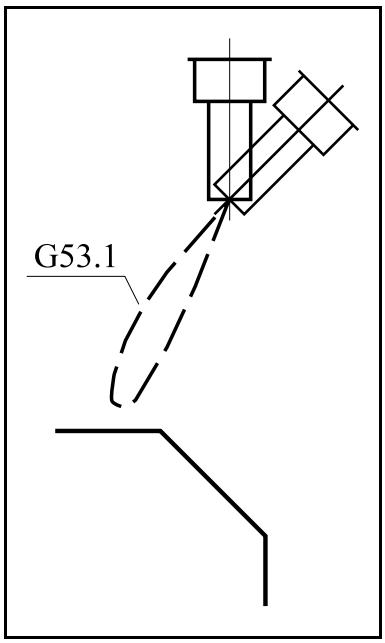


Fig. 25.11-1

The case of a machine tool of head-head configuration

Using the two rotary axes involved in the 5-axis machining, ***the tool will be set, by the instruction G53.1, in the direction Z'+ perpendicular to the plane defined by the instruction G68.2.*** Together with this, for the succeeding positioning, ***the tool tilting will also be taken into account in the tool length compensation*** changed by the instruction G43 previously. For example:

```
N10 G17 G54 G0 X0 Y0
N20 G43 Z0 H1
N30 G68.2 X100 Y100 Z50 I90 J45 K0
N40 G53.1
N50 X0 Y0 Z0    (motion is executed taking the changed tool
                length compensation into account)
```

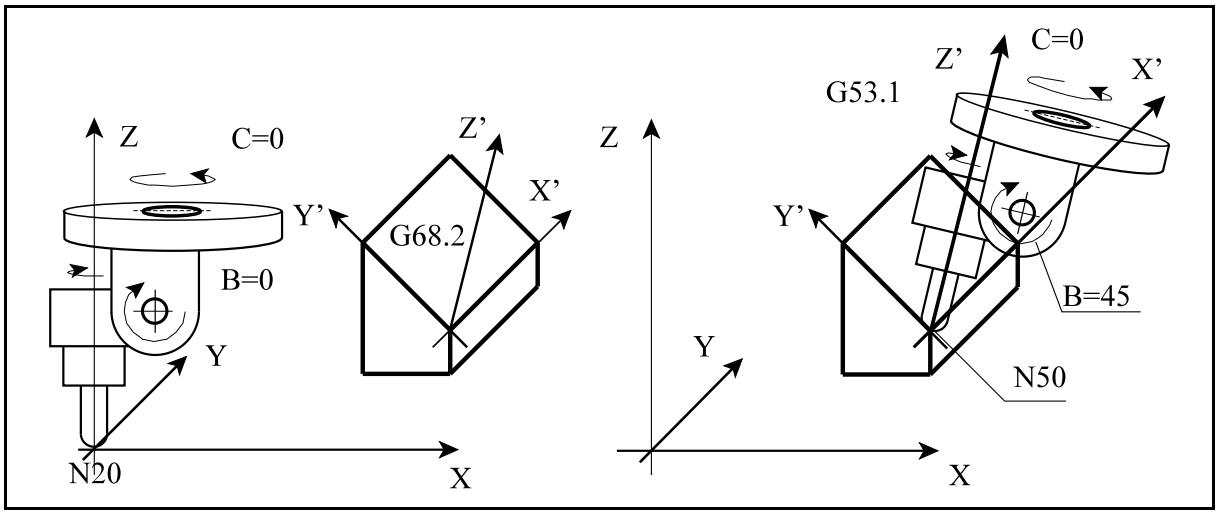


Fig. 25.11-2

The figure above illustrates such a machine tool of head-head configuration where the first rotary axis is the axis C, and the second rotary axis is the axis B. In accordance with the given Eulerian angles (I90J45K0), the rotary axes will be rotated by the control to the position B45 C0, due to the instruction G53.1.
On the left hand of the figure, the tool position at the end of ***the block N20*** is shown. In this situation, due to block G43 Z0 H1, the length compensation is managed in traditional way.
Due to the instruction ***N40 G53.1***, not only will moving the rotary axes to the required position be executed by the control, but also the inclination angle of the tool and the parameters describing geometry of the rotary axes will be taken into account in length compensation.
On the right hand of the figure, the tool position at the end of ***the block N50*** is shown, where the modified length compensation is also taken into account already.

The case of a machine tool of table-table configuration

Using the two rotary axes involved in the 5-axis machining, ***the plane defined by the instruction G68.2 will be set, by rotating the workpiece, in the direction perpendicular to the axis Z+.*** Together with this, for the succeeding positioning, ***the workpiece rotation will***

also be taken into account in the workpiece zero point offset changed by the instruction G54 previously. For example:

```
N10 G17 G54 G0 X0 Y0
N20 G43 Z0 H1
N30 G68.2 X100 Y100 Z50 I90 J45 K0
N40 G53.1
N50 X0 Y0 Z0   (motion is executed taking the changed zero
               point into account)
```

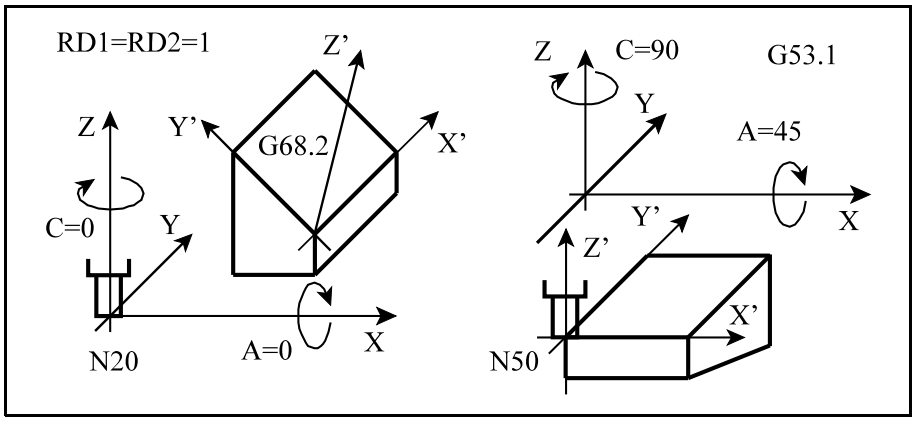


Fig. 25.11-3

The figure above illustrates such a machine tool of table-table configuration where the first rotary axis is the axis A, and the second rotary axis is the axis C. In accordance with the given Eulerian angles (I90J45K0), the rotary axes will be rotated by the control to the position A45 C90 due to the instruction G53.1, at the parameter state RD1=RD2=1.
On the left hand of the figure, the tool position at the end of ***the block N20*** is shown.
Due to ***the instruction N40 G53.1***, the workpiece will be rotated in such a way so that the direction Z'+ defined by the instruction G68.2 will be parallel with the direction Z+of the workpiece coordinate system. At the same time, the coordinate offset programmed in the instruction G68.2 and the position of the workpiece zero point will also be recalculated by the control, according to the table geometry given in parameter and the calculated angle of rotation.
On the right hand of the figure, the tool position at the end of ***the block N50*** is shown. In the course of positioning, motion is executed already in the recalculated coordinate system.

The case of a machine tool of head-table configuration

Using the two rotary axes involved in the 5-axis machining, ***the tool will be set, by rotating the workpiece and the tool, in the direction Z'+ perpendicular to the plane defined by the instruction G68.2.*** For the succeeding positioning, ***the workpiece rotation will be taken into account in the workpiece zero point offset*** changed by the instruction G54 previously and ***the tool tilting will be taken into account in the tool length compensation*** changed by the instruction G43 previously. For example:

```
N10 G17 G54 G0 X0 Y0
N20 G43 Z0 H1
N30 G68.2 X100 Y100 Z50 I0 J45 K0
N40 G53.1
N50 X0 Y0 Z0    (motion is executed taking the changed zero
                point and tool length compensation)
```

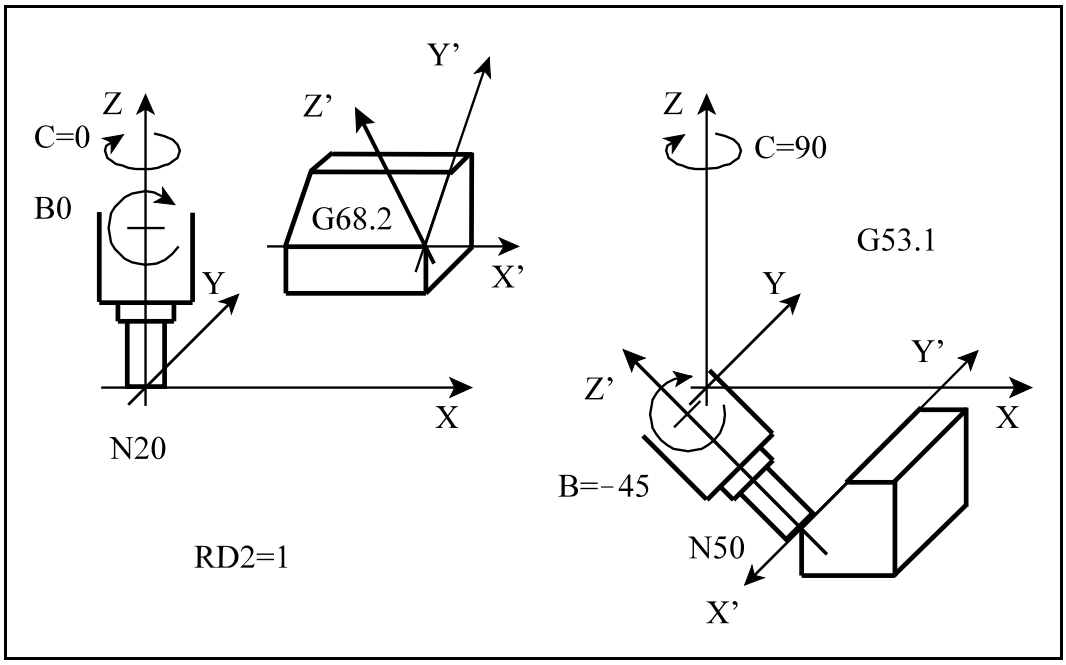


Fig. 25.11-4

The figure above illustrates such a machine tool of head-table configuration where the first rotary axis is the axis B tilting the tool around the axis Y, and the second rotary axis is the axis C rotating the workpiece around the axis Z. In accordance with the given Eulerian angles (I0J45K0), the rotary axes will be rotated by the control to the position B-45 C90 due to the instruction G53.1, at the parameter state RD2=1.
On the left hand of the figure, the tool position at the end of ***the block N20*** is shown.
Due to ***the instruction N40 G53.1***, the workpiece will be rotated around the axis C by 90° and the tool will be tilted by-45° in order that the tool position will be parallel with the axis Z'. At the same time, the position of the workpiece zero point and the offset given in the instruction G68.2 will be recalculated by the control, according to the table rotation. Modification of length compensation derived from the tool rotation will also be taken into account. All of these are calculated from the parameters describing the machine kinematics.

On the right hand of the figure, the tool position at the end of the block N50 is shown.
In the course of positioning defined ***in the block N50***, the tool is guided by the control to the specified position taking the abovementioned into account

Selection of output angles of rotation θ and ψ

The instruction G53.1 rotates in such a way so that the tool will point in the direction of the axis Z+ of the coordinate system selected by the instruction G68.2.
The configuration of the given machine tool (head-head, table-table, head-table) is taken into account in the instruction, as well as the direction into which the available rotary axes can rotate. From these data the following will be calculated by the control:

the position ***θ*** to which the ***first rotary axis*** has to be rotated, and
the position ***ψ*** to which the ***first rotary axis*** has to be rotated

in order that the tool will be perpendicular to the selected plane.
There are always two possible ways of rotation, i.e. the solution is two pairs of angles θ_1, ψ_1 és θ_2, ψ_2.
To illustrate this, let us have a machine tool of head-head configuration for example, where the direction of the tool is Z in normal situation, the first rotary axis is the axis C, the second rotary axis is the axis B. The task is to position the tool in the direction Y+.
According to the figure below, the first possible solution is rotation B90 C−90 (i.e. θ_1= −90, ψ_1=90), the second one is rotation B−90 C90 (i.e. θ_2=90, ψ_2= −90).

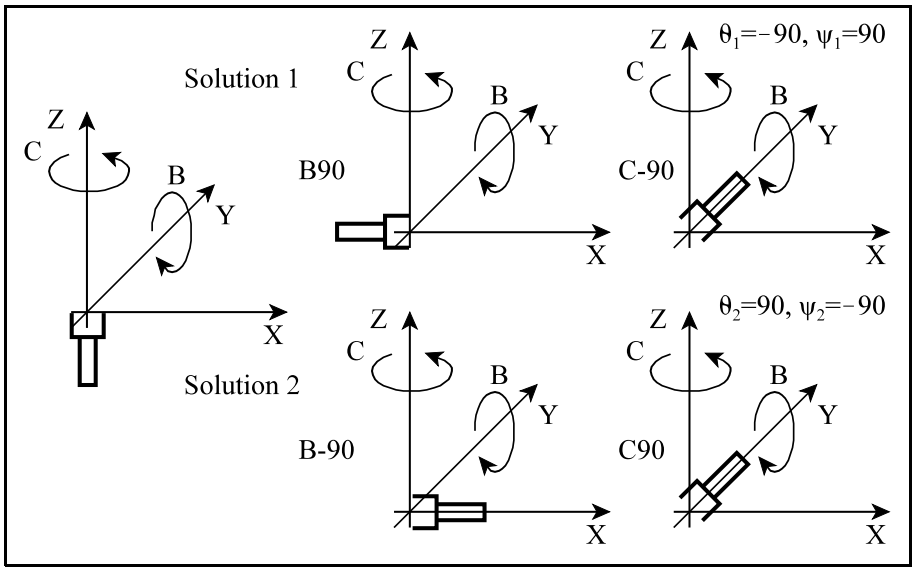


Fig. 25.11-5

Let the pairs of angles

θ_1, ψ_1 and θ_2, ψ_2

be called ***input*** or calculated angles, and the pair of angles

θ, ψ

be called ***output*** or selected angles.
From the two possible rotation, the angle of rotation will be selected by the control on the basis of the following strategy.

End point range check

It will be checked by the control whether one of the axes in any of the pairs of angles θ_1, ψ_1, and θ_2, ψ_2 falls beyond the end point range. Generally, one rotary axis, sometimes both of them can be moved in a limited range of angle. If any of them falls here, the other pair of angles will be the solution.

Determining the output angle in the case of head-head and table-table configurations

1 That one out of the two calculated pairs of angles will be the output pair of angles, in the case of which the first rotary axis has to rotate less from its actual position.
2 If, in the case of the first rotary axis, the same displacement takes place for both angles, the output pair of angles will be that one, in the case of which the second rotary axis has to rotate less from its actual position.
3 If the same displacement takes place for the second rotary axis too, the output pair of angles will be that one, in the case of which the calculated position of the first rotary axis is closer to 0 or to the integral multiple of 360.
4 If the distances of the calculated positions of the first rotary axis from 0 are the same, the output pair of angles will be that one, in the case of which the calculated position of the second rotary axis is closer to 0 or to the integral multiple of 360.

Determining the output angle in the case of head-table configuration

1 That one out of the two calculated pairs of angles will be the output pair of angles, in the case of which the second rotary axis (the table) has to rotate less from its actual position.
2 If, in the case of the second rotary axis, the same displacement takes place for both angles, the output pair of angles will be that one, in the case of which the first rotary axis (the tool) has to rotate less from its actual position.
3 If the same displacement takes place for the first rotary axis too, the output pair of angles will be that one, in the case of which the calculated position of the second rotary axis is closer to 0 or to the integral multiple of 360.
4 If the distances of the calculated positions of the second rotary axis from 0 are the same, the output pair of angles will be that one, in the case of which the calculated position of the first rotary axis is closer to 0 or to the integral multiple of 360.

The programmer can influence automatic determination of the output angles prior to issuing the instruction G53.1 by moving the rotary axes to appropriate positions.

25.12 Examples of Application of the Functions G68.2 and G53.1

Let us machine the three upper faces of a tetrahedron with edges of 100 mm. (Tetrahedron is a body all four faces of which are equilateral triangles.)
Let the workpiece zero point be at the top corner of the tetrahedron. The tool has to go round the triangle and the circle inscribed in the triangle, and drill a hole into the center of the triangle.

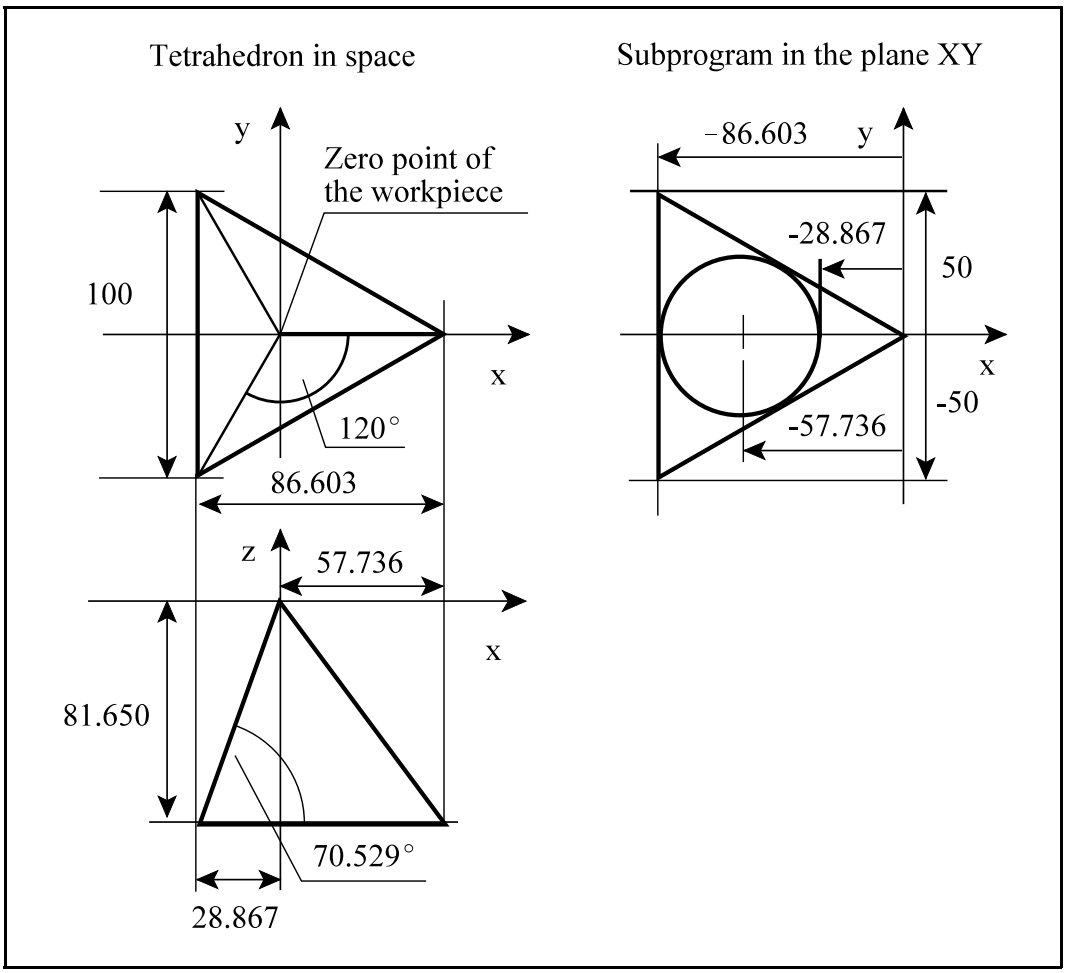


Fig. 25.12-1

Let the program machining the equilateral triangle be the subprogram O1000. The triangle spread on the plane XY is shown in the figure above. The subprogram O1000 is written according to this figure:

```
O1000 (machining one face of the tetrahedron))
N10 X20 Y0 Z-5
N20 G42 G1 X0 D1 F600 (going the triangle round)
N30 X-86.603 Y50
N40 Y-50
N50 X0 Y0
N60 G40 X20
N70 Z-2.5
```

```
N80 G42 X-28.867
N90 G3 I-28.867 (the circle inscribed in the triangle)
N100 G40 G0 X20 Z20
N110 G81 X-57.736 Y0 R2 Z-50 (drilling the hole)
N120 G80
N130 M99
```

The main program uses the plane definition instruction G68.2 together with the instruction G53. The workpiece zero point was measured in the coordinate system G54, the tool uses the length compensation cell H1.

At first, the triangle, the base edge of which is parallel with the axis Y, will be machined. By using the instruction

```
N50 G68.2 X0 Y0 Z0 I-90 J70.529 K90
```

the original coordinate system X, Y, Z will be rotated onto the plane of the triangle.
The first rotation, rotation around the axis Z (I−90) moves the axis X in the direction parallel with the base edge of the triangle. The coordinate system X', Y', Z is produced in such a way.
The second rotation, rotation around the axis X'(J70.529) rotates the axis Y' onto the face of the triangle. The coordinate system X', Y", Z" is produced in such a way.
Finally, the third rotation, rotation around the axis Z" (K90) rotates the axis X' onto the altitude line of the triangle. The plane X_E, Y_E of the coordinate system X_E, Y_E, Z_E produced in such a way already corresponds to the plane in which the subprogram O1000 was written.

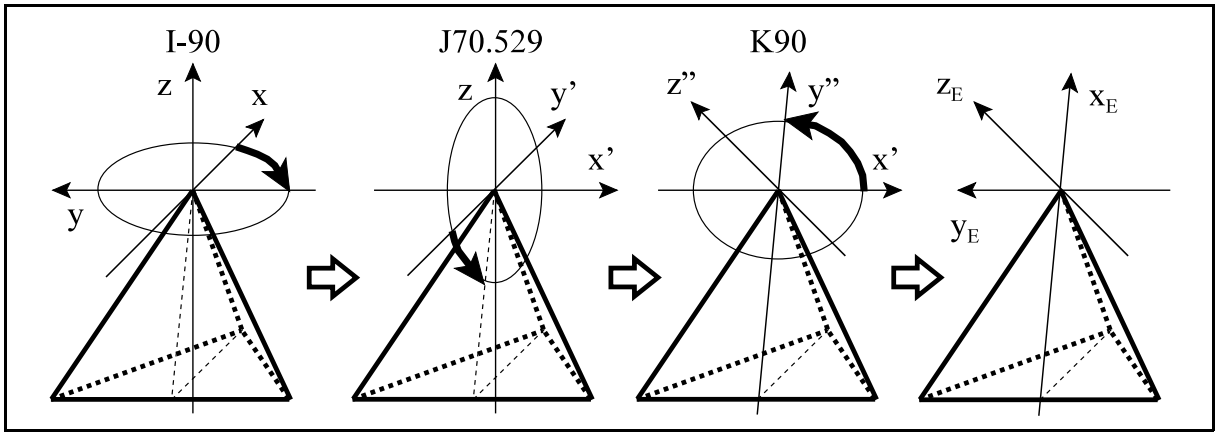


Fig. 25.12-2

In the second step, that triangle will be machined, the base edge of which locates at 30° from the axis X. By using the instruction

```
N110 G68.2 X0 Y0 Z0 I30 J70.529 K90
```

the original coordinate system X, Y, Z will be rotated onto the plane of the triangle.
The first rotation, rotation around the axis Z (I30) moves the axis X in the direction parallel with the base edge of the triangle. The coordinate system X', Y', Z is produced in such a way. The two succeeding rotations are the same as they were mentioned above.

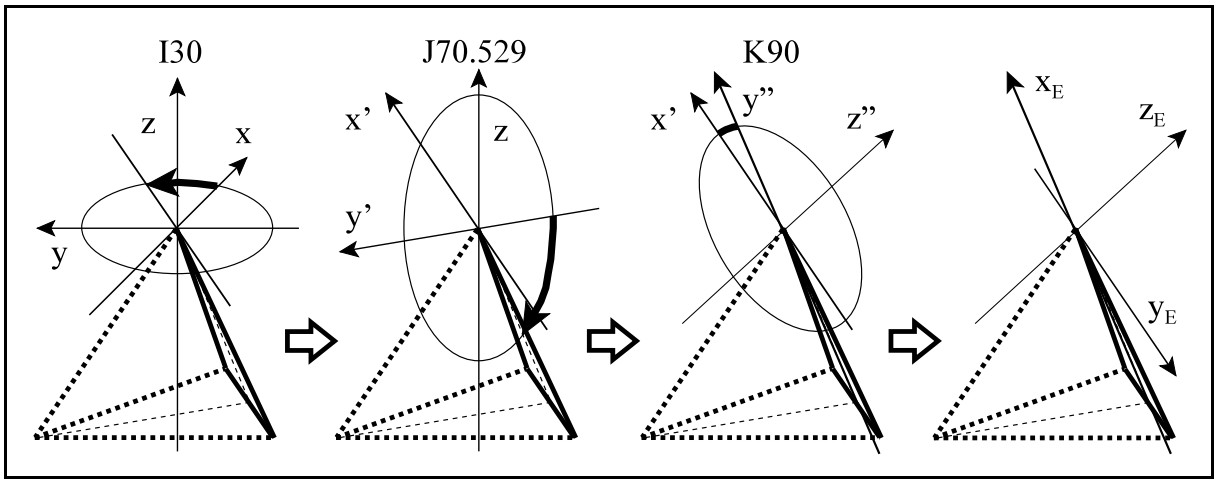


Fig. 25.12-3

Finally, that triangle will be machined, the base edge of which locates at 150° from the axis X. By using the instruction

```
N170 G68.2 X0 Y0 Z0 I150 J70.529 K90
```

the original coordinate system X, Y, Z will be rotated onto the plane of the triangle.
The first rotation, rotation around the axis Z (I150) moves the axis X in the direction parallel with the base edge of the triangle. The coordinate system X', Y', Z is produced in such a way. The two succeeding rotations are the same as they were mentioned above.

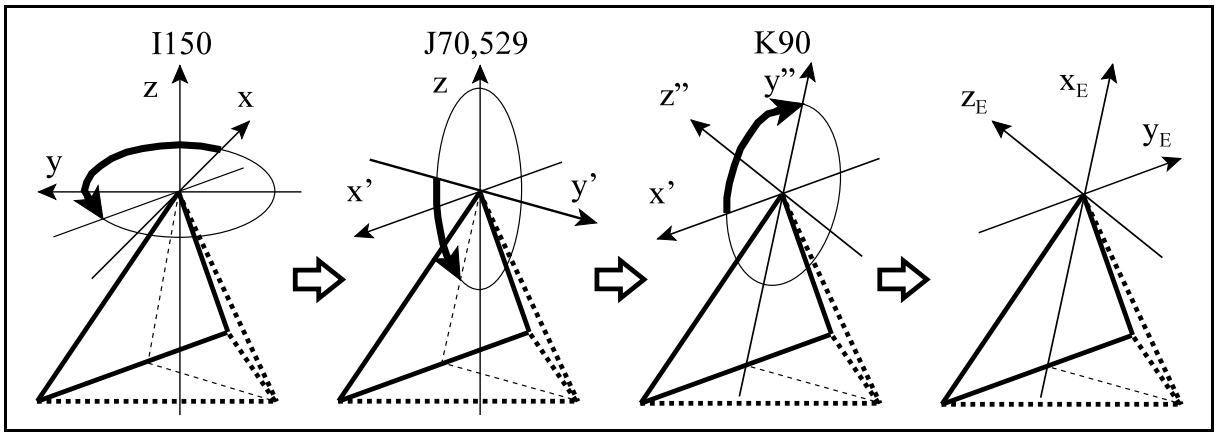


Fig. 25.12-4

The main program is as follows:

```
(Tetrahedron)
N10 G0 G54 G17 G90 G49 Z50
N20 X0 Y0
N30 M3 S1000
N40 G43 Z50 H1
N50 G68.2 X0 Y0 Z0 I-90 J70.529 K90
N60 G53.1
N70 (machining the plane 1)
N80 M98 P1000
N90 Z50
N100 X0 Y0
N110 G68.2 X0 Y0 Z0 I30 J70.529 K90
N120 G53.1
N130 (machining the plane 2)
N140 M98 P1000
N150 Z50
N160 X0 Y0
N170 G68.2 X0 Y0 Z0 I150 J70.529 K90
N180 G53.1
N190 (machining the plane 3)
N200 M98 P1000
N210 Z50
N220 X0 Y0
N230 G69.1 G49 (tool length compensation off)
N240 Z50
N250 X0 Y0
N260 M30
```

It can be seen from the main program above, that by using the pair of instructions G68.2 and G53.1 such a general program can be written, neither the configuration of the 5-axis machine tool nor the addresses of the rotary axes nor the axes around which they rotate are taken into account. Everything will be calculated by the control from the parameters related to the 5-axis machining.

By using the pair of instructions G68.2 and G53.1, programs portable between 5-axis machine tools of different configuration can be written.

25.13 Managing Other Transformations after G68.2

After the instruction G68.2, the following transformations can be applied:

G52 creating a local coordinate system
G92 creating a new workpiece coordinate system
G51 scaling
G51.1 mirroring
G68 rotation

The above transformations are always executed on the programmed coordinates, and only then the G68.2 transformation and the misalignment compensation:

– programmed coordinate – <the transformations above> – G68.2 – misalignment compensation

If more than one special transformation (G51, G51.1, G68) will also be programmed, the foregoing on page 207 in the subchapter 16.4 Programming rules for special transformations apply.

25.14 Controlling the Tool Center Point (G43.4, G43.5)

Controlling the tool center point (TCP) means that ***the tool center point is guided by the control along the programmed path*** even if the rotary axes are moved.

In the case, when the rotary axes tilt the tool, the control, during interpolation, recalculates the value of the tool length compensation according to the tilt angle of the tool from moment to moment, and, taking it into account, modifies the path of the control point continuously.

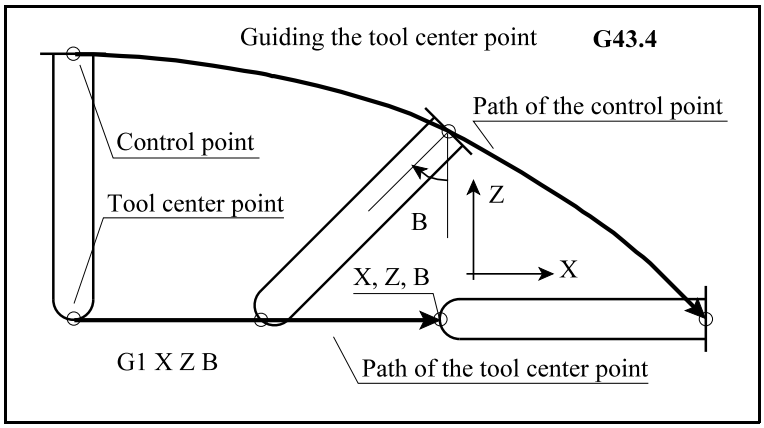


Fig. 25.14-5

In the case, when the rotary axes rotate the workpiece, the control, during interpolation, recalculates the position of the point according to the angle of rotation of the workpiece from moment to moment, and, taking it into account, modifies the path of the control point continuously.

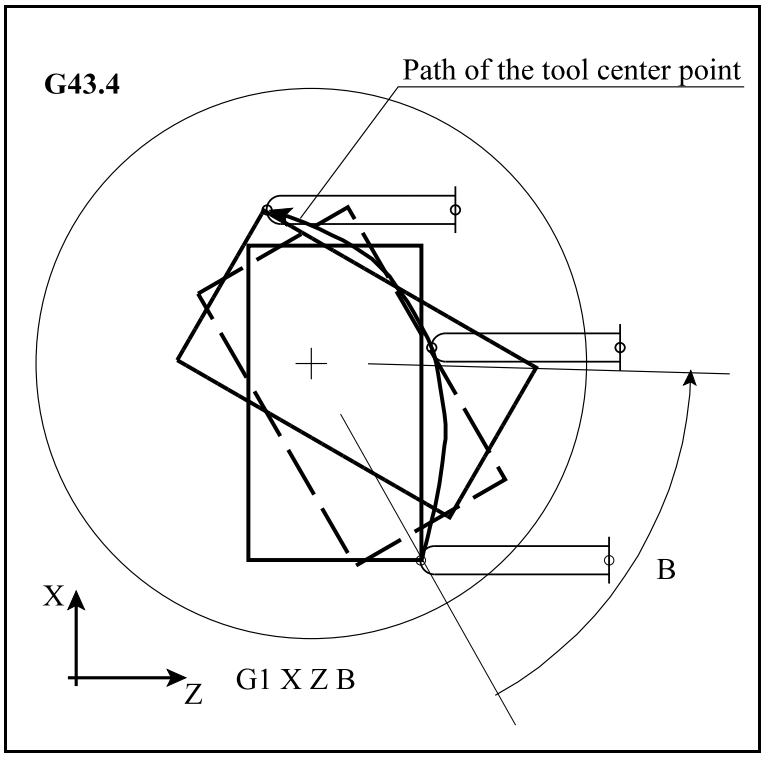


Fig. 25.14-6

In the case of guiding the tool center point, the control changes the angles of the two rotary axes in proportion to the distance covered:
If, in the given block, the distance to be covered is L and the displacements of the two rotary axes are Θ and Ψ, the control will modify the positions of the rotary axes in proportion to the distance covered along the segment L:

$$l=\lambda L$$

$\theta=\lambda\Theta$
$\psi=\lambda\Psi$

where l: the distance covered;
θ, ψ: angular displacement on the rotary axes 1 and 2;
λ=0...1 rate of the distance covered.

Type 1 controlling the tool center point (G43.4)

The instruction

G43.4 X Y Z Θ Ψ H

starts the type 1 controlling the tool center point. The instruction G43.4 can be issued in the states G0 or G1 only.

X, Y, Z: Block-end position of the linear main axes in the case of absolute data specification, and, in the case of incremental programming, displacement of them.

Θ, Ψ: Block-end position of the first and second rotary axes involved in the 5-axis machining in the case of absolute data specification, and, in the case of incremental programming, displacement of them.

H: Number of length compensation to be taken into account.

In the block G43.4, the control already positions with the application of controlling the tool center point and changes the length compensation.
In the state G43.4, the motions G0, G1, G2 and G3 can be programmed.

Positioning and linear interpolation in the state G43.4

In the blocks ***G0*** and ***G1*** succeeding the ***G43.4***:

G0 (G1)
X Y Z Θ Ψ
X Y Z Θ Ψ
...
X Y Z Θ Ψ

the control manages the control point already in such a way, so that the tool center point will move relative to the workpiece along straight path, either the tool or the workpiece rotates.

Circular interpolation in the state G43.4

In the blocks ***G2*** and ***G3*** succeeding the ***G43.4***:

$$G17\begin{Bmatrix}G2\\G3\end{Bmatrix}X\_Y\_Z\_\Theta\_\Psi\_\begin{Bmatrix}R\_\\I\_J\_\end{Bmatrix}F\_$$

$$G18\begin{Bmatrix}G2\\G3\end{Bmatrix}X\_Z\_Y\_\Theta\_\Psi\_\begin{Bmatrix}R\_\\I\_K\_\end{Bmatrix}F\_$$

$$G19\begin{Bmatrix}G2\\G3\end{Bmatrix}Y\_Z\_X\_\Theta\_\Psi\_\begin{Bmatrix}R\_\\J\_K\_\end{Bmatrix}F\_$$

the control manages the control point already in such a way, so that the tool center point will move relative to the workpiece along the defined circle path, either the tool or the workpiece tilts.

Type 2 controlling the tool center point (G43.5)

The instruction

G43.5 X Y Z I J K H

starts the type 2 controlling the tool center point. The instruction G43.5 can be issued in the states G0 or G1 only.

X, Y, Z: Block-end position of the linear main axes in the case of absolute data specification, and, in the case of incremental programming, displacement of them.

I, J, K: At the endpoint of the block, the X, Y and Z direction components of the tool direction vector relative to the workpiece.

H: Number of length compensation to be taken into account

The direction of the tool relative to the workpiece is defined by using not the angle position of the rotary axes available on the machine tool as it was done in the case of G43.4, but the vector given at the address I, J, K.

The vector components given at the addresses I, J, K do not have to form a unit vector. In order that the angle calculations will be more precise, it is suggested to give the components with as many digits as possible.

In the block G43.5, the control already positions with the application of controlling the tool center point and changes the length compensation.

In the state G43.5, the motions G0, G1, G2 and G3 can be programmed.

In the state G43.5, the address of the first an second rotary axes cannot be referred to; referring to it will result in the error message 'Specification error G43.5'**.**

Positioning and linear interpolation in the state G43.5

In the blocks ***G0*** and ***G1*** succeeding the ***G43.5***:

G0 (G1)
X Y Z I J K
X Y Z I J K
...
X Y Z I J K

the control manages the control point already in such a way, so that the tool center point will move relative to the workpiece along straight path, either the tool or the workpiece rotates.

Circular interpolation in the state G43.5

In the state G43.5, ***circular interpolation can only be programmed by specifying the circle radius***, because the addresses I, J and K are used for setting the tool direction.

In the blocks ***G2*** and ***G3*** succeeding the ***G43.5***:

$$G17\begin{Bmatrix}G2\\G3\end{Bmatrix}X\_Y\_Z\_R\_I\_J\_K\_F\_$$

$$G18\begin{Bmatrix}G2\\G3\end{Bmatrix}X\_Z\_Y\_R\_I\_J\_K\_F\_$$

$$G19\begin{Bmatrix}G2\\G3\end{Bmatrix}Y\_Z\_X\_R\_I\_J\_K\_F\_$$

the control manages the control point already in such a way, so that the tool center point will move relative to the workpiece along the defined circle path, either the tool or the workpiece tilts.

The output pair of angles θ, ψ calculated from the vector having the components I, J, K
Each vector direction can be produced by two different rotation. The way of selection between the two possible rotations by the control in the case of programming the G43.5, is the same as it is described at the function G53.1 in the subsection Selection of output angles of rotation θ and ψ on page 397.

The programmer coordinates to be taken into account in the course of controlling the tool center point
If one of the axes or both of them is table, i.e. the position Θ, Ψ programmed or the position θ, ψ calculated from the I, J an K rotates the workpiece, ***the programmer coordinates will be interpreted in the coordinate system fixed to the rotating workpiece***, and, having rotated the workpiece, both the origin and the axes of the programmer coordinate system will rotate dynamically, together with the workpiece.
In other words, based on the kinematic parameters of the machine tool, the change of the origin and rotation of the programmer coordinate system will be taken into account automatically by the control.

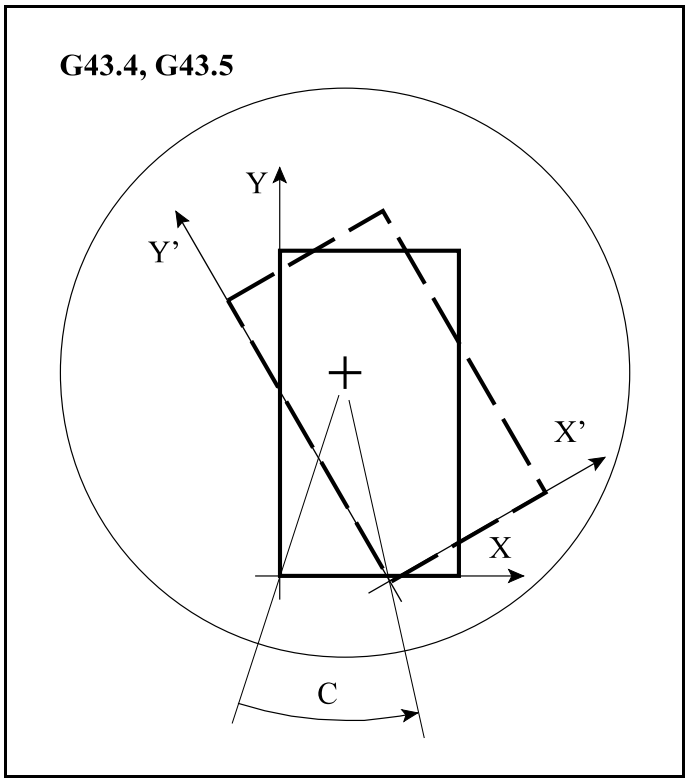


Fig. 25.14-7

Interpretation of the feed F in the case of controlling the tool center point

The programmed feed F always is the speed of the tool center point relative to the workpiece, i,e, the feed is valid along the axis X, Y and Z and presents itself along the Θ and Ψ:

$$\Delta t = \frac{\sqrt{\Delta x^2 + \Delta y^2 + \Delta z^2}}{F}$$

$$F_{\theta\psi} = \frac{\sqrt{\Delta\theta^2 + \Delta\psi^2}}{\Delta t} = \frac{\sqrt{\Delta\theta^2 + \Delta\psi^2}}{\sqrt{\Delta x^2 + \Delta y^2 + \Delta z^2}} F$$

Cancel of the controlling the tool center point (G49)

The instruction

G49

cancels the controlling the tool center point.

Only positioning (G0) or linear interpolation (G1) can be programmed together with the code G49:

G49 G0 (G1) X Y Z Θ Ψ

Example of controlling the tool center point

Let, in normal situation, the tool direction on the machine tool be Z.

By the use of controlling the tool center point, let us go along the upper edges of a square prism increasing the inclination angle of the tool relative to the axis Z from 30° to 45°, then decreasing it from 45°to 30° along the succeeding face of it, and so on. Then, let us go around the edge of the cylinder being on the square prism in such a way so that tool will have to be tilted by 30° relative to the axis Z, along the full circle.

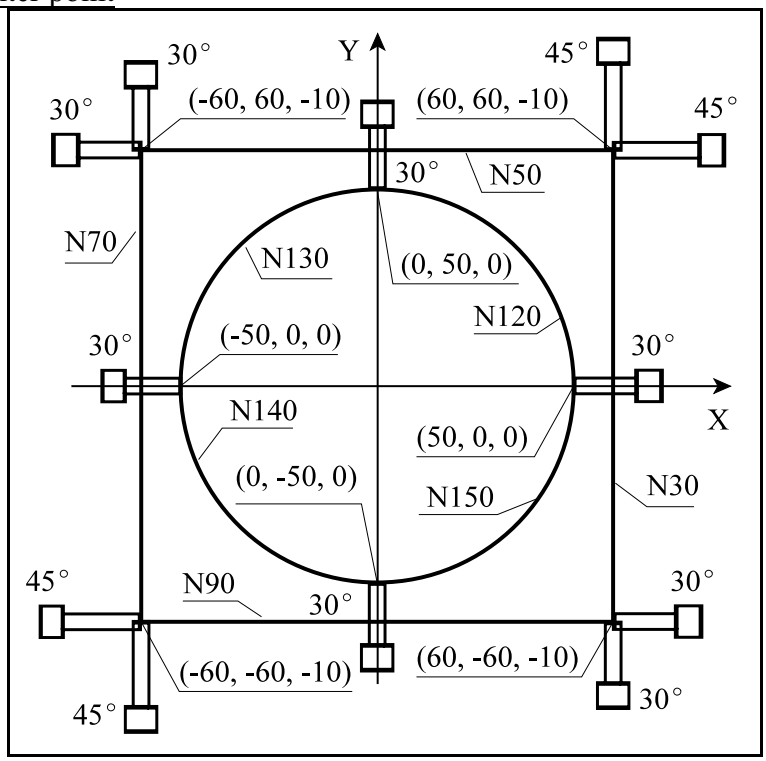


Fig. 25.14-8

Example of application of the G43.5

At first, the program has to be written using type 2 controlling the tool center point, G43.5. In the case of G43.5, the tool direction relative to the workpiece has to be specified using the vector components I, J and K.

The program is as follows:

```
N10 G54 G0 X0 Y0 Z100 S500 M3
N20 G43.5 X60 Y-60 Z-10 I500 J0 K866.025 H1
N30 G1 Y60 I707.107 J0 K707.107 F1200
N40 I0 J707.107 K707.107
N50 X-60 I0 J500 K866.025
N60 I-500 J0 K866.025
N70 Y-60 I-707.107 J0 K707.107
N80 I0 J-707.107 K707.107
N90 X60 I0 J-500 K866.025
N100 G49 G0 Z100
N110 G43.5 X50 Y0 Z0 I500 J0 K866.025 H1
N120 G3 X0 Y50 R50 I0 J500 K866.025
N130 X-50 Y0 R50 I-500 J0 K866.025
N140 X0 Y-50 R50 I-0 J-500 K866.025
N150 X50 Y0 R50 I500 J0 K866.025
N160 G49 G0 Z100
N170 X0 Y0
N180 M30
```

The figure below illustrates the value and interpretation of the directions I, J and Kin the different blocks.

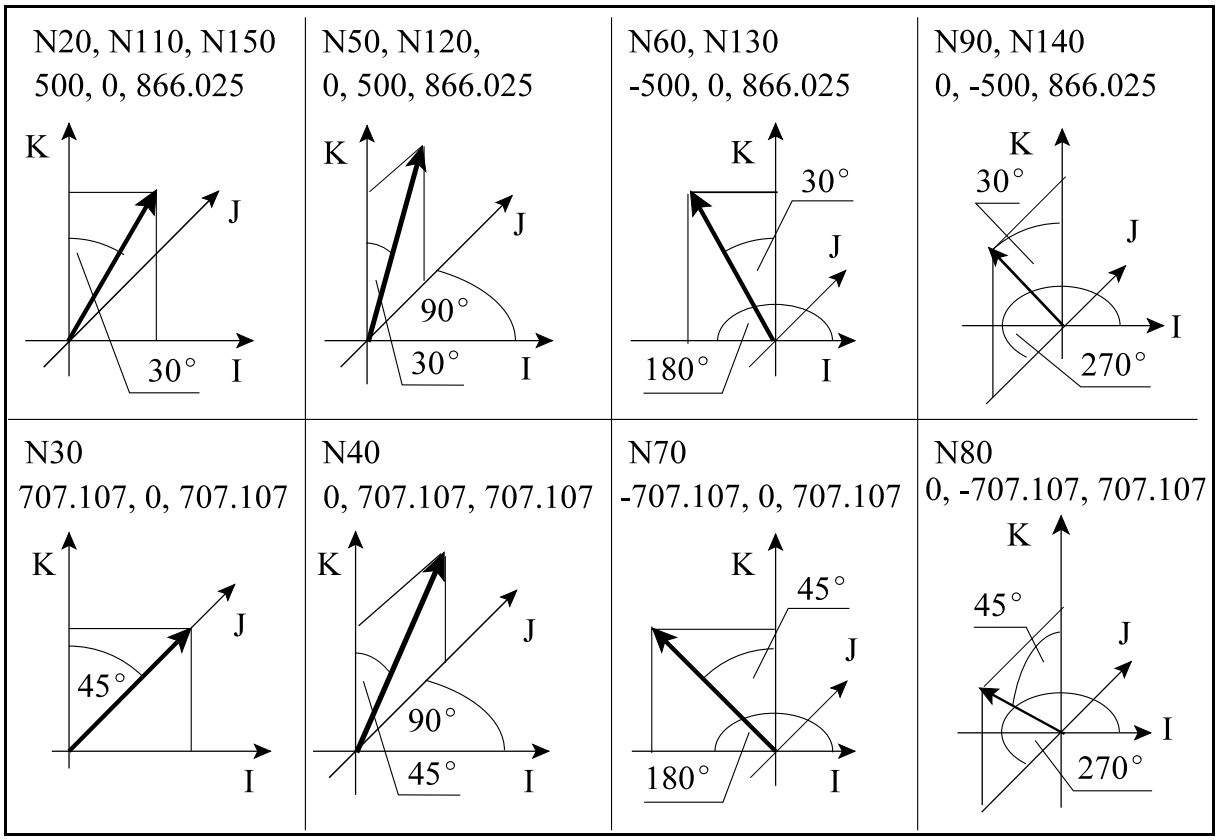


Fig. 25.14-9

The program above can be run on machine tools of arbitrary configuration (head-head, table-table or head-table) and when there are arbitrary rotary axes, provided that, in normal situation, the tool direction is Z.
On machine tools of different configuration, the program above is executed in different ways. It will be described in the following sections, how does the program above work and which are the programs that work equally and written using G43.4.

Example of application of G43.4 on the machine tool of head-head configuration
The program has to be written for a machine tool of head-head configuration, specifying G43.4.
On the machine tool of head-head configuration, let the first rotary axis be the axis C rotating around the axis Z and the second one be the axis B rotating around the axis Y. In normal situation, the tool direction is the axis Z.
In the case of G43.4, the tool direction has to always be given for the rotary axes available on the machine tool.
The program is as follows:

```
N10 G54 G0 X0 Y0 Z100 B0 C0 S500 M3
N20 G43.4 X60 Y-60 Z-10 B30 H1
N30 G1 Y60 B45 F1200
N40 C90
N50 X-60 B30
N60 C180
N70 Y-60 B45
N80 C270
```

```
N90 X60 B30
N100 G49 G0 Z100
N110 G43.4 X50 Y0 Z0 C0 H1
N120 G3 X0 Y50 R50 C90
N130 X-50 Y0 R50 C180
N140 X0 Y-50 R50 C270
N150 X50 Y0 R50 C360
N160 G49 G0 Z100
N170 X0 Y0
N180 M30
```

In this figure, motion of the tool in the block N30 is illustrated. The block starts from the point with the coordinates X60 Y−60 Z−10, where the tool is tilted by 30° relative to axis Z. In the course of motion, the tool tilts gradually, its half-way position is B37.5° and the endpoint position is B45°, while the tool center point always keeps on the straight line. It means, that the tool compensation components for each axis change continuously according to the tool tilt.

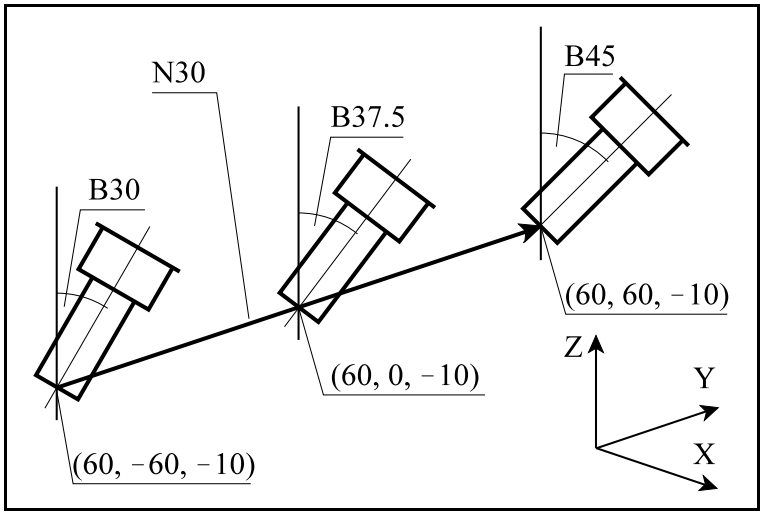


Fig. 25.14-10

The programmed feed F1200 is to be interpreted along the axis Y. Motion is similar along the other straight lines (N50, N70, N90).

In the block N40, motion of 90° is programmed so that the tool will turn onto the straight line of direction X programmed in the block N50. During rotation, the tool center point always keeps on the point with the coordinates X60 Y60 Z−10. The angle of the axis B does not change.
Motion is similar in the blocks N60 and N80, too.

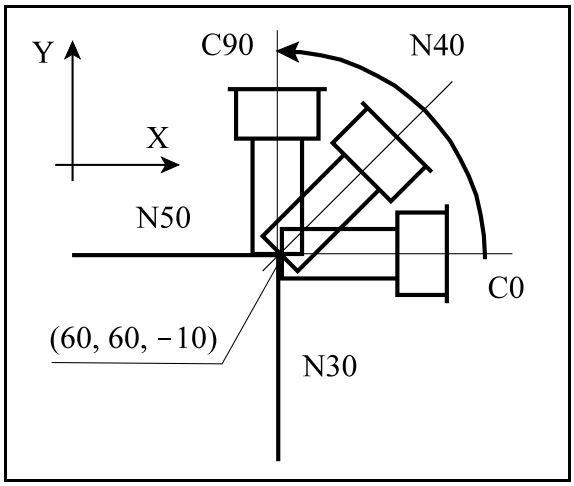


Fig. 25.14-11

In the course of circular interpolation, in the block N120, the tool rotates continuously from the block start position C0 to the block end position C90 in order that the projection of the tool onto the plane XY will always be perpendicular to the path. The axis B does not move, it keeps its position tilted by 30° relative to the axis Z.
Motion is similar in the blocks N130, N140 and N150, too.

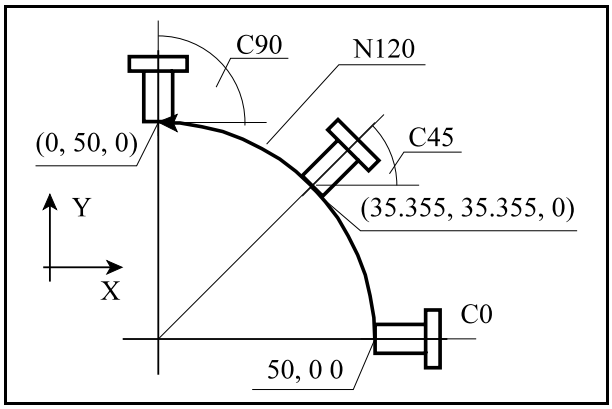


Fig. 404-12

In both cases, motion of the axes is always the same, either G43.4 (the example above) or G43.5 (the subsection Example of application of G43.5) was used for programming the path. This previous statement is true exactly in the case, when the program written with G43.5 is started from the position B0C0. Otherwise, the path could be different because of the rules of selection of the pair of angles.

Example of application of G43.4 on the machine tool of table-table configuration
The program has to be written for a machine tool of table-table configuration, specifying G43.4.
On the machine tool of table-table configuration, let the first rotary axis be the axis A rotating around the axis X and the second one be the axis C rotating around the axis Z. In normal situation, the tool direction is the axis Z.
The values A and C written in the program are specified in such a way, so that ***the positive direction of rotation of both axes will be according to the left-handed rule, so values RD1=RD2=1 have to be set in the parameter N3212 5D Control .***
The programmer coordinate system is automatically rotated together with the workpiece by the control, so the program would have to be written as if the workpiece did not rotate. The workpiece zero point has to be measured when the rotary axes are in the position of 0°.
In the case of G43.4, the tool direction has to always be given for the rotary axes available on the machine tool.

The program is as follows:

```
N10 G54 G0 X0 Y0 Z100 A0 C0 S500 M3
N20 G43.4 X60 Y-60 Z-10 A-30 C-90 H1
N30 G1 Y60 A-45 F1200
N40 C0
N50 X-60 A-30
N60 C90
N70 Y-60 A-45
N80 C180
N90 X60 A-30
N100 G49 G0 Z100
N110 G43.4 X50 Y0 Z0 A-30 C-90 H1
N120 G3 X0 Y50 R50 C0
N130 X-50 Y0 R50 C90
N140 X0 Y-50 R50 C180
```

```
N150 X50 Y0 R50 C270
N160 G49 G0 Z100
N170 X0 Y0
N180 M30
```

In the positioning block N20, the table has to be rotated into the position A−30, C−90 in order that the tool will incline by 30° relative to the axis Z of the coordinate system fixed to the workpiece.
It can be seen from the figure, that, because of rotation of the table, the programmer coordinate system fixed to the workpiece will also rotate. G53 was used for assigning the machine coordinate system (which is fixed to the axes of the machine tool) what, certainly, is not influenced by table rotations.

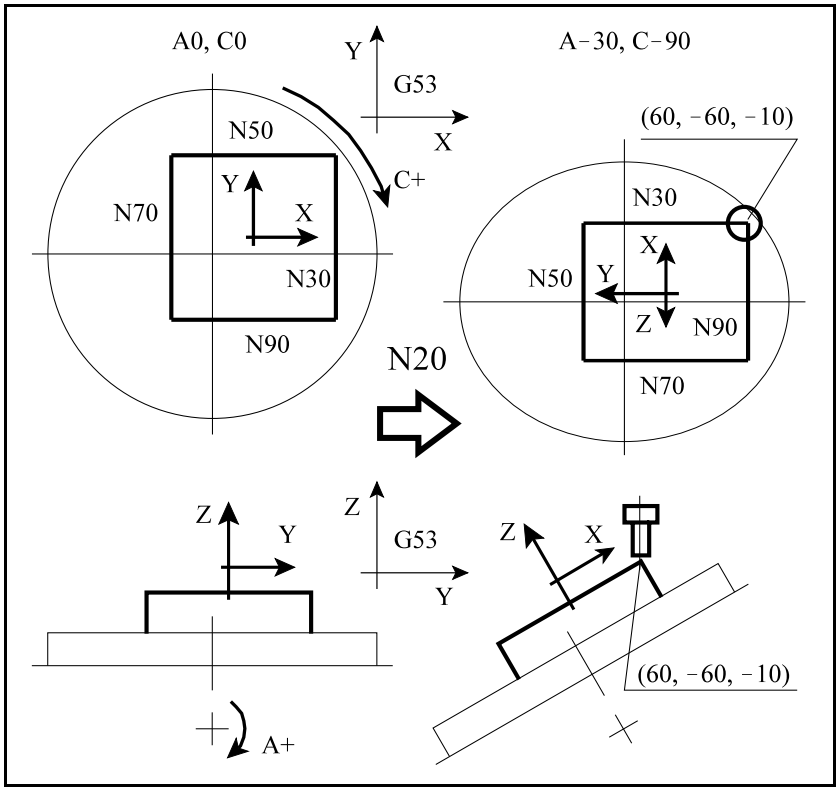


Fig. 25.14-13

In the block N30, the tool center point goes along the axis Y of the coordinate system fixed to the workpiece, while incline of the table on the axis A increases from −30°to −45°. Because of continuous tilting of the table, all the three linear axes move in the machine coordinate system.
During execution of the blocks N50, N70, N90, the situation is similar.

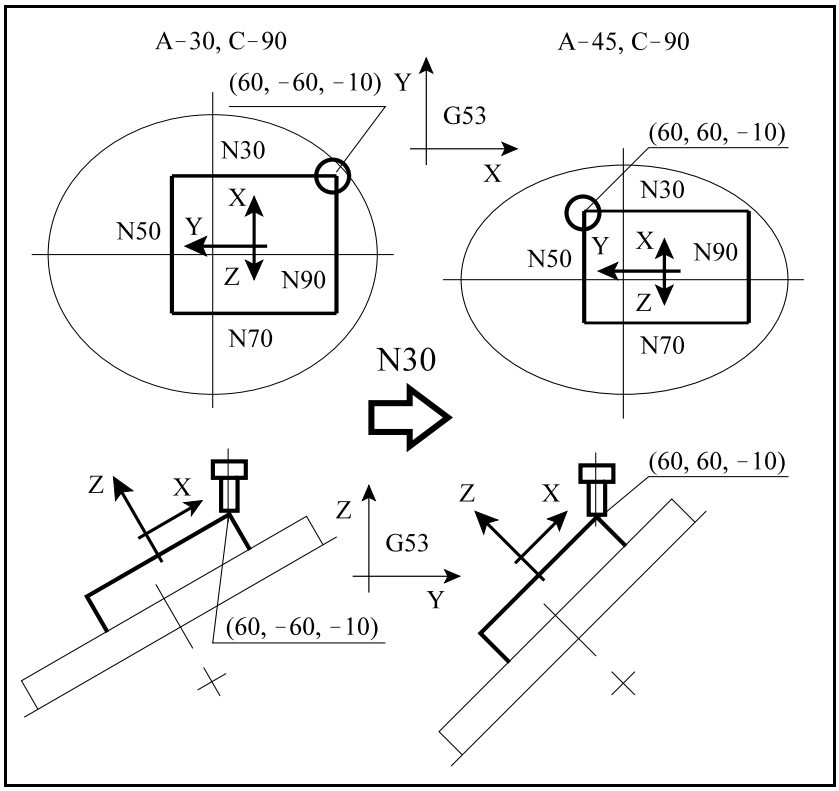


Fig. 25.14-14

In the block N40, displacement of 90°is programmed on the axis C in order that the tool will turn onto the straight line with direction X programmed in the block N50. During rotation, the tool center point keeps always on the point with the coordinates of X60 Y60 Z−10, relative to the workpiece. The angle of the axis A does not change. Because of rotation of the table, i.e. the workpiece, the positions of the tool and the workpiece relative to each other do not change, while, because of rotation of the axis C, all the three linear axis (X, Y and Z) movesin order that it will satisfy the condition above.
Motion is similar in the blocks N60 and N80, too.

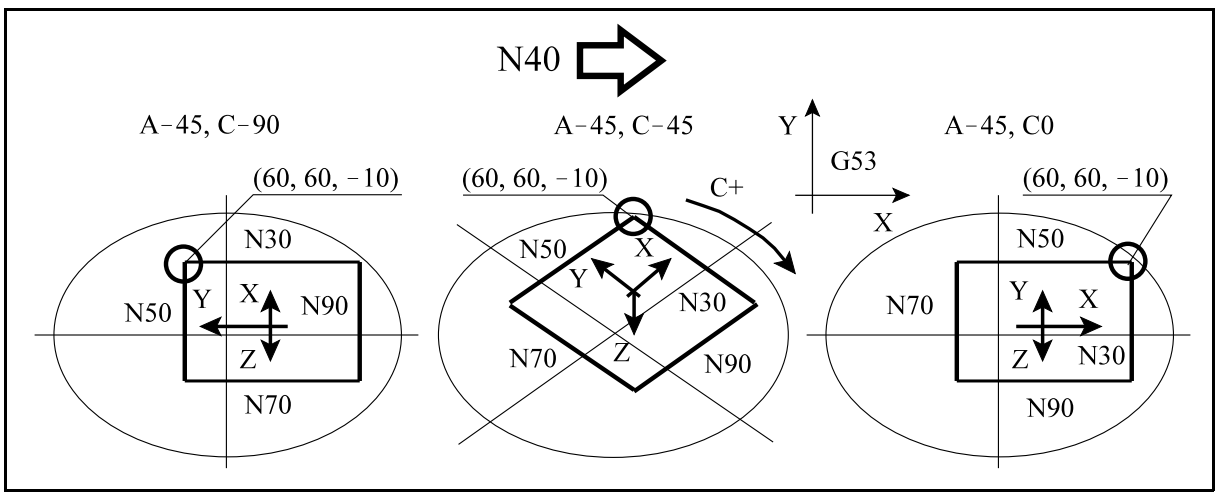


Fig. 25.14-15

In the course of circular interpolation, in the block N120, the tool rotates continuously from the block start position C−90 to the block end position C0 in order that the projection of the tool onto the plane XY will always be perpendicular to the path. The axis A does not move, it keeps its position tilted by −30° relative to the axis Z of the coordinate system fixed to the workpiece.
Motion is similar in the blocks N130, N140 and N150, too.

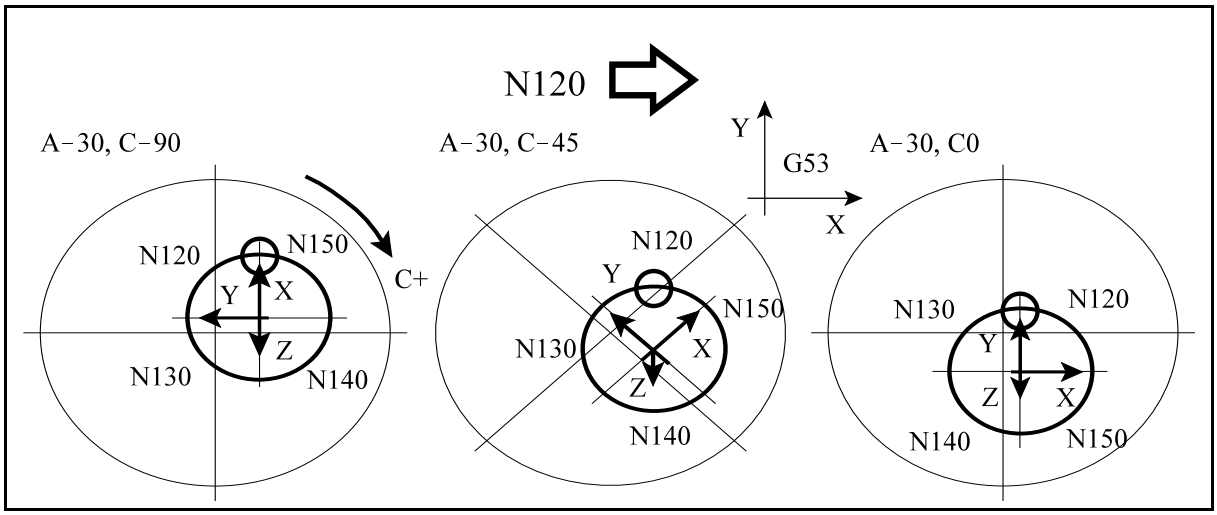


Fig. 25.14-16

In both cases, motion of the axes is always the same, either G43.4 (the example above) or G43.5 (the subsection Example of application of G43.5) was used for programming the path. This previous statement is true exactly in the case, when the program written with G43.5 is started from the position A0C0. Otherwise, the path could be different because of the rules of selection of the pair of angles.

Example of application of G43.4 on the machine tool of head-table configuration
The program has to be written for a machine tool of head-table configuration, specifying G43.4.
On the machine tool of head-table configuration, let the first rotary axis be the axis B rotating the tool around the axis Y and the second one be the axis C rotating the workpiece around the axis Z. In normal situation, the tool direction is the axis Z.
The value C written in the program is specified in such a way, so that ***the positive direction of rotation of the axis C will be according to the left-handed rule, so value RD2=1 has to be set in the parameter N3212 5D Control .***
The programmer coordinate system is automatically rotated together with the workpiece by the control, so the program would have to be written as if the workpiece did not rotate. The workpiece zero point has to be measured when the rotary axes are in the position of 0°.
In the case of G43.4, the tool direction has to always be given for the rotary axes available on the machine tool.
The program is as follows:

```
N10 G54 G0 X0 Y0 Z100 B0 C0 S500 M3
N20 G43.4 X60 Y-60 Z-10 B30 H1
N30 G1 Y60 B45 F1200
N40 C90
```

```
N50 X-60 B30
N60 C180
N70 Y-60 B45
N80 C270
N90 X60 B30
N100 G49 G0 Z100
N110 G43.4 X50 Y0 Z0 C0 H1
N120 G3 X0 Y50 R50 C90
N130 X-50 Y0 R50 C180
N140 X0 Y-50 R50 C270
N150 X50 Y0 R50 C360
N160 G49 G0 Z100
N170 X0 Y0
N180 M30
```

In this figure, motion of the tool in the block N30 is illustrated. The block starts from the point with the coordinates X60 Y−60 Z−10, where the tool is tilted by 30° relative to axis Z. In the course of motion, the tool tilts gradually, its half-way position is B37.5° and the endpoint position is B45°, while the tool center point always keeps on the straight line. It means, that the tool compensation components for each axis change continuously according to the tool tilt.

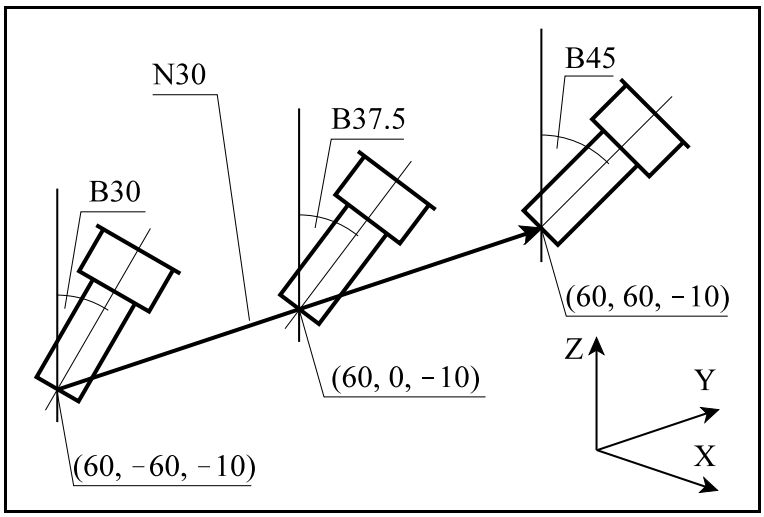


Fig. 25.14-17

The programmed feed F1200 is to be interpreted along the axis Y. Motion is similar along the other straight lines (N50, N70, N90).

In the block N40, displacement of 90°is programmed on the axis C in order that the tool will turn onto the straight line with direction X programmed in the block N50. During rotation, the tool center point keeps always on the point with the coordinates of X60 Y60 Z−10, relative to the workpiece. The angle of the axis B does not change. Because of rotation of the table, i.e. the workpiece, the positions of the tool and the workpiece relative to each other do not change, while, because of rotation of the axis C, the linear axes move in the plane XY.
Motion is similar in the blocks N60 and N80, too.

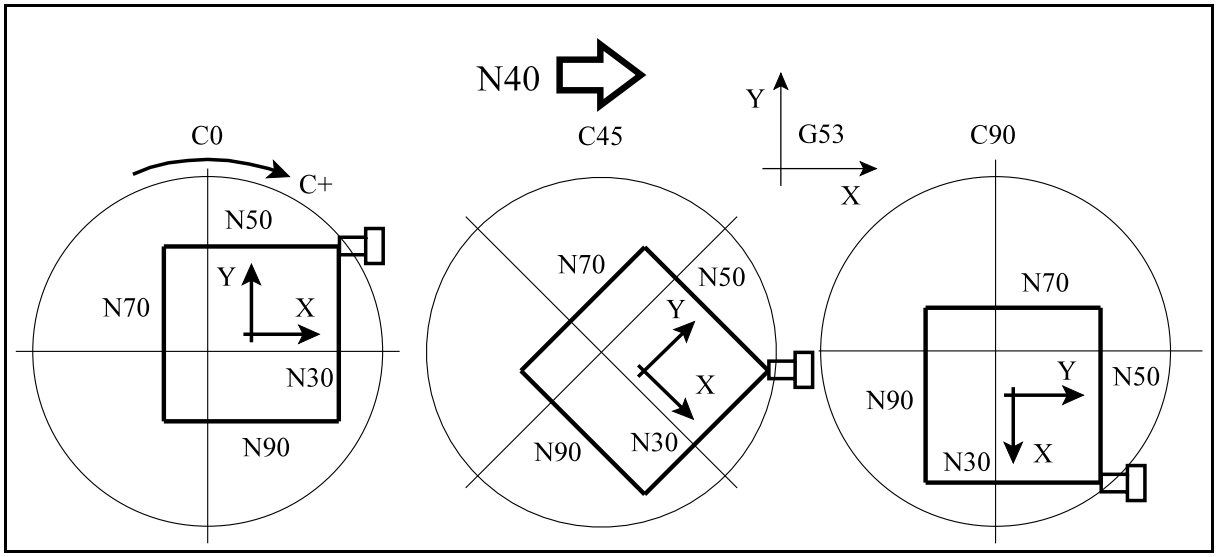


Fig. 25.14-18

In the course of circular interpolation, in the block N120, the workpiece rotates continuously from the block start position C0 to the block end position C90 in order that the projection of the tool onto the plane XY will always be perpendicular to the path. The axis B does not move, it keeps its position tilted by 30° relative to the axis Z of the coordinate system fixed to the workpiece.
Motion is similar in the blocks N130, N140 and N150, too.

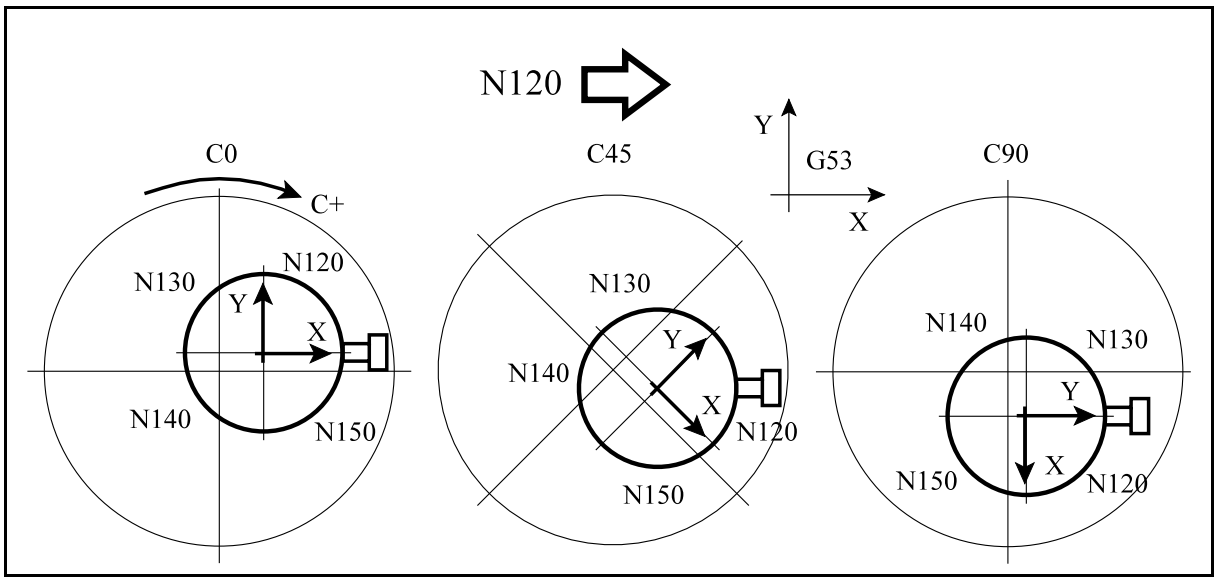


Fig. 25.14-19

In both cases, motion of the axes is always the same, either G43.4 (the example above) or G43.5 (the subsection Example of application of G43.5) was used for programming the path. This previous statement is true exactly in the case, when the program written with G43.5 is started from the position B0C0. Otherwise, the path could be different because of the rules of selection of the pair of angles.

25.15 Setting Rotary Axes in the Tool Direction by TCP (G53.6)

By the use of the command G68.2, a plane of arbitrary slope can be defined in space. The control is able to compute automatically the point where the rotary axes involved in the 5-axis machining have to be positioned to in order that plane defined in the command G68.2 is perpendicular to the axis of the tool . Meanwhile, using the command G53.6, the control is able to move the axes by TCP (Tool Center Point) control.
It is the command

G53.6 X Y Z R H

that realizes the function mentioned above, where

X, **Y**, **Z**: the endposition interpreted in the coordinate system defined in the command G68.2. It is not mandatory to provide it.

R: the distance between the tool centre point and the center point of rotation.It is not mandatory to provide it. If it is not provided, R0 will be taken into account by the control.

H: the number of the length compensation cell. It will have to be provided if there is no non-modal length compensation cell.

The instruction G53.6 has to be preceded by a command G68.2 defining the plane. Otherwise, the control will send the error message ‘2019 G53.6 is specified in G69.1 mode’.
If the address H is not provided and there is no a non-modal value different from H0, the control will send the error message ‘2003 H>0 not specified’.
The instruction G53.6 has to be provided ***in separate block***, and programming the addresses above is allowed.
The instruction is a ***one-shot*** and non-modal instruction. Motion is executed by modal interpolation code; in the case of G1, G2, G3 by the use of feed, in the case of G0 by the use of rapid traverse.
It is a waiting and buffer emptying function.
There is no sense in specifying R together with X, Y, Z in G53.6, therefore the control will send the error message ‘2042 Not specify 'R' together with positioning’.
It is the axis Z+ of the coordinate system specified in the instruction G68.2, that is taken into account by the control as direction of the tool.
The instruction can apply to all three machine tool configurations (head-head, table-table and head-table) too.

In contrast with the instruction G53.1, the instruction G53.6 is always executed by the use of TCP control, i.e. the control always keeps the tool centre point in the position relative to the workpiece even if the axes are moving; or, the control will keep the the tool center point on the programmed path if displacement X, Y, Z is also programmed.

The case when the instruction G53.6 is alone or only H is specified:

Let us have the following example:

```
...
G0 B0 C0
G68.2 X0 Y0 Z0 I90 J45
K0
G53.6 (H1)
...
```

The figure shows motion of the tool for the ***CB head-head*** machine tool.
The position of the tool center point relative to the workpiece remains unchanged even if the tool is rotating, while the control is tilting the tool in the direction perpendicular to the surface specified.
X and Z are the directions of the original workpiece coordinate system while Y' and Z' are the coordinate directions specified by the use of the instruction G68.2.
The program will be executed by the control in the coordinate system Y'Z'.

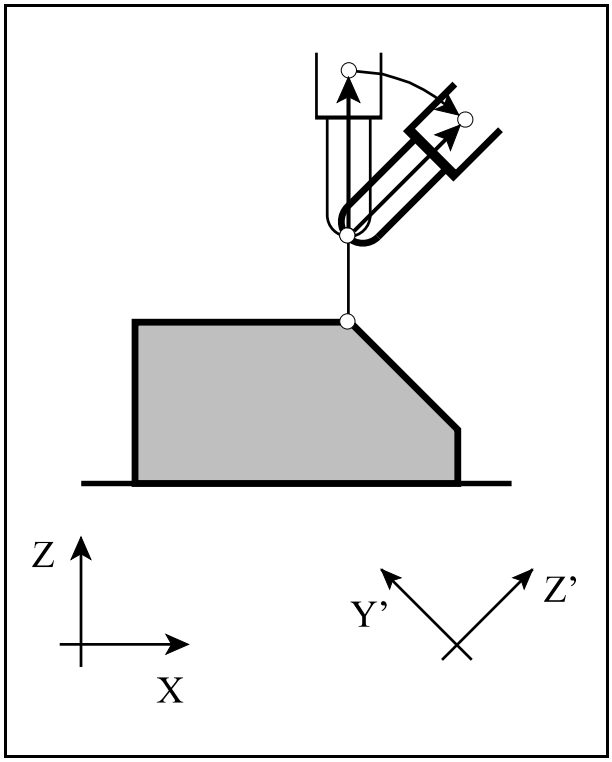


Fig. 25.15-1

The figure shows motion of the tool for the ***BC table-table*** machine tool.
The position of the tool center point relative to the workpiece remains unchanged even if the workpiece is rotating, while the control is tilting the workpiece in the direction perpendicular to the surface specified.
X and Z are the directions of the original workpiece coordinate system while Y' and Z' are the coordinate directions specified by the use of the instruction G68.2.
Following the G53.6, the program will be executed by the control in the coordinate system Y''Z'' originated after tilting the workpiece.

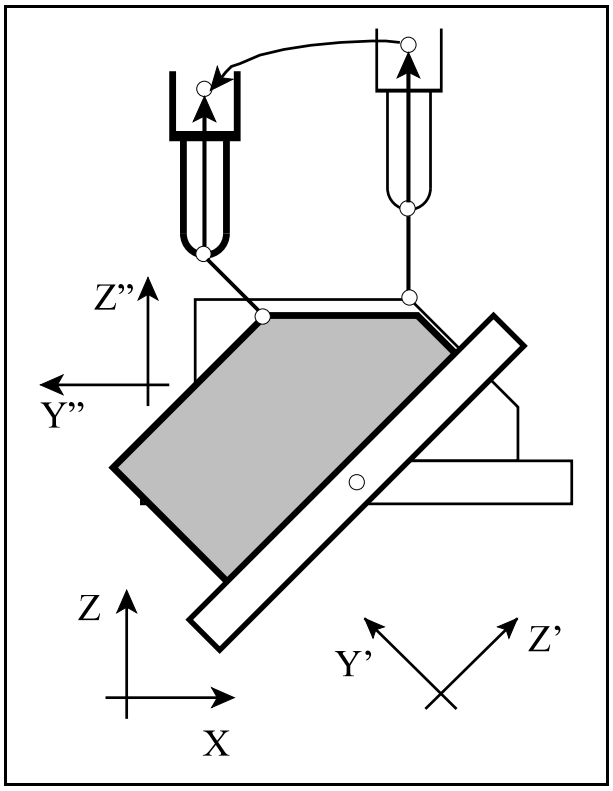


Fig. 25.15-2

The case when R (H) is specified in G53.6:

Let us have the following example:

```
...
G0 B0 C0
G68.2 X0 Y0 Z0 I90 J45
K0
G53.6 (H1) R100
...
```

The figure shows motion of the tool for the ***CB head-head*** machine tool.
In the case of specifying R, ***not the tool center point but its extension specified at the address R will be kept by the control in the position unchanged relative to the workpiece*** even if the tool is rotating, while the control is tilting the tool in the direction perpendicular to the surface specified.
X and Z are the directions of the original workpiece coordinate system while Y' and Z' are the coordinate directions specified by the use of the instruction G68.2.
The program will be executed by the control in the coordinate system Y'Z'.

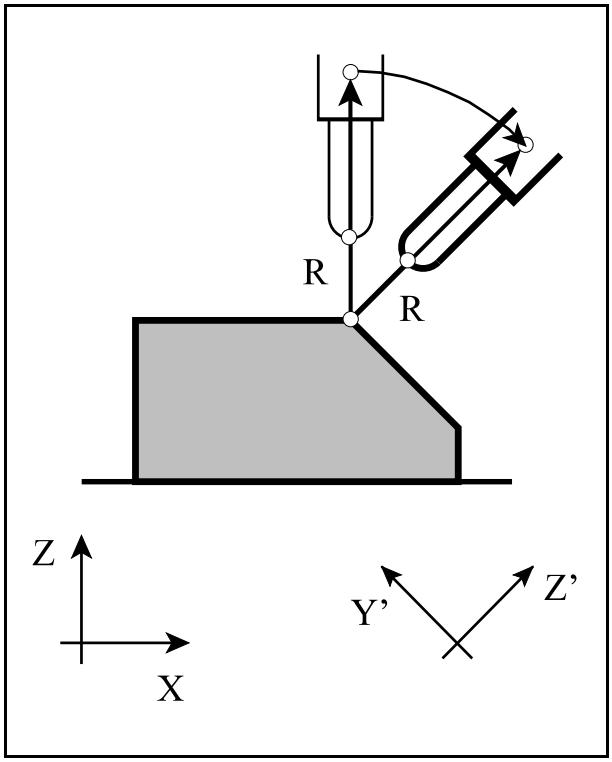


Fig. 25.15-3

The figure shows motion of the tool for the ***BC table-table*** machine tool.
In the case of specifying R, ***not the tool center point but its extension specified at the address R will be kept by the control in the position unchanged relative to the workpiece*** even if the workpiece is rotating, while the control is tilting the workpiece in the direction perpendicular to the surface specified.
X and Z are the directions of the original workpiece coordinate system while Y' and Z' are the coordinate directions specified by the use of the instruction G68.2.
Following the G53.6, the program will be executed by the control in the coordinate system Y''Z'' originated after tilting the workpiece.

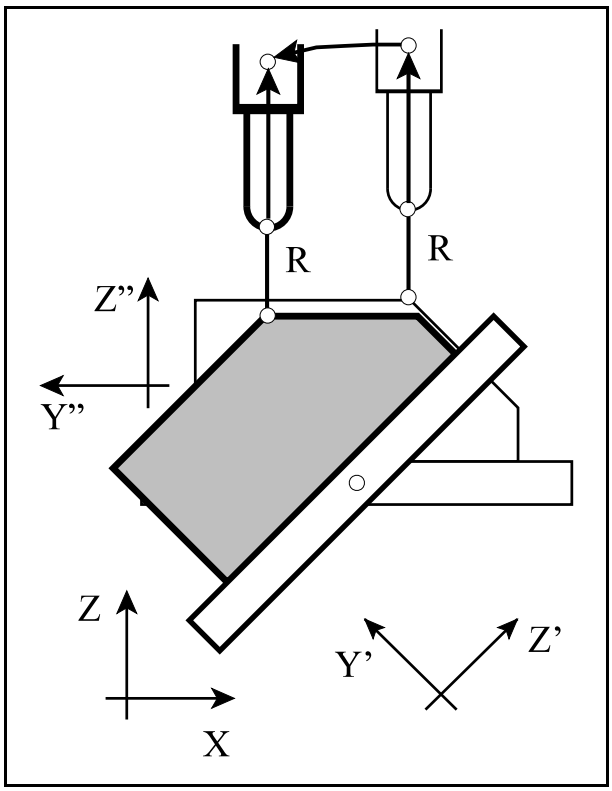


Fig. 25.15-4

The case when X Y Z (H) is specified in G53.6:

Let us have the following example:

```
...
G0 B0 C0
G68.2 X0 Y0 Z0 I90 J45 K0
G53.6 (H1) X0 Y-100 Z30
...
```

The figure shows motion of the tool for the ***CB head-head*** machine tool.
In the case of specifying X, Y, Z, ***the control moves the tool center point along a linear path to the target position interpreted in the coordinate system Y'Z' specified in the G68.2***, while tilting the tool continuously.
X and Z are the directions of the original workpiece coordinate system while Y' and Z' are the coordinate directions specified by the use of the instruction G68.2.
The program will be executed by the control in the coordinate system Y'Z'.

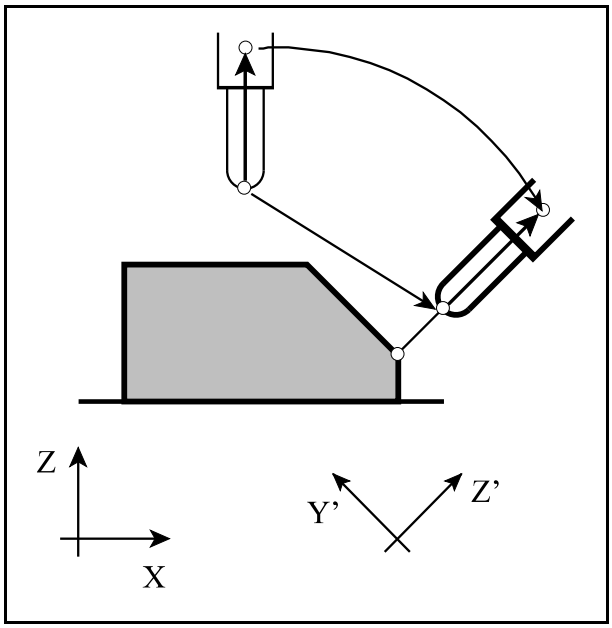


Fig. 25.15-5

The figure shows motion of the tool for the ***BC table-table*** machine tool.
In the case of specifying X, Y, Z, ***the control moves, with respect to the rotating*** **workpiece,** ***the tool center point along a linear path*** to the target position interpreted in the coordinate system Y'Z' specified in the G68.2.
X and Z are the directions of the original workpiece coordinate system while Y' and Z' are the coordinate directions specified by the use of the instruction G68.2.
Following the G53.6, the program will be executed by the control in the coordinate system Y''Z'' originated after tilting the workpiece.

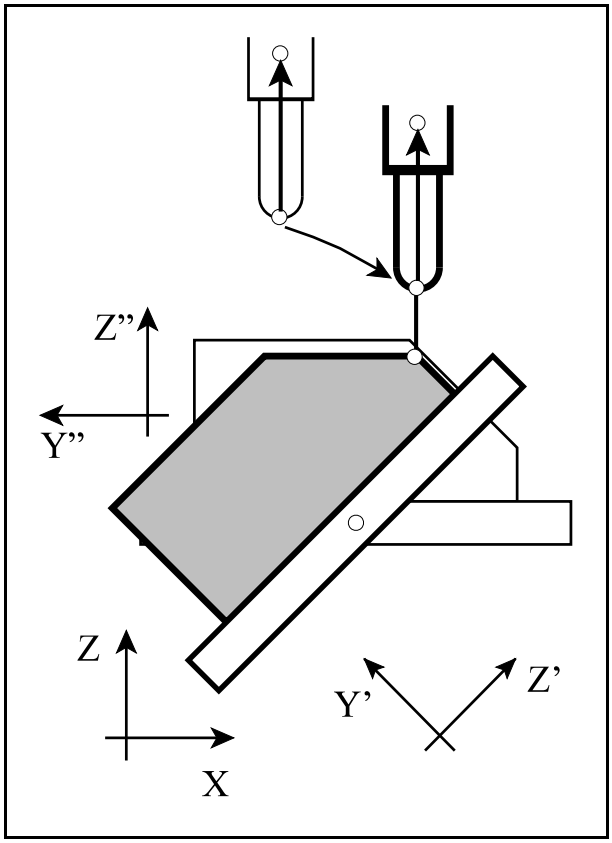


Fig. 25.15-6

25.16 Tool Posture Control (G43.4 P1, G43.5 P1)

In the case of the tool center point control, the control guides the tool center point along the programmed path in such a way so that the tool center point moves along the programmed path even if the rotary axes are moved. In this case, the movements of the two rotary axes are independent of their positions, therefore ***the tool posture is not controlled***.
The control changes the angles in proportion to the distance covered:
If, in the given block, the distance to be covered is L and the displacements of the two rotary axes are Θ and Ψ , the control will modify the positions of the rotary axes in proportion to the distance covered along the segment L:

$l = \lambda L$
$\Theta = \lambda\Theta$
$\psi = \lambda\Psi$

where l: the distance covered;
θ, ψ: angular displacement on the rotary axes 1 and 2;
λ=0...1 proportion of the distance covered.

In the case of theTool Posture Control (TPC), the control moves the tool so that, during linear interpolation, not only ***guiding the tool center point*** along the linear path, but it keeps the ***tool posture*** in the plane defined by the starting point and endpoint positions of the rotary axis. In this way, an arbitrary plane in space can be milled even by the side of the tool.

At the starting point and at the endpoint of the linear block, the direction of the tool relative to the workpiece is indicated by the vectors $\mathbf{v}_1$ and $\mathbf{v}_2$, respectively. If the ***vector*** with the direction $\mathbf{v}_1$ at the starting point of the block and the ***vector*** with the direction $\mathbf{v}_2$ at the endpoint of the block, ***together with the programmed straight line***, ***lie in the same plane***, milling ***the plane*** by the use of ***the side of the tool*** will be possible, by the application of the tool posture control.
In this case, the control continuously changes the angle between the two vectors in proportion to the distance covered along the linear path:

$l = \lambda L$
$\varphi = \lambda\Phi$

where L: the length of the straight line;
l: the distance covered;
Φ: the angle between the vectors $\mathbf{v}_1$ and $\mathbf{v}_2$;
λ = 0...1 proportion of the distance covered.

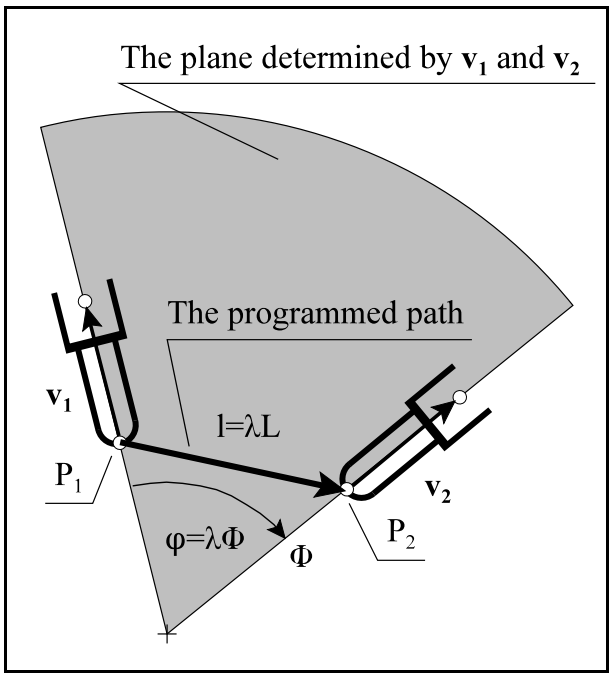


Fig. 25.16-1

During circular interpolation, at the starting point and at the endpoint of the circular block, the direction of the tool relative to the workpiece is indicated by the vectors $\mathbf{v}_1$ and $\mathbf{v}_2$, respectively. If both the ***vector*** with the direction $\mathbf{v}_1$ at the starting point of the circular block and the ***vector*** with the direction $\mathbf{v}_2$ at the endpoint of the circular block point to the apex of the cone perpendicular to the center point of the circle, machining ***the lateral surface of the cone*** by the use of ***the side of the tool*** will be possible, by the application of the tool posture control.
In this case, the control continuously changes the angle between the components of the two vectors in the plane of the circle, in proportion to the distance covered along the arc.

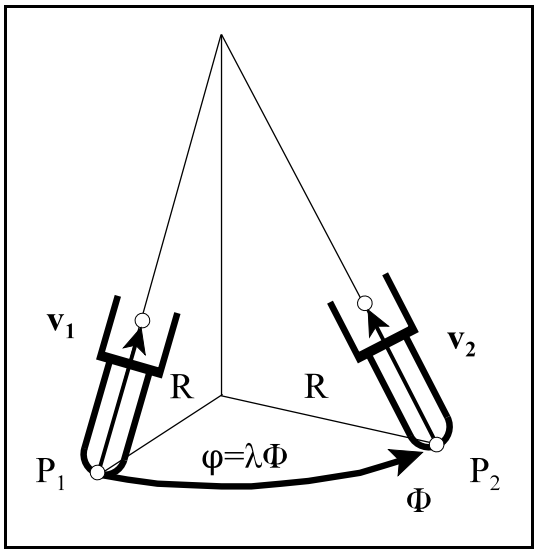


Fig. 25.16-2

Type 1 Tool Posture Control (G43.4 P1)

The instruction

G43.4 P1 X Y Z Θ Ψ H

switches the Type 1 Tool Posture Control on. The instruction G43.4 can be issued only in the state G0 or G1.

P=1: Tool Posture Control together with Tool Center Point Control
P=0: only Tool Center Point Control
X, Y, Z: block-end position of the linear main axes in the case of absolute data specification, and their movement in the case of incremental programming.
Θ, Ψ: block-end position of the rotary axes 1 and 2 participating in the 5-axis machining in the case of absolute data specification, and their movement in the case of incremental programming.
H: number of the tool length compensation to be taken into account

In the block G43.4, the control activates the length compensation.
In the state G43.4, the motions G0, G1, G2 and G3 can be programmed.

Positioning and linear interpolation in the state G43.4 P1

In the blocks ***G0*** and ***G1***:

G0 (G1)
X Y Z Θ Ψ
X Y Z Θ Ψ
...
X Y Z Θ Ψ

after the ***G43.4 P1***, the control already manages the control point using the tool posture control together with the tool center point control, either the tool or the workpiece tilts.

Circular interpolation in the state G43.4 P1

In the blocks ***G2*** and ***G3***:

$$G17\begin{Bmatrix}G2\\G3\end{Bmatrix}X\_Y\_Z\_\Theta\_\Psi\_\begin{Bmatrix}R\_\\I\_J\_\end{Bmatrix}F\_$$

$$G18\begin{Bmatrix}G2\\G3\end{Bmatrix}X\_Z\_Y\_\Theta\_\Psi\_\begin{Bmatrix}R\_\\I\_K\_\end{Bmatrix}F\_$$

$$G19\begin{Bmatrix}G2\\G3\end{Bmatrix}Y\_Z\_X\_\Theta\_\Psi\_\begin{Bmatrix}R\_\\J\_K\_\end{Bmatrix}F\_$$

after the ***G43.4 P1***, the control already manages the control point using the tool posture control together with the tool center point control, either the tool or the workpiece tilts.

Type 2 Tool Posture Control (G43.5 P1)

The instruction

G43.5 P1 X Y Z I J K H

switches the Type 2 Tool Posture Control on. The instruction G43.5 can be issued only in the state G0 or G1.

P=1: Tool Posture Control together with Tool Center Point Control
P=0: only Tool Center Point Control
X, Y, Z: block-end position of the linear main axes in the case of absolute data specification, and their movement in the case of incremental programming.
I, J, K: in the endpoint of the block, the X-, Y- and Z-direction components of the tool-direction vector, relative to the workpiece.
H: number of the length compensation to be taken into account

The direction of the tool relative to the workpiece will be determined not with angular position of the rotary axes available on the machine as it was done in the case of the G43.4, but with the vector given at the address I, J, K.

The vector components given at the address I, J, K do not have to form a unit vector. In order that the angle calculations will be more precise, it is suggested to give the components with as many digits as possible.

In the block G43.5, the control activates the length compensation.

In the state G43.5, the motions G0, G1, G2 and G3 can be programmed.

In the state G43.5, the addresses of the rotary axes 1 and 2 cannot be referred to; doing this will result in the error message 'G43.5 definition error'.

Positioning and linear interpolation in the state G43.5 P1

In the blocks ***G0*** and ***G1***:

G0 (G1)
X Y Z I J K
X Y Z I J K
...
X Y Z I J K

after the ***G43.5 P1***, the control already manages the control point using the tool posture control together with the tool center point control, either the tool or the workpiece tilts.

Circular interpolation in the state G43.5 P1

In the state G43.5, circular interpolation ***can be programmed only by giving the radius of the circle***, because the addresses I, J, K are used for setting the tool direction.

In the block ***G2*** and ***G3***

$$G17\begin{Bmatrix}G2\\G3\end{Bmatrix}X\_Y\_Z\_R\_I\_J\_K\_F\_$$

$$G18\begin{Bmatrix}G2\\G3\end{Bmatrix}X\_Z\_Y\_R\_I\_J\_K\_F\_$$

$$G19\begin{Bmatrix}G2\\G3\end{Bmatrix}Y\_Z\_X\_R\_I\_J\_K\_F\_$$

after the ***G43.5***,the control already manages the control point using the tool posture control together with the tool center point control, either the tool or the workpiece tilts.

The output pair of angles θ and ψ calculated from the vector with the components I, J, K
Every vector direction can be produced by two different rotations. In contrast to the tool center point control, the control will never select the output pair of angles in the way described in the section 'Selection of output angles of rotation θ and ψ' on page 397. ***The output pair of angles will always be selected so that the tool posture remains on the surface (plane, cone).***

The programmer coordinates to be taken into account in the case of the tool posture control
The situation is the same as it was for the tool center point control, i.e. the axes have to always be programmed in the coordinate system fixed to the workpiece even if the workpiece rotates.

Interpretation of the feed F in the case of the tool posture control
The situation is the same as it was for the tool center point control, i.e. the programmed feed F is always the speed of the tool center point relative to the workpiece.

Cancelling the tool posture control (G49)

The instruction

G49

cancels the tool posture control.
Together with the code G49, only positioning (G0) or linear interpolation (G1) can be programmed:

G49 G0 (G1) X Y Z Θ Ψ

☞ ***Attention!***

The control will continuously change the angle between the starting point vector and the endpoint vector, even if the straight lines that are laid along the tool directions given at the starting point and the endpoint of the linear block, and the programmed straight line do not fall in the same plane; but it will not be a plane what the tool posture will machine.

If the tool direction in the starting point and in the endpoint of the circular block points not to the apex of the cone (not to the given point on the straight line perpendicular to the center point of the circle), ***it will not be a cone*** what the tool posture will machine.

Data specification error in the case of the tool posture control

In the case of the Type 1 tool posture control, when the block-end tool position is defined by the angle position of the rotary axes, the control will indicate error if the tool posture control is not possible by using the given data.

Let us assume that our machine is a head-head machine on which the axis 1 (the axis C) rotates around the axis Z, and the axis 2 (the axis B) rotates around the axis Y. On this machine, we would like to mill in the plane XY using the the side of the tool. We write the program using the function G43.4 P1.

At the beginning of the block N60, let the anguler position of the tool be parallel with the axis X and the positions of the rotary axes be:

B90, C0,

as it is shown in the figure. At the end point let the tool be parallel with the axis Y. This position can be reached by two different rotations:

The case 1: B90, C90

The case 2: B-90, C-90

Because in the case 2 the value by which the axis B would rotate is 180 degree and the tool posture would not remain in the plane XY, the control will send the error message ‘2192 Invalid posture in TPC: -90'. The following program parts show examples for correct and wrong definitions:

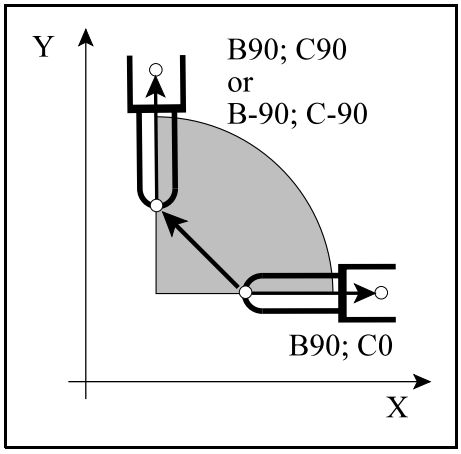


Fig. 25.16-3

Correct definition:

```
...
G43.4 P1 H1 ...
...
N50 X_ Y_ Z_ B90 C0
N60 X_ Y_ B90 C90
...
```

Wrong definition:

```
...
G43.4 P1 H1 ...
...
N50 X_ Y_ Z_ B90 C0
N60 X_ Y_ B-90 C-90
...
```

In the case of the Type 2 tool posture control, if the tool direction valid at the beginning of the block and the tool direction specified for the end of the block are very different, error will also be indicated by the control:

```
...
G43.5 P1 H1 ..
...
G1 X100 Y100 I-20 J-20 K40
Y-100 I20 J20 K-40 (ERROR!!!)
...
```

An example of milling a plane with the application of the tool posture control

In the following program, the lateral faces of a truncated pyramid are gone around with the application of the tool posture control. It is the side of the tool that executes cutting operation. Interpretation of

```
I20 J-20 K40
```

programmed in the block N40 can be seen in the figure. The movement of the tool is indicated in the block N70, too.

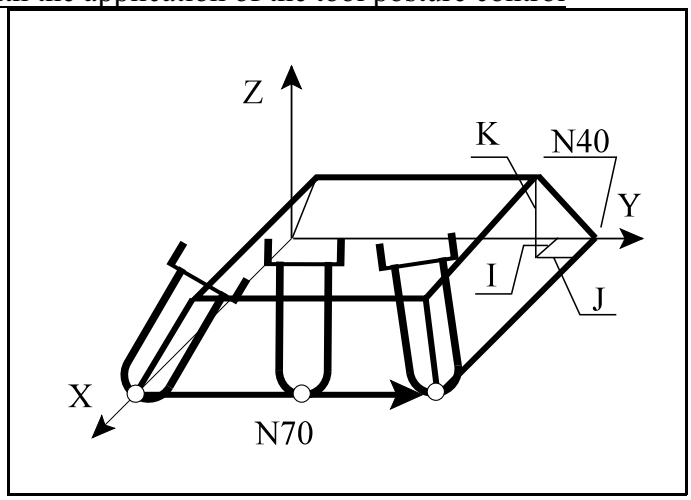


Fig. 25.16-4

```
N10 G54 G0 X150 Y150
Z100
N20 G43.5 P1 H1 Z0 I-20
J-20 K40 M3 S1000
N30 G1 X100 Y100 F1000
N40 X0 I20 J-20 K40
N50 Y0 I20 J20 K40
N60 X100 I-20 J20 K40
N70 Y100 I-20 J-20 K40
N80 G0 X150 Y150
N90 G49 Z100
...
```

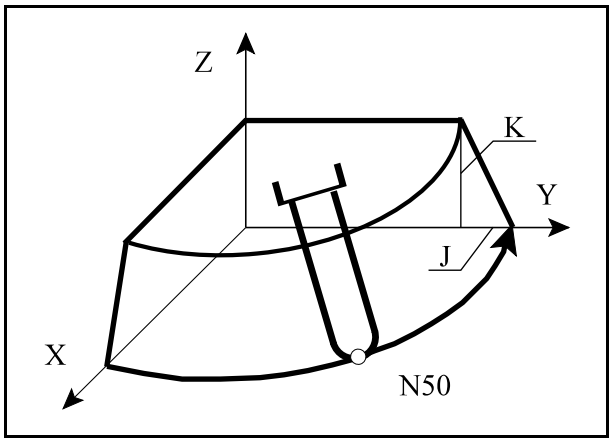


Fig. 25.16-5

An example of milling a truncated cone with the application of the tool posture control

In the following program, the 90° segment of a truncated cone is milled with the application of the tool posture control. It is the side of the tool that executes cutting operation. Interpretation of

```
I0 J-20 K40
```

programmed in the block N50 can be seen in the figure. The movement of the tool is indicated in the block N50, too.

```
N10 G17 G54 G49 X-50 Y-50 Z100
N20 G43.5 P1 H1 X0 Y0 I0 J0 K40 S1000 M3
N30 G1 Z0
N40 X100 I-20 J0 K40
N50 G3 X0 Y100 R100 I0 J-20 K40
N60 G1 Y0 I0 J0 K40
N70 G0 X-50 Y-50
N80 G49 Z100
...
```

25.17 Cutting Point Control (G43.8, G43.9)

In the case of the cutting point control (CPC: Cutting Point Control), the control, using the tool center point control or the tool posture control, moves the tool taking into account not only the length of the tool but also the shape of the tool, i.e. the radius of the tool (R) and the corner rounding (r). In this case, both the direction of the tool (the $\mathbf{v}_1$ vector in the figure) at the block-end point and the normal of the surface in the cutting point (the direction perpendicular to the surface) (the $\mathbf{v}_2$ vector in the figure) has to also be given. Then, from the length, radius and rounding data and from the two directions ($\mathbf{v}_1$, $\mathbf{v}_2$), the control will calculate automatically the way of reaching the cutting point, i.e. the programmed point.

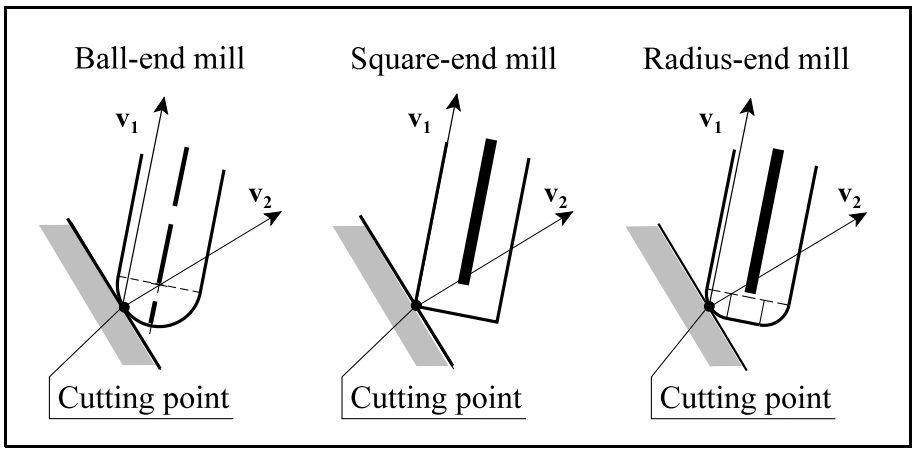


Fig. 25.17-1

There are two new column in the tool compensation table: the column ‘r geometry’ and the column ‘r wear’, where the tool rounding can be specified. The tool rounding can be referred at the D address which, at the same time, calls also the R tool radius and the r tool rounding.

	Length compensation L: code H		**Radius compensation R and rounding r: code D**				**Tool position: Q**
Compen-sation number	**L geometry**	**L wear**	**R geometry**	**R wear**	**r geometry**	**r wear**	
1	350.000	-0.130	5.000	-0.012	5.000	-	0
2	830.000	-0.102	10.000	-0.008	6.000	0.010	0
3	400.000	.	30.000	-0.160	0.000	-	0
.	.	.	.	.		0.006	.
.	.	.	.	.		0.000	.

Interpretation of the compensations above can be found in the figure below.

If ***r=0***: the tool is square-end mill and the tool radius is R.
If ***r<R***: the tool is radius-end mill with the radius r and the tool radius is R.
If ***r=R***: the tool is ball-end mill and its radius is R.

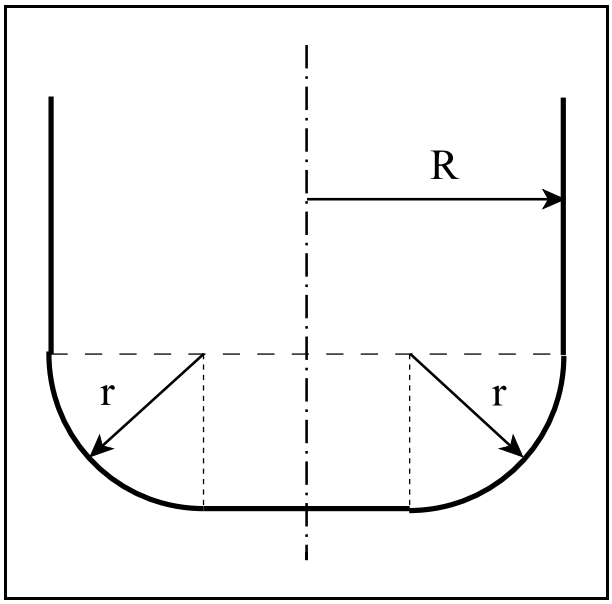


Fig. 25.17-2

Type 1 Cutting point control (G43.8)

The instruction

G43.8 (P1) X Y Z Θ Ψ H D ,L2 I J K

switches the Type 1 cutting point control and ***calls the values of the length compensation, the radius compensation*** and the tool ***corner rounding radius***. The instruction G43.8 can be issued only in the state G0 or G1.

If

P=0, or the address P is not filled: ***the cutting point control will be executed with tool center point control;***

P=1: ***the cutting point control will be executed with tool posture control***.

X, Y, Z: The block end positions of the linear main axes in the case of the absolute data specification or their movements in the case of incremental programming.

Θ, Ψ: The angles determining the direction of the tool ($\mathbf{v_1}$); the block end positions of the rotary axes 1 and 2 participating in 5-axis machining in the case of the absolute data specification or their movements in the case of incremental programming.

H: The number of the length compensation to be taken into account.

D: The number of the radius compensation and the tool corner rounding radius at the same time to be taken into account.

I, J, K ***after the address ,L2***:

At the end point of the block, the direction perpendicular to the surface ($\mathbf{v_2}$), looking from the coordinate system fixed to the table.

The vector components specified at the address I, J, K need not create unit vector. For a more accurate angle calculation, it is advisable to specify the components using as many digits as possible.

In the state G43.8, it is possible to program G0, G1, G2 and G3 movements.

Positioning and linear interpolation in the state G43.8
In the blocks ***G0*** and ***G1*** following the ***G43.8***:

G0 (G1)
X Y Z Θ Ψ ,L2 I J K
X Y Z Θ Ψ ,L2 I J K
...
X Y Z Θ Ψ ,L2 I J K

the control manages the programmed point by the use of the cutting point control already, taking into account the length, the radius and the corner round radius of the tool, irrespective of whether the tool or the workpiece is tilting.

Circular interpolation in the state G43.8
In the blocks ***G2*** and ***G3*** following the ***G43.8***:

$$G17\begin{Bmatrix}G2\\G3\end{Bmatrix}X\_Y\_Z\_\Theta\_\Psi\_\begin{Bmatrix}R\_\\I\_J\_\end{Bmatrix},L2\_I\_J\_K\_F\_$$

$$G18\begin{Bmatrix}G2\\G3\end{Bmatrix}X\_Z\_Y\_\Theta\_\Psi\_\begin{Bmatrix}R\_\\I\_K\_\end{Bmatrix},L2\_I\_J\_K\_F\_$$

$$G19\begin{Bmatrix}G2\\G3\end{Bmatrix}Y\_Z\_X\_\Theta\_\Psi\_\begin{Bmatrix}R\_\\J\_K\_\end{Bmatrix},L2\_I\_J\_K\_F\_$$

the control manages the programmed point by the use of the cutting point control already, irrespective of whether the tool or the workpiece is tilting.

Type 2 Cutting point control (G43.9)
The instruction

G43.9 (P1) X Y Z I J K H D ,L2 I J K

switches the Type 2 cutting point control and ***calls the values of the length compensation, the radius compensation*** and the tool ***corner rounding radius***. The instruction G43.9 can be issued only in the state G0 or G1.
Ha
P=0, or the address P is not filled: ***the cutting point control will be executed with tool center point control***;
P=1: ***the cutting point control will be executed with tool posture control***.

X, Y, Z: The block end positions of the linear main axes in the case of the absolute data specification or their movements in the case of incremental programming.
I, J, K: At the end point of the block, the X-, Y- and Z-direction components of the tool-direction vector ($\mathbf{v_1}$) in relation to the workpiece.
H: The number of the length compensation to be taken into account.
D: The number of the radius compensation and the tool corner rounding radius at the same time, to be taken into account.

I, J, K ***after the address ,L2***:
At the end point of the block, the direction perpendicular to the surface ($\mathbf{v_2}$), looking

from the coordinate system fixed to the table.
The vector components specified at the address I, J, K need not create unit vector. For a more accurate angle calculation, it is advisable to specify the components using as many digits as possible.

In the state G43.9, it is possible to program G0, G1, G2 and G3 movements.
In the state G43.9, the addresses of the rotary axis 1 and the rotary axis 2 cannot be referred to; when they are referred, the error message 'Specification error G43.9' will be sent by the control.

Positioning and linear interpolation in the state G43.9
In the blocks ***G0*** and ***G1*** following the ***G43.9***:

G0 (G1)
X Y Z I J K ,L2 I J K
X Y Z I J K ,L2 I J K
...
X Y Z I J K ,L2 I J K

the control manages the programmed point by the use of the cutting point control already, taking into account the length, the radius and the corner round radius of the tool, irrespective of whether the tool or the workpiece is tilting.

Circular interpolation in the state G43.9
In the state G43.9, circle interpolation ***can only be programmed by specifying the radius of the circle***, because the addresses I, J, K are used to set the tool direction or the direction of the normal of the surface.
In the blocks ***G2*** and ***G3*** following the ***G43.9***:

$$G17\begin{Bmatrix}G2\\G3\end{Bmatrix}X\_Y\_Z\_I\_J\_K\_R\_,L2\_I\_J\_K\_F\_$$

$$G18\begin{Bmatrix}G2\\G3\end{Bmatrix}X\_Z\_Y\_I\_J\_K\_R\_,L2\_I\_J\_K\_F\_$$

$$G19\begin{Bmatrix}G2\\G3\end{Bmatrix}Y\_Z\_X\_I\_J\_K\_R\_,L2\_I\_J\_K\_F\_$$

the control manages the programmed point by the use of the cutting point control already, irrespective of whether the tool or the workpiece is tilting.

Output θ, ψ angle pair calculated from the I-, J-, K-component vector
For G43.9 P0 the foregoing for tool center point control (G43.5) apply, and for G43.9 P1 the rules for tool posture control (G43.5 P1) apply.

The programmer's coordinates to be taken into account for the cutting point control
The foregoing for tool center point control are valid, i.e. the axes have to always be programmed in the coordinate system fixed to the workpiece even in the case, when the workpiece rotates.

Interpretation of the F feed in the case of the tool posture control
The foregoing for tool center point control are valid, i.e. the programmed F feed is always the velocity of the tool center point relative to the workpiece.

Data specification error in the case of the cutting point control with tool posture control
In the case of programming the G43.8 P1 and G43.9 P1, if the block cannot be executed using the block start and block end tool directions, the error message

'2192 Invalid pusture in TPC: xx'

will be sent by the control. See the Tool posture control.

The case when the angle between the direction of the tool and the direction perpendicular to the surface greater than 90°
The angle between the direction of the programmed tool (the $\mathbf{v}_1$ vector in the figure) and the normal of the surface in the cutting point (the direction perpendicular to the surface) may be even greater than 90°. It supposes tool shape and concave surface illusztrated in the figure.

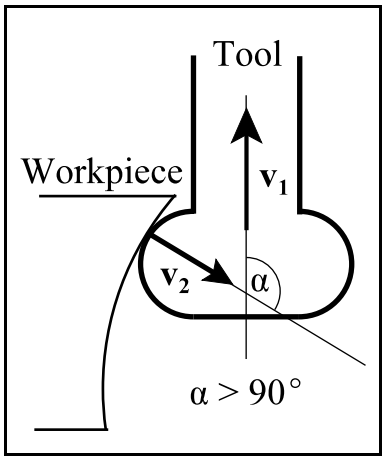


Fig. 25.17-4

Switching the cutting point control off (G49)
The instruction

G49

switches the cutting point control off.
Only positioning (G0) or linear interpolation (G1) can be programmed together with G49:

G49 G0 (G1) X Y Z Θ Ψ

An example for the cutting point control

The sample program below goes around the workpiece due to the instruction G43.9 using the cutting point control and the tool posture control.
The motion starts in the direction of the X axis (N40) and a side face of a pyramid will be machined.
The tool direction at the starting point ($\mathbf{v}_1$) in the N20 block is specified by the I20 J20 K40 data while the direction perpendicular to the side face of the pyramid ($\mathbf{v}_2$) is specified by the ,L2 I0 J-40 K20 vector components.

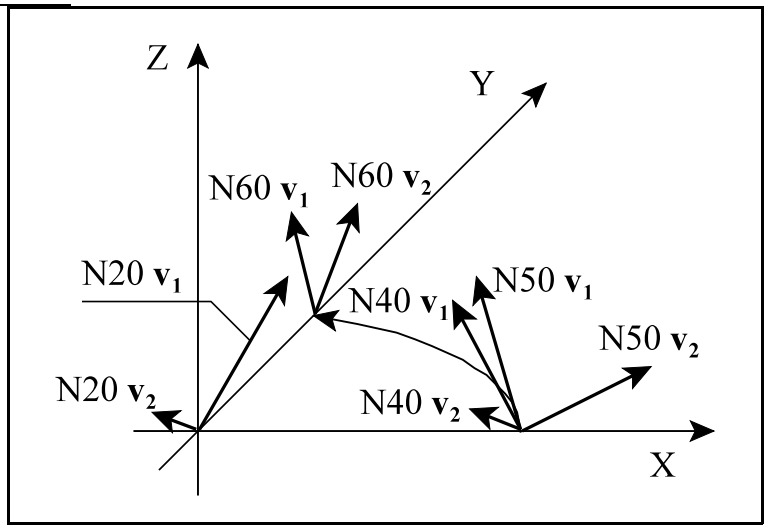


Fig. 25.17-5

At the end of the N40 block, the direction of the tool ($\mathbf{v}_1$) will be I-20 J20 K40 while the direction perpendicular to the surface ($\mathbf{v}_2$) will be unchanged.
In the N60 block, the tool has to be moved along the curved surface of a cone. For this, in the block N50, the tool direction ($\mathbf{v}_1$) has to be set in the direction of the generatrix of the cone I-20 J0 K40 while the direction perpendicular to the surface ($\mathbf{v}_2$) will be set by the ,L2 I40 J0 K20 data queue.
At the end point of the N60 block, the direction of the generatrix of the cone, i.e. the direction of the tool ($\mathbf{v}_1$) will be I0 J-20 K40 while the direction perpendicular to the surface ($\mathbf{v}_2$) will be ,L2 I0 J40 K20.

```
...
N10 G17 G54 G49 G0 X-50 Y-50 Z100 S1000 M3
N20 G43.9 P1 H1 D1 X0 Y0 I20 J20 K40 ,L2 I0 J-40 K20
N30 G1 Z0 F1500
N40 X100 I-20 J20 K40
N50 I-20 J0 K40 ,L2 I40 J0 K20
N60 G3 X0 Y100 R100 I0 J-20 K40 ,L2 I0 J40 K20
N70 I20 J-20 K40 ,L2 I-40 J0 K20
N80 G1 Y0 I20 J20 K40
N90 G0 X-50 Y-50 I0 J0 K40
N100 G49 Z100
...
```

25.18 Smooth Interpolation in the Case of Tool Centre Point Control

In the case of 5D machining, the smooth interpolation can be switched on even if the the tool centre point is being guided (G43.4, G43.5), too. In such cases, always the fine smoothing (G5.1 Q3) has to be used.
In the course of smoothing, the path specified by coordinates X, Y and Z will be managed by the control in the way described at fine smoothing. See the subsection '4.7.1 Fine smoothing (G5.1 Q3)' on page 55.
In such a case, in addition to the coordinates X, Y and Z, motion of the first and the second rotary axes has to also be smoothed in order that motion of the rotary axes is more steady and the machining time decreases.

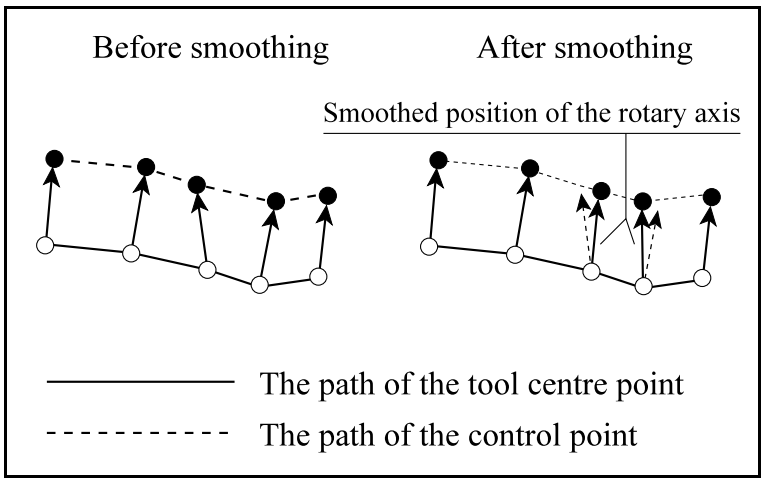


Fig. 25.18-1

By modification of the position of the rotary axes, only the position of the control points will change; the path of the the tool centre point remains unchanged as it is illustrated in the figure above. The control changes the position of the rotary axes only within the range specified in parameter.

The instruction

G5.1 Q3 X0 Y0 Z0 Θ0 Ψ0 5D smoothing on

switches the 5D smoothing on, where

Θ: the name of the first rotary axis;

Ψ: the name of the second rotary axis.

The instruction, at the same time, switches on the mode of multiple block preprocessing and the mode of high-speed and high-precision (HSHP) machining, too.
The instruction

G5.1 Q0 smoothing and HSHP off

switches off the mode of 5D smoothing and the mode of high-speed and high-precision machining.
The instruction

G5.1 Q1 5D smoothing off

switches the 5D smoothing off, but it keeps the mode of high-speed and high-precision machining switched on.
After turn-on, program end and reset, the control always switches the 5D smoothing off.

When the 5D smoothing is switched on, it is forbidden to switch off or on the tool length compensation. Otherwise, tbe control will send the error message ‘2188 G<code> not allowed in FineSmooth mode’.

Correct

```
...
G43.4 X Y Z B C
G5.1 Q3 X0 Y0 Z0 B0 C0 (5D
smoothing on)
G1 X Y Z B C
....
....
G5.1 Q0 (5D smoothing off)
G49
...
```

Wrong

```
...
G5.1 Q3 X0 Y0 Z0 B0 C0 (5D
smoothing on)
G43.4 X Y Z B C
G1 X Y Z B C
....
....
G49
G5.1 Q0 (5D smoothing off)
...
```

During 5D smoothing, in per-block mode, the control stops always at the corrected pont but not at the programmed one.

The parameters

N3100 Max. Distance of a Block
N3101 Min. Distance of a Block
N3102 Angle Cancelling Smooth
N3103 Accuracy
N3106 Tolerance of Smooth2
N3107 Dist. Cancelling of Angular Calc. in Smooth2

also apply to the 5D smoothing in the way described at fine smoothing. See the subsection ‘4.7.1 Fine smoothing (G5.1 Q3)’ on page 55.
In the case of 5D smoothing, the following parameter is also used by the control:

N3108 Tolerance of Rot. Ax. in Smooth2
It is the parameter, which limits modification of programmed position of the rotary axes in the case of 5D smoothing, i.e. the maximum tolerance of the smoothed position from the programmed position. It can be specified in degree to both rotary axes involved in 5D machining.

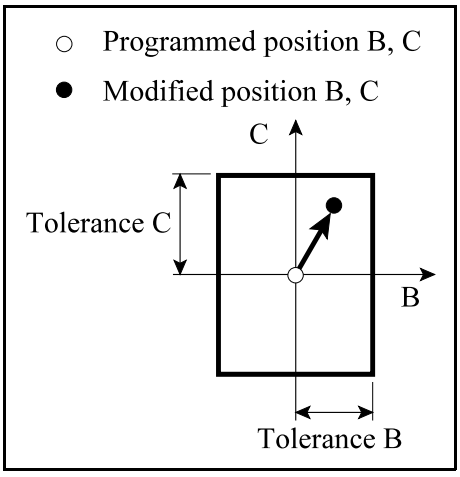


Fig. 25.18-2

25.19 Position Information of the Macro Language in the Case of 5D Machining

Position information

The following position information can be used in the case of 5D machining too:

Instantaneous position of the axes in the workpiece coorditate system:

#5041...#5060, #100101...#100150, #_ABSOT[n]

Skip position of the axes in the workpiece coorditate system:

#5061...#5080, #100151...#100200, #_ABSKP[n]

If any of the 3D rotations (G68.1, G68.2) is switched on, interpretation of instantaneous positions and Skip positions, which can be read out from macro variables above can be influenced by parameter setting.

If the ***bit #5 PCS of the parameter N1755 Macro Contr*** is

=0: they will be read back in the workpiece coordinate system;

=1:they will be read back in the programmer's coordinate system, after rotation back. In this case, the length compensation will also be taken into account with rotation back if TLC=0.

In addition, it can be influenced by parameter setting whether the length compensation is contained in the information mentioned above, or not.

If the ***bit #6 TLC of the parameter N1755 Macro Contr*** is

=0: the length compensation will be contained in the position information (it is the basic case of position information read-out);

=1: the length compensation components will be taken off from position information.

Number	**Position information**			
	PCS=0 TLC=0	**PCS=1 TLC=0**	**PCS=0 TLC=1**	**PCS=1 TLC=1**
#5041...#5060 #5061...#5080	workpiece coordinate system with length compensation	programmer's coordinate system with length compensation	workpiece coordinate system without length compensation	programmer's coordinate system without length compensation
#100101...#100050 #100151...#100200				
#_ABSOT[n] #_ABSKP[n]				

Reading the position information out with taking length compensation into account:

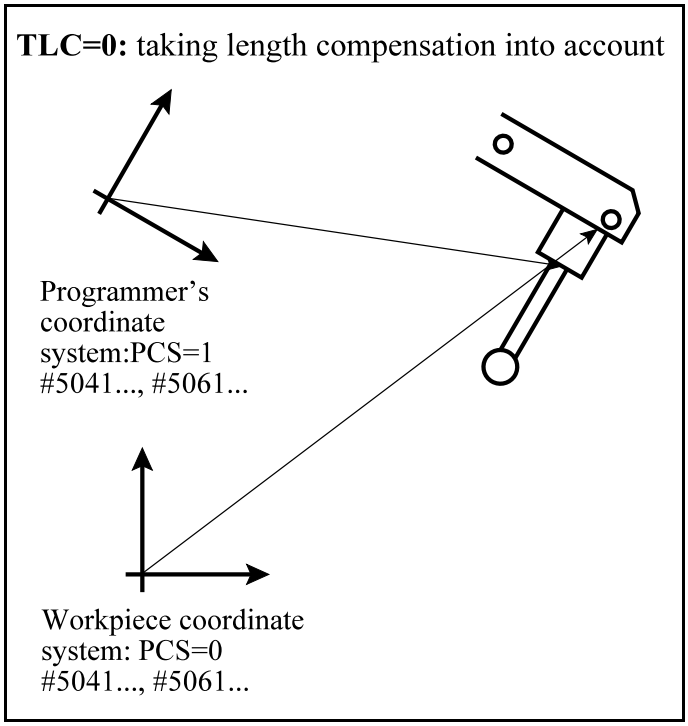


Fig. 25.19-1

Reading the position information out without taking length compensation into account:

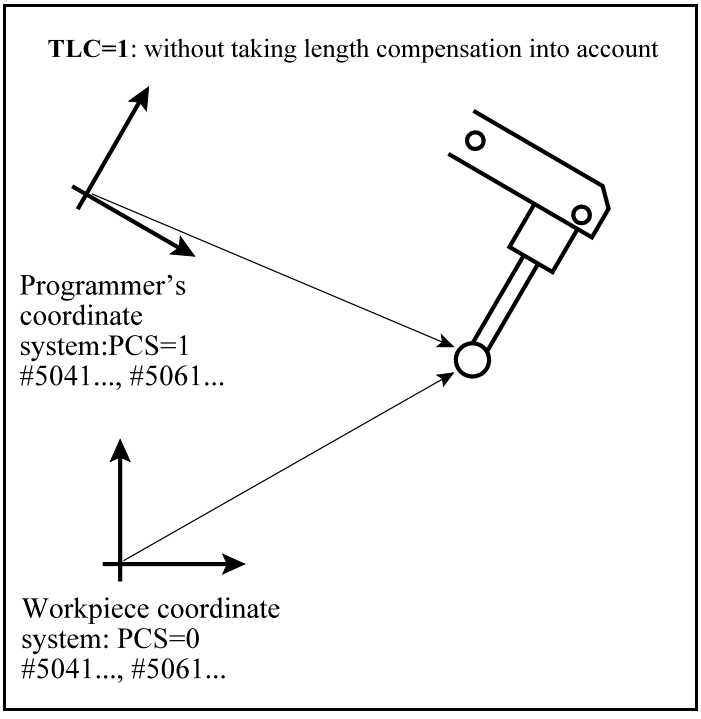


Fig. 25.19-2

New skip positions

The following two new skip positions have been introduced:

#151001...#151050, which return positions of the workpiece coordinate system or the programmer's coordinate system, and which, in terms of the parameter PCS, act contrary to the variables #5061...#5080, #100151...#100200. The parameter TLC exerts the same influence on both of them.

#151101...#151150, which return position always in the machine coordinate system, but with taking the parameter TLC, i.e. they containe the length compensation or not.

<table>
<tr><th rowspan="2">Number</th><th colspan="4">Skip position returned</th></tr>
<tr><th>PCS=0
TLC=0</th><th>PCS=1
TLC=0</th><th>PCS=0
TLC=1</th><th>PCS=1
TLC=1</th></tr>
<tr><td>#5061...#5080</td><td rowspan="2">workpiece coordinate system with length compensation</td><td rowspan="2">programmer's coordinate system with length compensation</td><td rowspan="2">workpiece coordinate system without length compensation</td><td rowspan="2">programmer's coordinate system withou length compensation</td></tr>
<tr><td>#100151...#100200</td></tr>
<tr><td>#151001...#151050</td><td>programmer's coordinate system with length compensation</td><td>workpiece coordinate system with length compensation</td><td>programmer's coordinate system without length compensation</td><td>workpiece coordinate system without length compensation</td></tr>
<tr><td>#151101...#151150</td><td colspan="2">machine coordinate system with length compensation</td><td colspan="2">machine coordinate system without length compensation</td></tr>
</table>

Skip positions with taking length compensation into account:

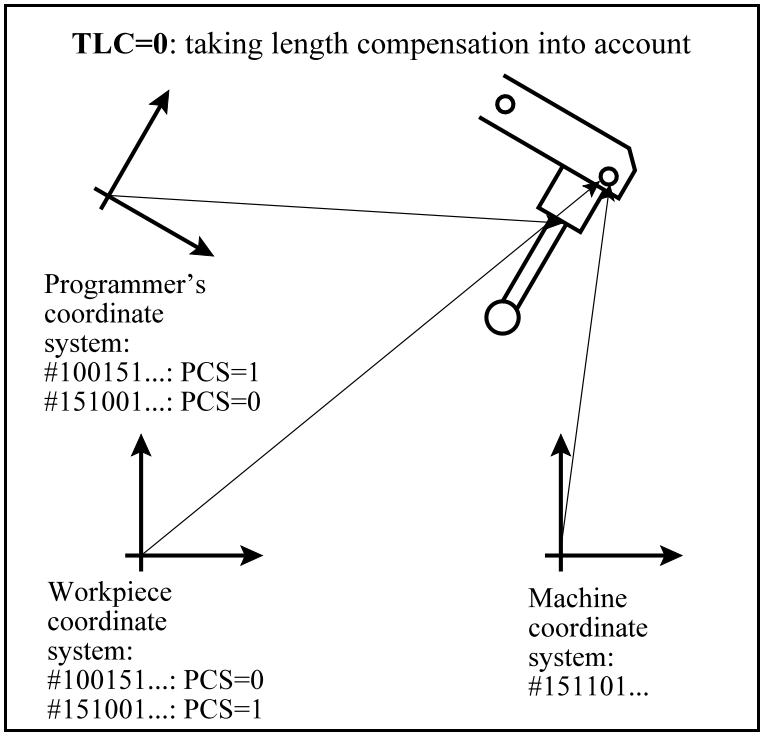


Fig. 25.19-3

Skip position without taking length compensation into account:

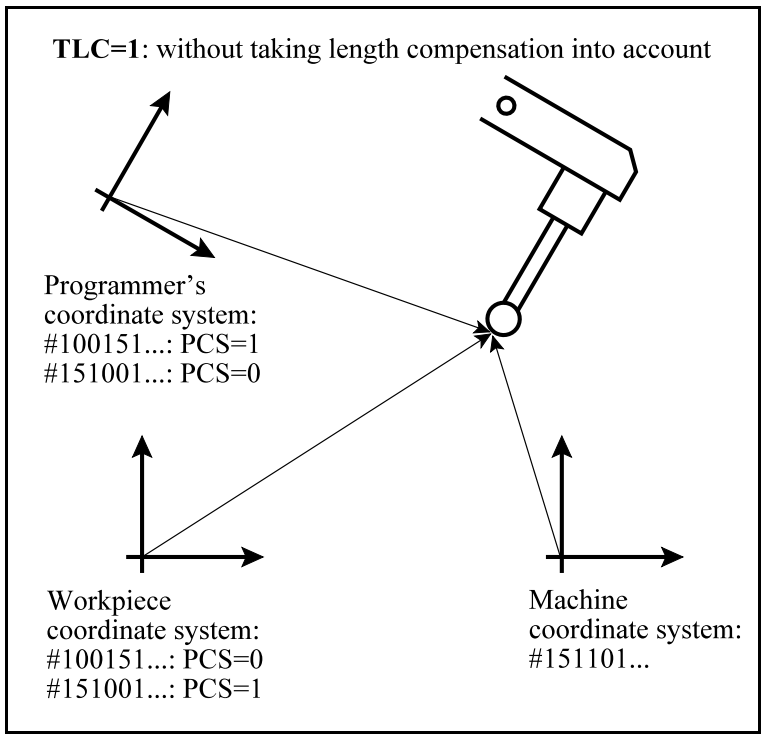


Fig. 25.19-4

25.20 Compensating the Angular Position of the Workpiece on 5-axis Machines

Enabling misalignment compensation for rotary axes, on 5-axis macines

On 5-axis machines, if both rotary axes can be moved continuously, the misalignment compensation can be taken into account for the rotary axes.
On Hirth disc rotary axes, when the rotary axes can only move to a discrete position, it is not possible to take into account the misalignment compensation for the rotary axes.
Interpretation of #4 **BSE** in the parameter N3212 5D Control parameter is as follows:
If the value of the parameter:

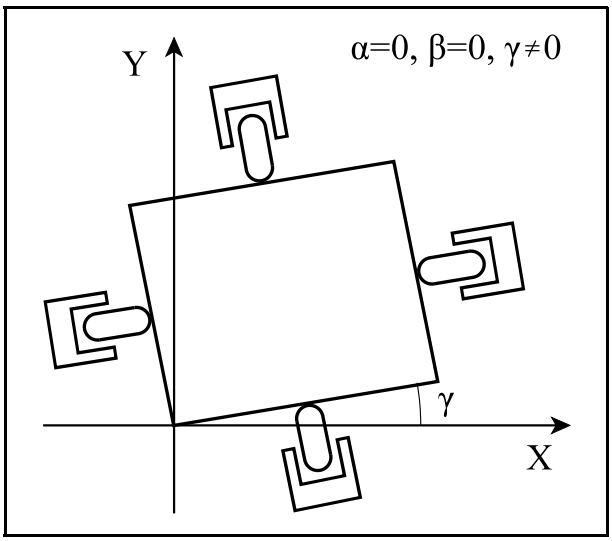
=0: the misalignment compensation will not be taken into account for the rotary axes,
=1: the misalignment compensation will be taken into account for the rotary axes.

If the value of the parameter bit BSE is 0, the misalignment compensation on the five-axis machine works as described for three-axis machines.

If the value of the parameter bit BSE 1, the misalignment compensation will also be applied for the two rotary axes in 5D function.

In the case where the misalignment compensation is about the tool axis (the Z axis in figure below):

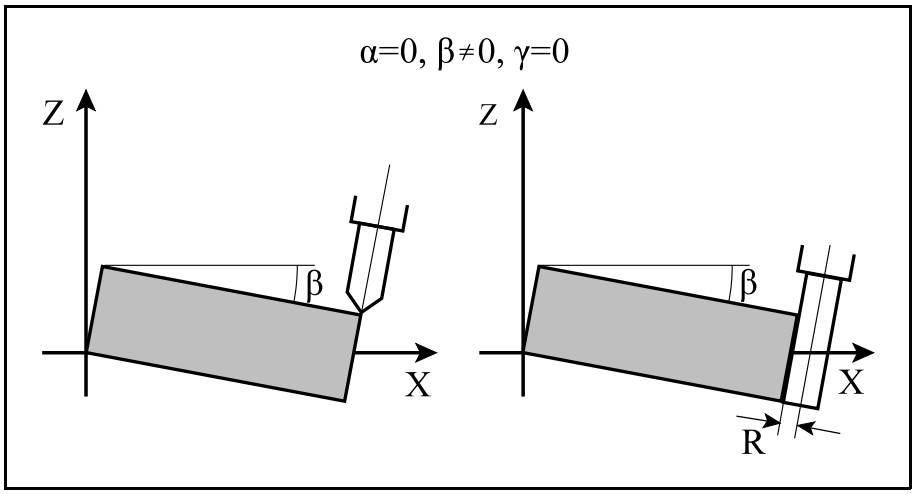
– Operations on the side of the workpiece, e.g. direction of the hole, will be perpendicular to the lateral surface of the workpiece,



25.20Fig.

In the case where the misalignment compensation is about the axis perpendicular to the tool (the Y axis in figure below):

– Both the tool endpoint position and the tool direction is good, i.e., it will be perpendicular to the upper plane of the workpiece,
– In the case of milling a side, the tool will mill the side perpendicular to the upper plane of the workpiece.



25.20Fig.

On the five-axis machines, when BSE=1, the axes taking part in compensation of angular position of the workpiece are the following:

X, Y, Z: the 3 main axes,

Θ, Ψ: the rotary axes 1 and 2 (if there are) taking part in 5-axis motions,

α, β, γ: the angles of misalignment compensation, α in the G19 plane, β in the G18 plane, γ the G17 plane.

	X	Y	Z	Θ	Ψ	G17(γ)	G18(β)	G19(α)
G54								
...								
G54.1 P								
...								
G54.2 P								
...								

The misalignment compensation transformations applied to the longitudinal axes are the same as those described on page 93 in the subchapter ‘10.2.3 Compensating the angular position of the workpiece’.

The misalignment compensation transformations ***are always taken into account in the current workpiece coordinate system, when there are motion commands***.
Never applies to commands on moving to machine position (G53, G28, G30).

In the case of manual moving, the switch ‘In accordance with misalignment compensation‘ (the flag CP_WMCAXF PLC) will be taken into account, i.e., rotation about all three axes will apply.

Note: ***On rotary axes, the zero point offsets*** *are always taken into account* ***linearly*** *by the control* ***in 3D functions,*** *i.e., machine position − zero point offset****.*** *However,* ***in 5D functions,*** *the zero point offset is managed through* ***transformations****.*

Application of workpiece compensation on 5-axis continuous head-head machine tool

On head-head machine tools, in 5D functions, the zero point offset (Θ_0, Ψ_0) on the rotary axes 1 and 2 is not taken into account by the control*.*
Compensation of the workpiece position should always be done with misalignment compensation, i.e., with specifying α, β, γ, because this is what the block preparator takes into account during programmed X, Y, Z, Θ, Ψ motions.
Only the misalignment compensation modifies the programmed angles of the rotary axes.

On the head-head machine tools, the following functions switch on the misalignment compensation on rotary axes:

G43.1: length compensation in the direction of the tool axis
G43.4, G43.5: tool center point control
G43.4 P, G43.5 P1: tool posture control
G43.8, G43.9: cutting point control
G53.1, G53.6: positioning the rotary axes in the tool direction **after G68.2**

☞ ***Attention!***

After specifying G43.1, G43.4, G43.8, absolute positioning must be carried out on both rotary axes in order to misalignment compensation is valid!
After specifying G43.5 and G43.9, the tool direction must always be specified at the I, J, K addresses.

CORRECT

```
G54 X100 Y20 Z600
G43.1 H1 A30 C60
X50 Y-20 Z10
```

or

```
G54 X100 Y20 Z600
G43.1 H1
A30 C60
X50 Y-20 Z10
```

WRONG

```
G54 X100 Y20 Z600 A30 C60
G43.1 H1
X50 Y-20 Z10
```

or

```
G54 X100 Y20 Z600
A30 C60
G43.1 H1
X50 Y-20 Z10
```

Application of workpiece compensation on 5-axis continuous table-table machine tool

On table-table machine tools, in 5D functions, the zero point offset on the rotary axes 1 and 2 is taken into account by the control*.*

☞**Attention:** *When both the zero point offset of the rotary axes and the misalignment compensation are applied to a given workpiece, the zero point offset of the rotary axes must always be measured first and then the misalignment compensation angles must be determined at 0 degree workpiece zero point position of the rotary axes.*

In addition, the angles of the misalignment compensation set are also taken into account by the block preparator during programmed X, Y, Z, Θ, Ψ motions.
The programmed angles of the rotary axes are modified first by the misalignment compensation, then by the zero point offset.

On the table-table machine tools, the following functions switch on the misalignment compensation on rotary axes:

G43.4, G43.5: tool center point control
G43.4 P, G43.5 P1: tool posture control
G43.8, G43.9: cutting point control
G53.1, G53.6: positioning the rotary axes in the tool direction **after G68.2**
G54.2: dynamic managing the zero point of the rotary tables

☞ ***Attention!***

After specifying G43.4, G43.8, absolute positioning must be carried out on both rotary axes in order to misalignment compensation is valid!
After specifying G43.5 and G43.9, the tool direction must always be specified at the I, J, K addresses.
After specifying G54.2 absolute positioning must be carried out on both rotary axes in order to misalignment compensation is valid!

CORRECT

```
G54 X100 Y20 Z600
G54.2 P1 A30 C60
X50 Y-20 Z10
```

or

```
G54 X100 Y20 Z600
G54.2 P1
A30 C60
X50 Y-20 Z10
```

WRONG

```
G54 X100 Y20 Z600 A30 C60
G54.2 P1 H1
X50 Y-20 Z10
```

or

```
G54 X100 Y20 Z600
A30 C60
G54.2 P1
X50 Y-20 Z10
```

Application of workpiece compensation on 5-axis continuous head-table machine tool

On head-table machine tools, in 5D functions, the zero point offset is taken into account by the control only on the table, i.e., on the rotary axes 2.

☞**Attention:** *When both the zero point offset of the rotary axes and the misalignment compensation are applied to a given workpiece, the zero point offset of the rotary axes must always be measured first and then the misalignment compensation angles must be determined at 0 degree workpiece zero point position of the rotary axes.*

In addition, the angles of the misalignment compensation set are also taken into account by the block preparator during programmed X, Y, Z, Θ, Ψ motions.

The programmed angles of the table are modified first by the misalignment compensation, then by the zero point offset. The programmed angle of the head is modified only by the misalignment compensation.

On the head-table machine tools, the following functions switch on the misalignment compensation on rotary axes:

G43.1: length compensation in the direction of the tool axis
G43.4, G43.5: tool center point control
G43.4 P, G43.5 P1: tool posture control
G43.8, G43.9: cutting point control
G53.1, G53.6: positioning the rotary axes in the tool direction **after G68.2**
G54.2: dynamic managing the zero point of the rotary tables

☞ ***Attention!***

After specifying G43.1, G43.4, G43.8, absolute positioning must be carried out on both rotary axes in order to misalignment compensation is valid!
After specifying G43.5 and G43.9, the tool direction must always be specified at the I, J, K addresses.
After specifying G54.2 absolute positioning must be carried out on both rotary axes in order to misalignment compensation is valid!

CORRECT

```
G54 X100 Y20 Z600
G43.1 H1
G54.2 P1 A30 C60
X50 Y-20 Z10
```

or

```
G54 X100 Y20 Z600
G43.1 H1 A30 C60
G54.2 P1
X50 Y-20 Z10
```

or

```
G54 X100 Y20 Z600
G43.1 H1
G54.2 P1 A30 C60
X50 Y-20 Z10
```

WRONG

```
G54 X100 Y20 Z600 A30 C60
G43.1 H1
G54.2 P1 H1
X50 Y-20 Z10
```

or

```
G54 X100 Y20 Z600
A30 C60
G43.1 H1
G54.2 P1
X50 Y-20 Z10
```

Notes