AMK

AMKASYN

VARIABLE SPEED DRIVES

AZ-MC1 Multi-station Multi-channel CNC contouring control

ISO programming

Rights reserved to make technical changes

<Din9807e.doc> Part-No.: 27875

AMK

Arnold Müller, Antriebs- und Steuerungstechnik GmbH & Co.KG, D-73230 Kirchheim/Teck,
Tel.: 07021/50 05-0, Telefax: 07021/50 05-176

Contents

1 OVERVIEW OF THE LANGUAGE SCOPE

The decoder of the AMK MC1 control system processes NC control data according to DIN66025. Extended functions for program influencing are processed in addition. NC control data are NC programs which are filed as files in the NC program memory of the control system. They can be produced on the AMK control panel or with any text editor on a PC.

The following tasks are performed by the decoder:

⇒ Decoding and processing distance information according to DIN 66025
- * Selection of the linear interpolation (G01,G00) and of the circular interpolation in a main plane (G02, G03) with decoding of the circle center data I, J, K,
- * Selection of the dwell (G04, $TIME),
- * Selection of the feed adaptation at the start or end of the block (G08,G09),
- * Selection of an interpolation main plane (G17, G18, G19),
- * Selection of the radius offset (G40, G41, G42).
- * Selection of a zero offset from a table (G53, G54, G55, G56, G57, G58, G59),
- * Selection of the in-position programming at the end of the current block (G60),
- * Holding override at 100% (G63),
- * Statement of coordinates in mm or inch (G71, G70),
- * Triggering homing cycle (G74) axis-specifically,
- * Statement of coordinates absolute or relative (G90 and G91),
- * Datum point offset relative (G92),
- * Revolution feed ON/OFF (G94/G95)
- * Statement of software limit switch values in negative or positive direction (G98, G99),

⇒ Control blocks based on the programming language C

* Conditional jumps :	$IF, $ELSEIF, $ELSE, $ENDIF, $SWITCH, $CASE, $DEFAULT, $ENDSWITCH, $BREAK,
* Counter loops :	$FOR, $ENDFOR, $CONTINUE, $BREAK,
* Loops with run condition:	$WHILE, $ENDWHILE, $CONTINUE, $BREAK,
* Loops without run condition:	$DO, $ENDDO, $CONTINUE, $BREAK.

⇒ Processing coordinate notation A, B, C, ..., U, V, W, unless otherwise used,

⇒ Decoding of feed speed (F) in MM/MIN, INCH/MIN, DEGREE/MIN or NEWDEGREE/MIN for CONTOURING AXES and for several STRAIGHT-CUT AXES

⇒ Decoding technological information:
- * Spindle speed in rpm (S),
- * Tool data call (D0..D40) with length and radius calculating the length in the tool axis (for G17 Z axis, G18 Y axis, G19 X axis) and axis-specific axis offsets for controlling the tool radius offset for offset path calculation with the tool radius,
- * Selection of the tool location (T1..T40),
- * Additional functions (M0...M99) with configurable effect of each function:
- * Auxiliary functions (H),

⇒ Differentiating between global (reachable by all main programs) and local subroutines (reachable only from the associated main program); here the maximum nesting depth = 10.

⇒ Parameter calculation with:
- * Arithmetic basic operations:
  (+ , - , * , / , ** , MOD),
- * Numerical functions
  (ABS,SQR,SQRT,EXP,LN,DEXP,LOG),
- * Angle functions
  (SIN,COS,TAN,ASIN,ACOS,ATAN),
- * Conversion functions
  (INT,FRACT),
- * Grouping intermediate results with the aid of brackets [ ],
- * Direct and indirect addressing of parameters,
- * Mathematical expressions are allowed for parameter designation. Parameters are allowed within mathematical expressions, in this way multiplied indexed parameter programming is possible,

⇒ Possibility of block deletion (/),

⇒ Extensive adaptation through machine data record (MDS),

2 PROGRAMMING BASICS

2.1 Character and number formats

2.1.1 Character set and file format

The decoder used processes NC control data in the ASCII file format. NC control data can be produced with the programming system or editor integrated in the operating system. NC control data can also be produced with PC-based text editors. Refer to the "Description NC operation" for the handling and copying of NC control files in the AMK CNC system.

The decoder processes the following characters:

Letters	{ A B C ... Z a b c ... z }
Numbers	{ 1 2 3 ... 9 0 }
Special characters	{ ! @ # $ % & * - _ = ~ { } ( ) [ ] , ; : " < > / ? }
Control character = HT	Horizontal tabulator TAB
SP Space	SPACE
CR Carriage return	CARRIAGE RETURN
LF Line feed	LINE FEED

2.1.2 Numerical entry

Numbers are entered as integer or decimal numbers. Decimal places are basically separated with a ".", whereby leading zeros can be left out. Length and position values are entered according to configuration and programming in mm or inch, angles are entered in degree (deg) or new degree (ndeg).

Example:

Values: (µm)	Entries: (mm)
0.1	0.0001 or .0001
1	0.001 or .001
10	0.01 or .01
100	0.1 or .1
1000	1.0 or 1
10000	10.0 or 10
100000	100.0 or 100
1000000	1000.0 or 1000

The entry range of numbers is limited only by the internal number notation.

This number notation allows travels in the range of 200 m with a resolution of 0.1µm!

2.2 Part program structure

NC programs are part of the NC control data in addition to tool data, zero offset data etc. NC programs describe the course of machining processes.

A NC program consists of:

* **Program start identification**
* Individual **NC blocks**
* **Program end identification**

The **program start** consists of the **%** character followed by a maximum 8-digit number (%nnnn). This program start identification corresponds to DIN66025 but is not essential for the decoder. Using the program start identification is recommended for reasons of program documentation and archiving. The AMK CNC system uses the file name agreed in the file system for recognising the part program.

The **NC blocks** of a part program contain the actual control information. Each block starts as a rule with a block number consisting of the **N** character followed by a maximum 4-digit number (Nnnnn). The decoder of the AMK CNC control system does not depend upon the presence of N numbers. However, their use serves for making NC programs clearer. They should be used in ascending numbering in steps of 5 or 10, which enables NC blocks to be inserted subsequently. Furthermore, a block consists of individual block words (see below) which are separated from one another by delimiters. The end of block character, blanks, tabulators, the character with the ASCII notation 0, the file end character or comments are interpreted by the decoder as delimiters.

The command **M30 or M02** must be used as **program end identification**, subroutines must be terminated with **M29**.

2.3 NC block structure

A NC block consists of a
block number (optional), a
number of **words** and an
end of block identification.

Each block starts as a rule with a block number, consisting of a **N** character followed by a maximum 4-digit number. However, the block number is of no importance for the program run.

Example of a block structure:

Example of a NC program structure:

Without numbering	Partial numbering	Complete numbering
% 100	% 100	% 100
"Block 1"	N10 "Block 1"	N10 "Block 1"
"Block 2"	"Block 2"	N20 "Block 2"
"Block 3"	N20 "Block 3"	N30 "Block 3"
.	"Block 4"	N40 "Block 4"
.	.	.
.	.	.
M30	M30	N700 M30

Words must be differentiated with regard to their meaning into words with

- * Geometrical information (e.g. positions),
- * Technological information (e.g. spindle speed, feed speed, spindle clockwise running),
- * Information for controlling the program run (so-called control blocks such as counting loops),
- * Arithmetical information (e.g. calculation of a size, parameter calculation).

Several words can stand in one block, whereby the sequence of execution of the control information within the block is already determined by the control system. The individual word information of a block can generally be entered in arbitrary sequence by the programmer.

The end of block identification can be configured and can be an arbitrary control character. Normally the control character "CR" or "LF" is used.

2.3.1 Skipping blocks

Blocks can be skipped selectively by prefixing the "/" character (e.g. /N3412), i.e. the control system skips these blocks if the "Skip block" function has been switched on at the control panel. Skipped blocks serve for determining optional machining steps such as measuring loops or empty steps within a NC program.

2.3.2 Comments

Comments contain non-decodable ASCII information which must stand between the characters "(" and ")". If further "(" stand within a comment, then corresponding ")" are expected. The end of block character and the end of file character terminate the comments.

Explanatory comments can be inserted at any place in a NC program, e.g. also at the program start. Comments have no effects on execution of a NC part program. They can be of an arbitrary length, but the maximum number of characters per block must not be exceeded.

Nesting comments is possible.

Example:

```
 % 100  (Workpiece designation, date)

 N200 ... (Circular pocket)
 N300     (Comment (comment))

 N800 M30
```

2.4 Word structure

A word consists of address letters, arithmetic expressions and texts <string>. The meaning of the individual address letters is described in the following chapters. The "Assigned address letters" chapter provides an overview.

2.4.1 Mathematical expressions

Mathematical expressions are built up from numbers, parameters, operators and functions. They are evaluated by a built-in computing function.

Examples: "100", "100+20", "100+R10", "100+P10", "100*[2+R3]", "DEXP[2]", "100+SIN[R10+P20]".

Numerical values are classified into
integer numbers <int> and
decimal numbers <float> .

Arithmetical expressions <expr> are built up from numerical values, operators, functions and parameter designations, such as for example [[sin[R1*30.00] + R2] / R5].

2.4.1.1 Integers <int>

Integers are stated without decimal point. The internally permissible value range corresponds to that of "long integer" variables of the programming language C, i.e. from $-2.14*10^9$ to $+2.14*10^9$. If calculations are made internally with the unit 0.1 µm, then on entry of values in mm, the number range lies between $-2.14*10^5$ and $+2.14*10^5$. This corresponds to a theoretical travel of more than 400 metres. Due to the internal conversion of this metric value into the incremental absolute value of the axis, the actual travel depends upon the position resolution of the axis (increments/0.1µm). Negative numbers are prefixed with a "-" character, this is not allowed for positive numbers.

Examples: 0 1 2 ... -100 etc.

2.4.1.2 Decimal numbers <float>

Decimal numbers are stated with decimal point. The value range corresponds to that of integer numbers. If calculations are performed internally with the unit 0.1 µm, 4 places after the comma are accordingly taken into account on entry of values in mm. Negative numbers are prefixed with a "-" character, no sign is necessary for positive numbers.

Examples: -30.45 0.4321

2.4.1.3 Arithmetical expressions <expr>

The normal calculating rules apply in the treatment of arithmetical expressions:

- Dot before dash calculation
- The brackets rule, whereby square brackets "[ ]" must be used

Parameters are frequently used in arithmetical expressions; the notation of parameters is:
R followed by an integer, e.g. R12 .

Indirect parameters (letter P) can be used apart from direct parameters (letter "R").

Example of an arithmetical expression:

R5 = [[sin[R1*30.00] + R2] / R5]

Overview of all available calculating operations:

Basic types of calculation

Addition	+	R1 = R2 + R3 + 0.357
Subtraction	-	R1 = R2 - 0.031
Multiplication	*	R1 = R2 * [R3 + 0.5]
Division	/	R1 = R2 * R3 / [R5 + R6]
Exponent calculation	**	R1 = 2**R3 (two to the power R3)
Modulo calculation	MOD	R1 = 11 MOD 3 (== 2)

Numerical functions:

Absolute value formation	ABS [..]	R1 = ABS [R2 - R4]
Squaring	SQR [..]	R1 = SQR [R2] + SQR [R3]
Square root	SQRT [..]	R1 = SQRT[SQR[R2]+SQR[R3]]
e - function	EXP [..]	R1 = EXP [R2 * R4]
Natural logarithm	LN [..]	R1 = LN [R2] + LN [R3]
Decimal exponent	DEXP [..]	R1 = DEXP [R2]
Decimal logarithm	LOG [..]	R1 = LOG [R2]

Caution:
With the numerical functions LN, LOG and SQRT, the argument must always be greater than 0!

Comparison functions:

Comparison operations are required in loop constructions (cf. Chapter 9: "Instructions for influencing the NC program run"):

Checking for:

- Equality	==	$IF R1 == 5
- Inequality	!=	$IF R1 != 5
- Greater than or equal	>=	$IF R1 >= 10
- Smaller than or equal	<=	$IF R1 <= 10
- Smaller	<	$IF R1 < 10
- Larger	>	$IF R1 > 10

Caution:
In programming, pay attention to the difference between value allocation by "=" and the comparison function "==".

Angle functions:

Angles are stated in degrees in angle functions.

Sine	SIN [..]	R1 = SIN [R2 * 30 +10]
Cosine	COS [..]	R1 = COS [R2 * 30 +10]
Tangent	TAN [..]	R1 = TAN [R2 * 30 +10]
Arc sine	ASIN [..]	R1 = ASIN [R2 * 10]
Arc cosine	ACOS [..]	R1 = ACOS [R2 * 10]
Arc tangent	ATAN [..]	R1 = ATAN [R2 * 10]

Caution:
The argument must always be between -1 and +1 for the numerical functions ASIN and ACOS. The argument must not assume the values ... -90, 90, 270 ... degrees for the TAN numerical function.

Conversion functions:

Integer	INT [..]	Cuts off places after the decimal point
Fract	FRACT [..]	Removes the integer part

2.4.2 Assigned address letters

Fixed meanings are assigned to the following address letters in the decoder used.

Address letters concerning the technology:

Name	Format	Meaning
D	<int>	Tool offset call
F	<int, float, expr>	Feed at the block start
H	<int>	Auxiliary function
M	<int>	Switching function or M function
S	<int, float, expr>	Spindle speed, synchronous ratio etc.
T	<int>	Tool location call

Address letters concerning the geometry:

Name	Format	Meaning
G	<int>	Preparatory function
I	<int, float, expr>	Interpolation parameter for 1st contouring axis
J	<int, float, expr>	Interpolation parameter for 2nd contouring axis
K	<int, float, expr>	Interpolation parameter for 3rd contouring axis

Address letters concerning the program run:

Name	Format	Meaning
L	<int>	Subroutine call, global
LL	<int>	Subroutine call, local
N	<int>	Block number
$		Identification for control block statements and extended language elements

Address letters concerning the arithmetic:

Name	Format	Meaning
R	<int>	Direct parameter

The address letters for designating the numerical axes can be assigned variably and per machine data record. Normally the letters X, Y, Z are used to designate the 3 linear axes of a Cartesian system of spatial coordinates. A, B, C are used frequently to designate circular axes. Going beyond this, all remaining uppercase letters can be used for designating numerical axes.

2.4.3 Program examples

To illustrate the facts presented up to now, a program example is introduced in advance of the following chapters.

The following address letters are used:
Preparatory functions:
G00 "Drive in rapid traverse"
G01 "Drive on linear path with feed according to F word"
Spindle:
S1000 "Spindle speed 1000 rpm"
Feed:
F1000 "Feed 1000mm/min"
Numerical axes:
X, Y, Z "three Cartesian axes"
Machine functions:
M03 "Spindle clockwise with prog. speed"
M05 "Spindle off"
M30 "Program end"

```
% 100 (Example program)

N10 G00 X100 Y100               ( Drive in rapid traverse X and Y axis to   )
                                ( position 100, 100                         )
N20 Z100                        ( Drive in rapid traverse Z axis to 100.    )
                                ( G00                                       )
                                ( remains effective until there is          )
                                ( deselection by another G function.        )
                                (                                           )
N30 G1 Z50 F1000 S1000 M3       ( Spindle clockwise with 1000 rpm and       )
                                (                                           )
                                ( Drive linearly with feed speed            )
                                ( 1000 mm/min to position Z=50              )
                                (                                           )
N40 Z100                        ( Drive with feed speed to Z=100            )
                                (                                           )
N50 G0 X200 Y200 Z200           ( Drive in rapid traverse to position       )
                                ( X=Y=Z=200                                 )
N50 M5                          ( Spindle off                               )
N60 M30                         ( Program end                               )
```

2.5 Distance information

2.6 Axis commands

Axis designations can be configured and must be taken from the configuration-specific description.

The following are available as axis designations:

- Single address letters: {A, B, C, Q, U, V, W, X, Y, Z}
  After programming of an axis designation, which consists only of one address letter, there must be a blank after the position value before the next character in order not to cause confusion with a following allocation by the equal sign.

  Example:

X50R1=7	(ERROR)
X50 R1=7	(CORRECT)

The following conventions also apply:

- * Each axis designation must be defined in the configuration.
- * A numerical value or an expression must always follow an axis designation:

X <int, float, expr>

```
Examples:
X 100.0

Y 0.001

Z SIN [30]

A SQRT [2]/2

B 4 * R1/R2
```

The usual designations X, Y, Z for the three linear axes of a Cartesian coordinate system and A, B for two further straight-cut axes should be used in these programming instructions.

2.7 Dimensioning systems, entry and accuracy ranges

The dimensioning systems for the statements of position values, angles and speeds are agreed as follows according to user configuration:

Length and position statements:	mm or inch,
Linear speeds:	mm or inch per s or min,
Angles:	deg or ndeg,
Angular speed:	deg or ndeg per s or min.

The above dimensioning systems are the permanently configured variables as standard. Moreover the programmer is free to use self-defined dimensioning systems with the aid of the parameter calculation.

All length statements and linear speeds are calculated in the AMK CNC control system as standard with an accuracy of 0.1 µm. The maximum travel which can be entered with this resolution is then -214m .... +214m. Numerical entries in programming accordingly may not exceed the range from -214 000.0000 mm to +214 000.0000 mm. This does not apply for individual elements (e.g. parameters) of an arithmetical expression if the result of the arithmetical expression lies within the named range. It should be taken into account that this numerical range may also not be exceeded in connection with offsets and corrections.

If it is a rotary axis (round axis or spindle), then angles are treated as standard with a resolution of 0.001 deg. With a resolution of 0.001°, angles in a range of

$(-2.14 \ldots +2.14) * 10^6 / 360 = 2 * 5940$ revolutions are possible.

Apart from the standard, fixed configuration of the above entry and accuracy ranges, customer-specific configurations are possible. The entry and accuracy ranges must then be taken from the customer-specific description.

2.8 Coordinate systems

After the homing cycle, the control system is in the machine zero or in the machine coordinate system. If entries are now made from the NC program (e.g. X100), then the programmed coordinates coincide with the absolute coordinates. Example for configured axes x and y:

$$x_a = x_p$$
$$y_a = y_p$$

The zero point can be offset by a zero offset (ZPO) from the machine zero M to a freely selectable workpiece zero W or a workpiece coordinate system. The absolute coordinates result from this from the addition of ZPO and programmed coordinates:

$$x_a = x_{NPV} + x_p$$
$$y_a = y_{NPV} + y_p$$

A total of six different ZPOs which have to be selected with G54 ... G59 can be defined in the zero offset data record. G53 cancels a selected ZPO again.
Independent of these ZPOs which can be defined through the zero offset data record, a further offset can be programmed in the part program explicitly with G92 X... Y... Z... . This

datum point offset (BPV) is added to the preceding ZPO. The absolute coordinates can be determined with this as follows (Fig. 1):

$$x_a = x_{NPV} + x_{BPV} + x_p$$
$$y_a = y_{NPV} + y_{BPV} + y_p$$

x_a, y_a	Absolute coordinates
x_p, y_p	Programmed coordinates
x_{NPV}, y_{NPV}	Zero offset
x_{BPV}, y_{BPV}	Datum point offset

M:	Machine zero
W:	Workpiece zero
B:	Datum point for the coordinate statements
P:	Position

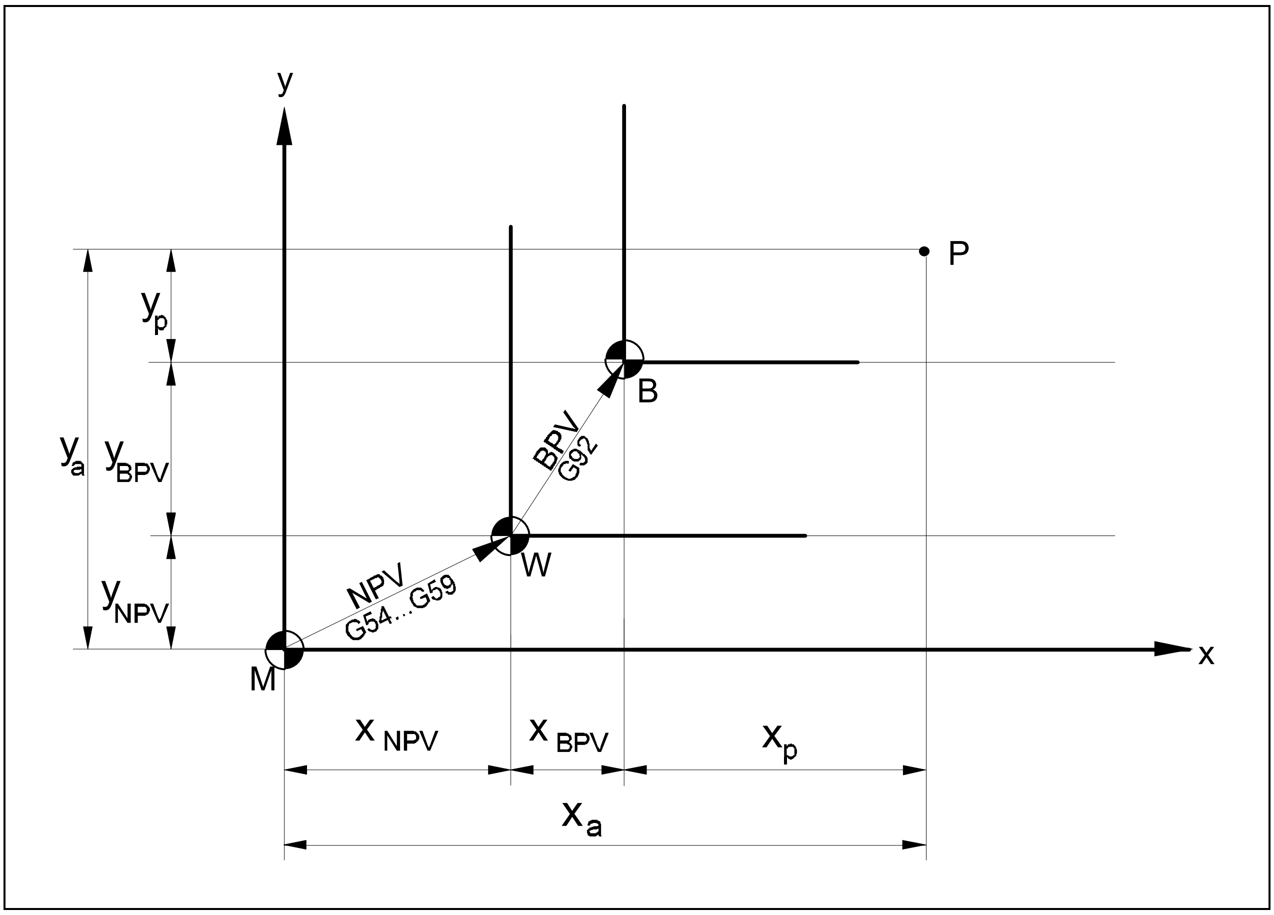


Fig. 2.8.1: Display of possible offsets (ZPO= Zero point Offset)

3 G FUNCTIONS

The so-called "G" functions describe the type of feed movement, interpolation mode, dimensioning mode, the temporal influencing and activation of certain operating states. The syntax is

```
        G <int>
```

Different distinguishing features must be taken into account with the F functions:

Effectiveness: There are G functions which are valid only for the relevant block (block effective, non-modal) and those which are valid after the first selection until they are explicitly deselected (modal).

Exclusion: Certain G functions are mutually exclusive. For instance, G01 (linear interpolation) and G02 (circular interpolation) cannot be selected simultaneously. Mutually exclusive G functions are grouped together.

G functions are classified into the following groups:

3.1 Preparatory functions,
3.2 Acceleration regulations,
3.3 Feed conventions
3.4 Plane selection,
3.6 Dimensional units,
3.7 Dimensioning systems,
3.8 Braking conditions,
3.9 Tool radius offset selection,
3.1.6 Zero point offsets (ZPO),
3.11 Center point data for circle definition,
3.13 Rounding of linear blocks

A modal function is deselected automatically if another function of the same group is selected in a following block.

```
Example:
.
N50 G01 X100 Y200       (Linear interpolation active)
N60 G41 X200 Y200       (Linear interpolation active)
N70 X300 Y250           (Linear interpolation active)
N80 X100 Y50            (Linear interpolation active)
N90 G02 X100 Y50 I100   (Circular interpolation active )
        .
        .
```

Basic position:
The control system is in the basic position on switching on, after RESET or at the program end. Some G functions are already effective without explicit selection in this basic position. The basic position is stated in the description of the individual G functions.

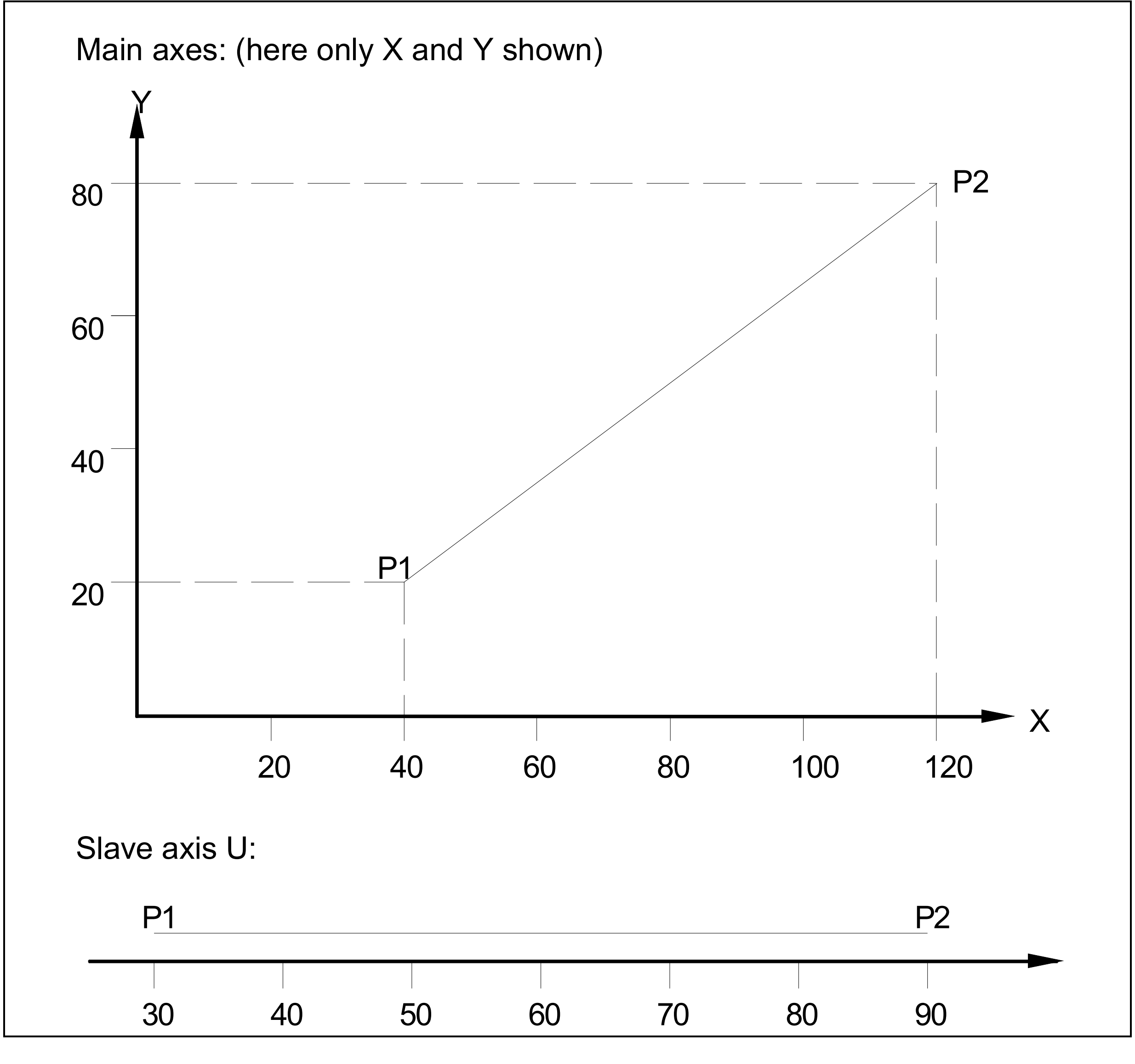
3.1 Preparatory functions

G00	Rapid traverse	(modal)
G01	Straight line	(modal, basic state)
G02	Circle clockwise cw	(modal)
G03	Circle counter-clockwise ccw	(modal)
G04	Dwell time	
G74	Homing cycle	
G92	Datum point offset	
G98	Set negative software limit switch	
G99	Set positive software limit switch	

3.1.1 Rapid traverse G00

G00	Rapid traverse	(modal)

With G00 selected, the rapid traverse speed of the axis (defined in the machine parameters) is the basis for the traversing speed. Here the speed of the axes results so that at least one axis is moved with its rapid traverse speed. Arbitrary straight lines can be programmed in the Cartesian coordinate system (X, Y, Z). All programmed slave axes are moved with linear speed so that the start and end of their movement occurs at the same time as the main axes.



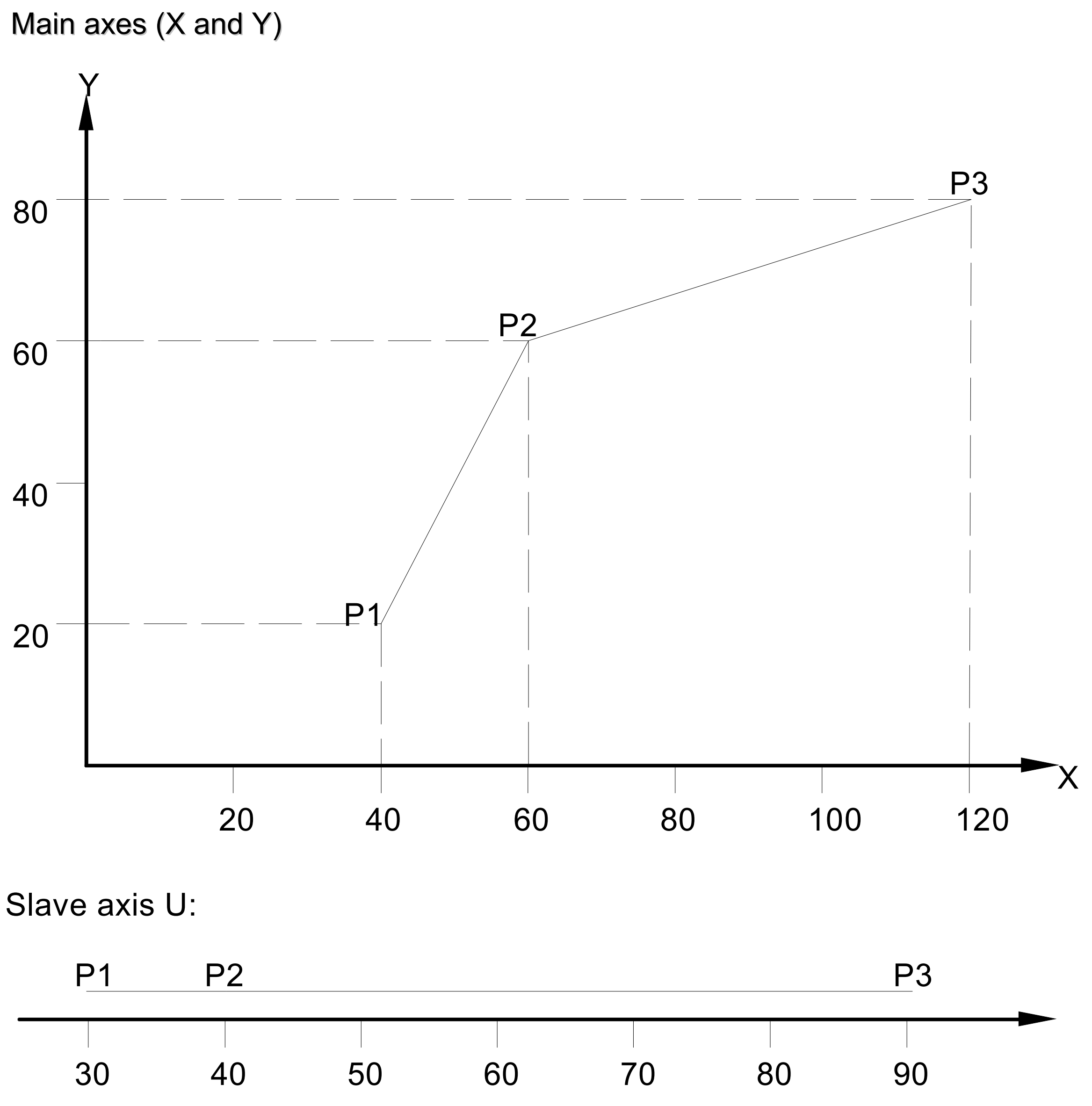
Absolute coordinate entry:	Nnnnn G00 G90 X120 Y80 U90	(drive from P1 to P2)
Incremental coordinate entry:	Nnnnn G00 G91 X80 Y60 U60	(drive from P1 to P2)

Fig. 3.1.1: Positioning in rapid traverse

3.1.2 Linear interpolation G01

G01	Linear interpolation	(modal)

With G01 selected, the programmed distance is traversed on a straight line to the target position with the feed speed stated under the F word (unit configurable, generally mm/min). Arbitrary straight lines can be programmed in the Cartesian coordinate system (X, Y, Z). All programmed slave axes are moved with linear speed so that the start and the end of their movement take place at the same time as the main axes.



Absolute coordinate entry:

```
Nnnnn   G01 G90 X60 Y60 U40 F1000     (drive from P1 to P2 Feed 1000mm/min)
Nnnnn    X120 Y80 U90                 (drive from P2 to P3 Feed 1000mm/min)
```

Incremental coordinate entry:

```
Nnnnn  G01 G90 X20 Y40 U10 F1000      (drive from P1 to P2 Feed 1000mm/min)
```

Nnnnn	X60 Y20 U50	(drive from P2 to P3 Feed 1000mm/min)

Fig. 3.1.2: Linear interpolation (G01)

Main axes (here only X and Y shown)

3.1.3 Circular interpolation G02/G03

G02	Circular interpolation cw	(modal)
G03	Circular interpolation ccw	(modal)

With G02 selected (Circle clockwise, CW) or G03 (Circle counter-clockwise, CCW), the programmed distance is traversed to the target position on a circle with the feed speed stated under the F word. Circles can be traversed in the three main planes of the coordinate system (X-Y, Z-X, Y-Z). The main plane is selected with the functions G17, G18, G19 (see also Section 4.3: Plane selection).
All programmed slave axes are moved with linear speed so that the start and end of their movement takes place at the same time as the main axes.

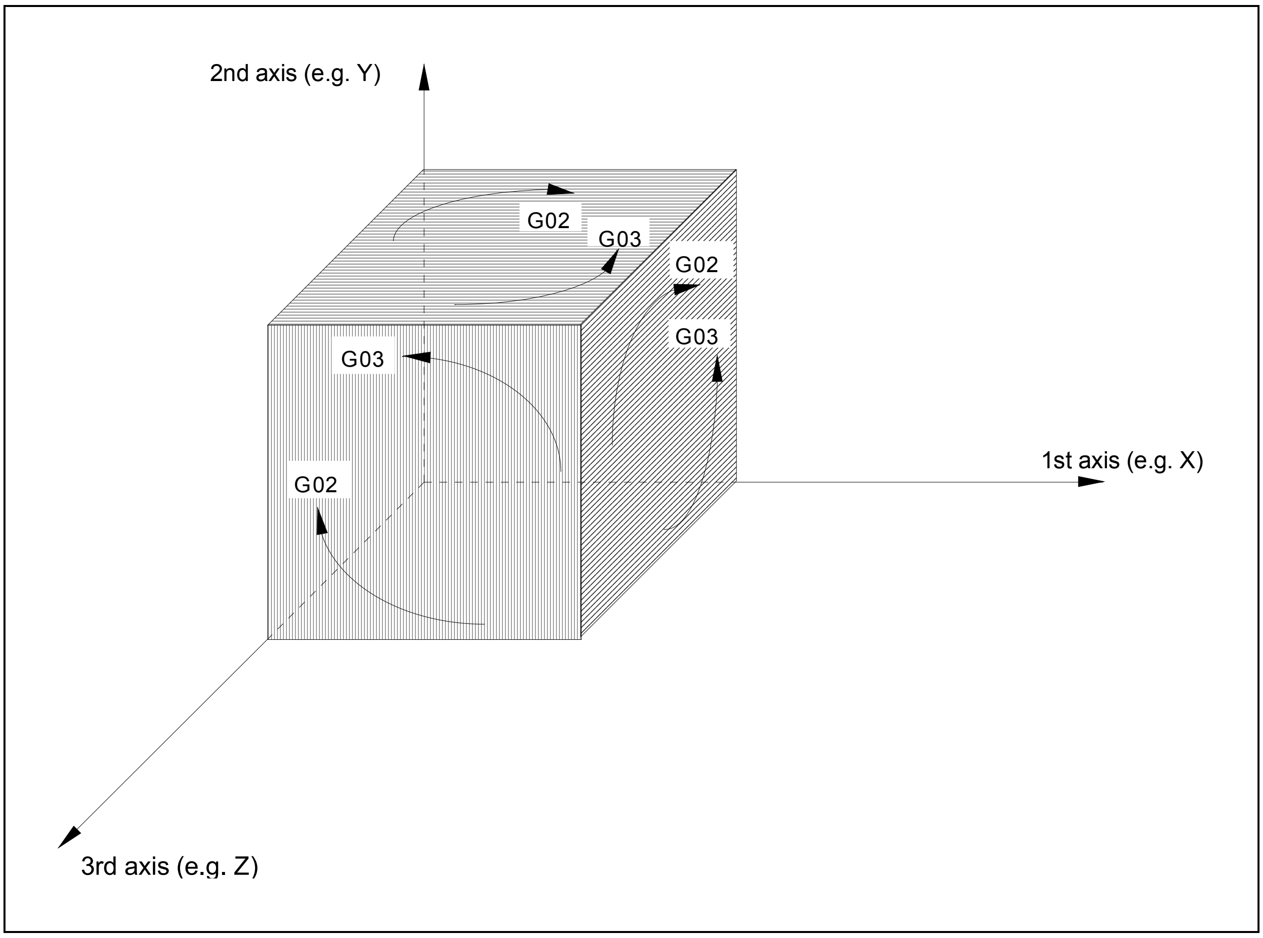


Fig. 3.1.3.1: Illustration of the circular functions G02 and G03

The circle starting point "K_a" (is defined by the preceding block), the circle end point "K_e" and the circle center point "K_m" are used to define the circle. The circle center point is stated by the interpolation parameters I, J, K relative to the circle start point with G162 effective, absolutely with G161 effective:

G162:	(Basic setting)	
	I	- relative position of Km in X direction
	J	- relative position of Km in Y direction
	K	- relative position of Km in Z direction
G161:		
	I	- absolute position of Km in X direction
	J	- absolute position of Km in Y direction
	K	- absolute position of Km in Z direction

There is an error message if the circle center point is wrongly defined. The circle center point coordinates are "not " modal.

If a full circle should be traversed, the circle start point must be stated as circle end point.

Example:

```
        N10 G01 X10 Y10
        N20 G02 X30 Y30 I10 J10         (Semi circle)
        N30 X30 Y30 I10 J10             (Full circle)
        N40 X50 Y50                     (Error message, since no center point)
                                        (or radius was stated)
```

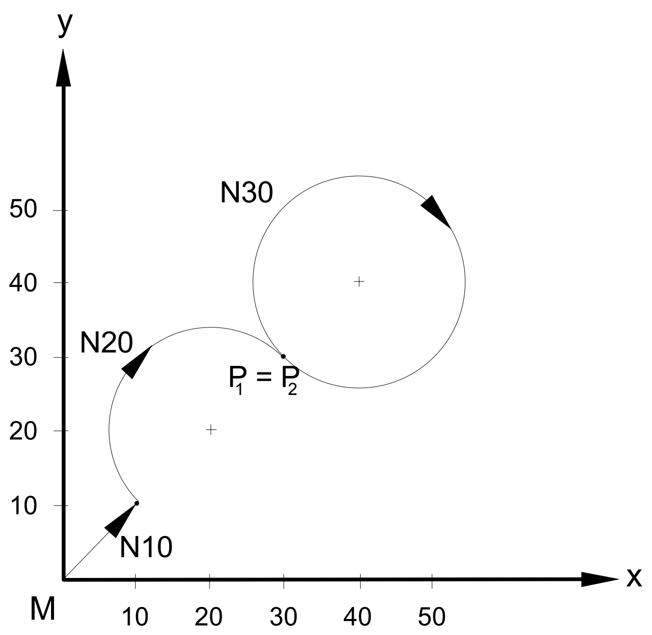
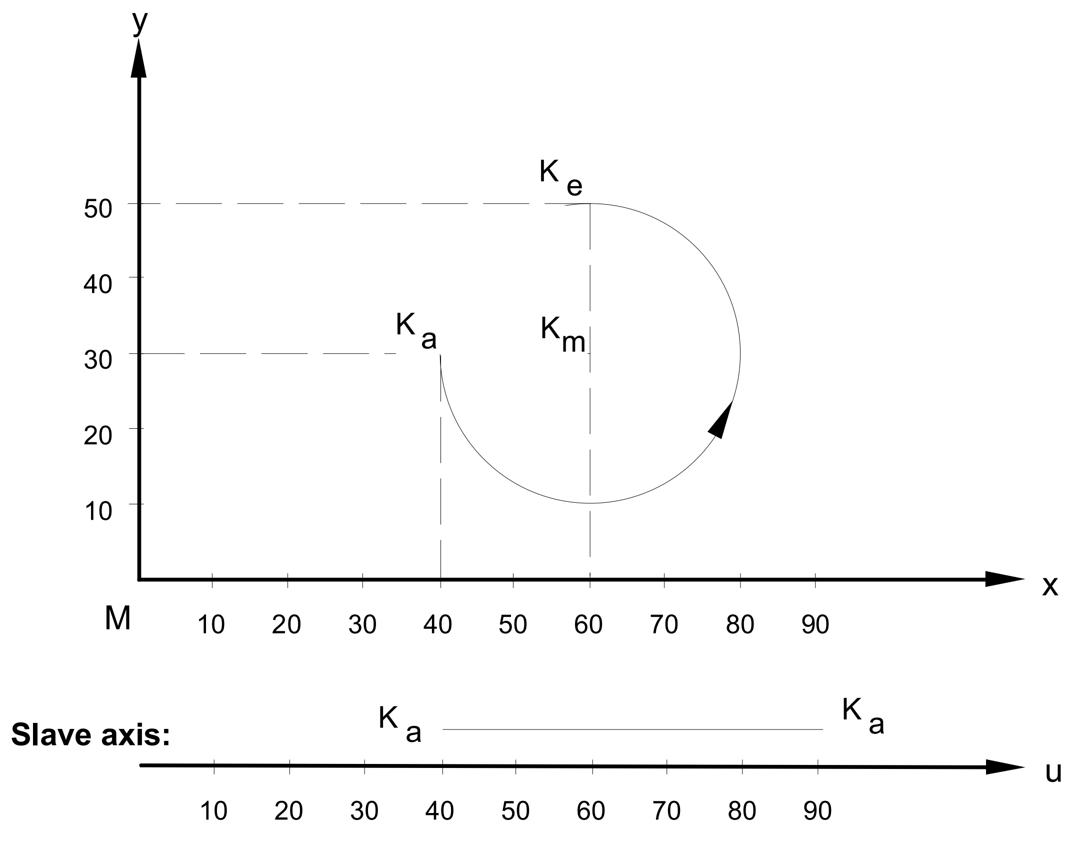


Fig. 3.1.3.2: Full circle

Main axes (here only X and Y):



```
Absolute coordinate entry:
Nnn G90 F1000                        (Absolute coordinate, feed)
Nnn G17                              (Selection of the X-Y plane)
Nnn G03 G161 X60 Y50 I60 J30 U90     (Circle: Ka -> Ke and )
                                 (Straight line: P1 -> P2    )
Incremental coordinate entry:
Nnn G91 F1000                        (Incremental coordinate, feed )
Nnn G17                              (Selection X-Y plane    )
Nnn G03 G162 X20 Y20 I20 U50         (Circle: Ka -> Ke and )
                                     (Straight line: P1 -> P2    )
```

Fig. 3.1.3.3: Example of circular interpolation

3.1.4 Dwell (G04), ($TIME)

G04	Dwell	(nonmodal)

Dwells are required for retracting or possibly in machine functions. The dwell is stated in s under the address of the 1st axis (here: X):

```
N10 G04 X 4.13           (Wait 4.13 seconds)
```

A further possibility of stating the dwell is with the extended function $TIME:

```
N10 $TIME 4.13           (Wait 4.13 seconds)
```

3.1.5 Programmable homing cycle (G74)

G74	Homing cycle	(nonmodal)

G74 allows the NC program controlled performance of a homing cycle whereby the axes to be homed and the order in which the axes should perform the homing cycle must be stated. Indices define the order of the homing cycle. When the same indices are allocated, the homing cycle takes place simultaneously.

Examples:

```
N10 G74 X3 Z1 Y4   Order of the homing cycle:  Z-X-Y

N10 G74 X3 Z3 Y3   Simultaneous homing cycle
```

The spindle is speed controlled and thus cannot be homed. Should the spindle axis be operated in closed loop position control, the associated logical position controlled axis, e.g. the C axis, must be homed.

The programming of the spindle M functions (see also Section 5.1.6: Spindle functions) is not allowed in the same NC record as G74.

Special case index "99":
No homing cycle is performed with allocation of the index "99" (e.g. G74 Z99), but the control-internal interpolators of the stated axis are homed to the current axis positions. This is performed for example instead of the homing cycle if a drive was not guided in the meantime by the NC interpolator (synchronous axis, spindle) and should be interpolated again from now on.

Special case index "98":
In contrast to homing with index "99", the addressed axes are set to their parameterizable modulo value or zeroed at the current position (if the modulo value=1) with index "98". This "Modulo value" stands in the axis data record in the distance resolution numerator parameter (see Document "NC configuration").

3.1.6 Datum point offset (G92)

G92	Datum point offset	(nonmodal)

G92 allows a programmable datum point offset in the stated axes and a freely programmable value: additive zero offset. According to the set G90/ G91, the currently programmed datum point offset is set absolutely or added to the previous one.

Note: "Nonmodal" applies only for G92; the datum point offset itself applies up to the new G92 programming.

Example:

```
N10 G90           (Absolute coordinate statement           )
N20 G92 X10 Z30   (Offsets programmed and absolute)
 .                (coordinates by 10 in X, 30 in Z.    )
 .
 .
Nnn G92 X0 Z0     (Withdrawal of the datum point offset)
```

3.1.7 Set negative software limit switch (G98)

G98 Set negative software limit switch	(nonmodal)

G98 sets the negative limit switch positions in all programmed axes. This occurs absolutely or additively to the since then software limit switch position according to set G90/ G91.

At the machine runup, these variables are preallocated with the values from the axis machine data record.

Note:
"Nonmodal" applies only for function G98. The software limit switches remain effective.

Example:

```
Sets negative software limit switches in X to -1000 and in Y to -2000
 N10 G98 X-1000 Y-2000
```

3.1.8 Set positive software limit switch (G99)

G99 Set positive software limit switch	(nonmodal)

G99 sets the positive limit switch positions in all programmed axes. This occurs absolutely or additively to the since then software limit switch position according to set G90/G91.

At the machine runup, these variables are preallocated with the values from the axis machine data record.

Note:
"Nonmodal" applies only for G99. The software limit switches remain effective.

```
Example:      Sets positive software limit switches in X to +1000 and in Y to +2000
N10 G99 X+1000 Y+2000
```

3.2 Delay at end of block (G09)

G09 Delay at end of block	(nonmodal)

If two consecutive NC blocks are programmed with different feed speeds, then there is a "soft" adaptation at the block boundary. An acceleration takes place exclusively at the block start. G09 states that a delay for the feed of the next block should take place already at the end of the current block.

Exception:
Delay at end of block acts basically on the speed of the following block in the transition from G00 to G01, G02 or G03.

Beispiel:

```
N10  G01 X500 F400
N20      X900 F1000
```

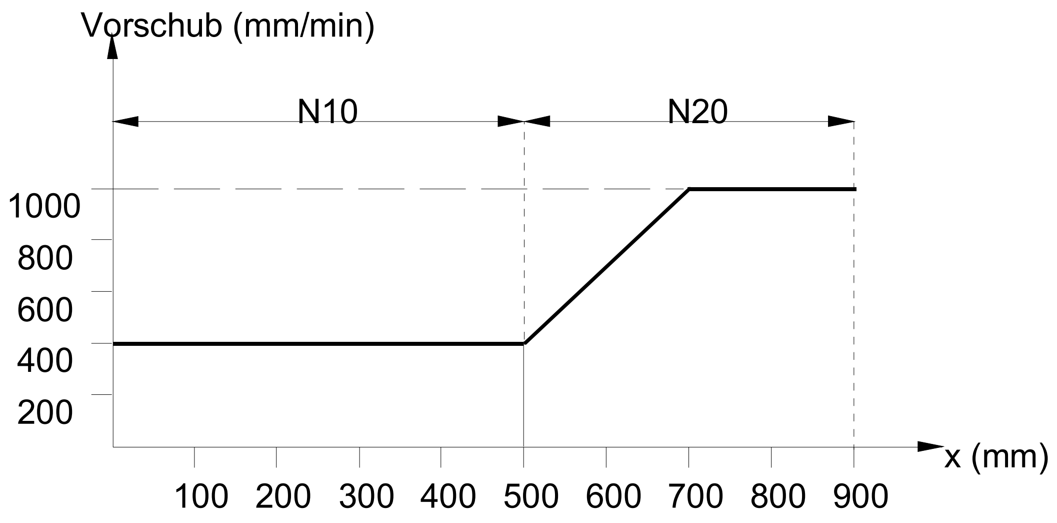
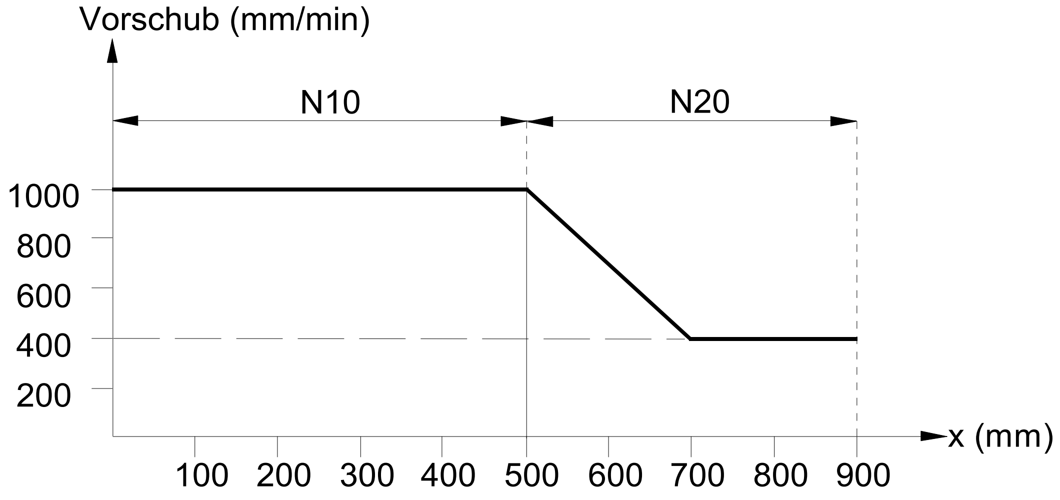


Fig. 3.2.1: Acceleration at the block transition in the basic state

Beispiel = Example Vorschub = Feed

Beispiel:

```
N10 G01 X500 F1000
N20     X900 F400
```



Beispiel:

```
N10 G01 G09 X500 F1000
N20         X900 F400
```

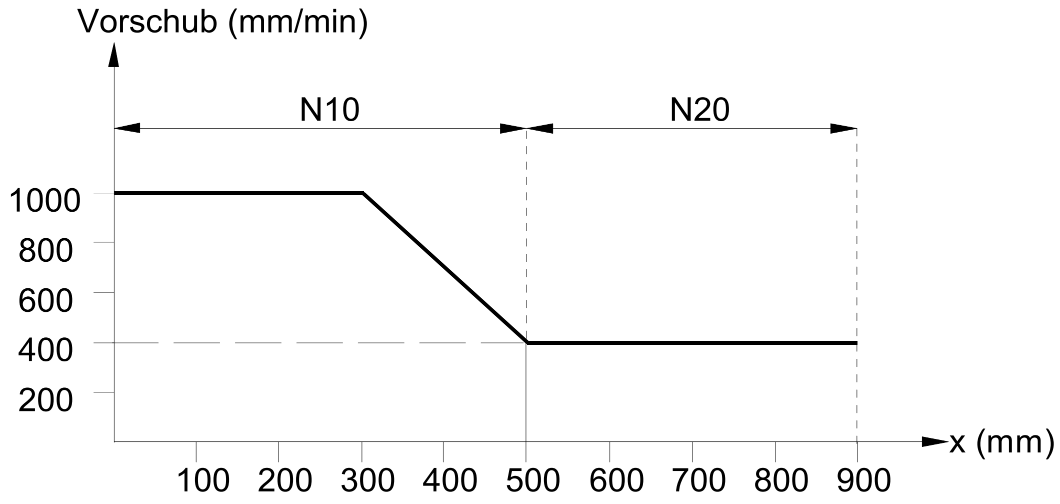


Fig. 3.2.2: Delay at the block transition without and with G09

Beispiel = Example Vorschub = Feed

3.3 Revolution feed (G94/G95)

G94	Revolution feed OFF:[mm/min]	(modal)
G95	Revolution feed ON:[mm/U]	(modal)

The feed speed of the contouring axes becomes dependent upon the spindle speed with G95. The unit is [mm/revolution]. The axis data record defines which axis (spindle) is used as master axis for determining the feed speed.

3.4 Plane selection (G17, G18, G19)

G17	X-Y plane	(modal, basic state)
G18	Z-X plane	(modal)
G19	Y-Z plane	(modal)

The plane in which the tool radius offset, the tool length offset and the circular interpolation (see also Section 4.1.3: "Circular interpolation G02/G03") should be effective is defined by programming G17, G18 or G19.

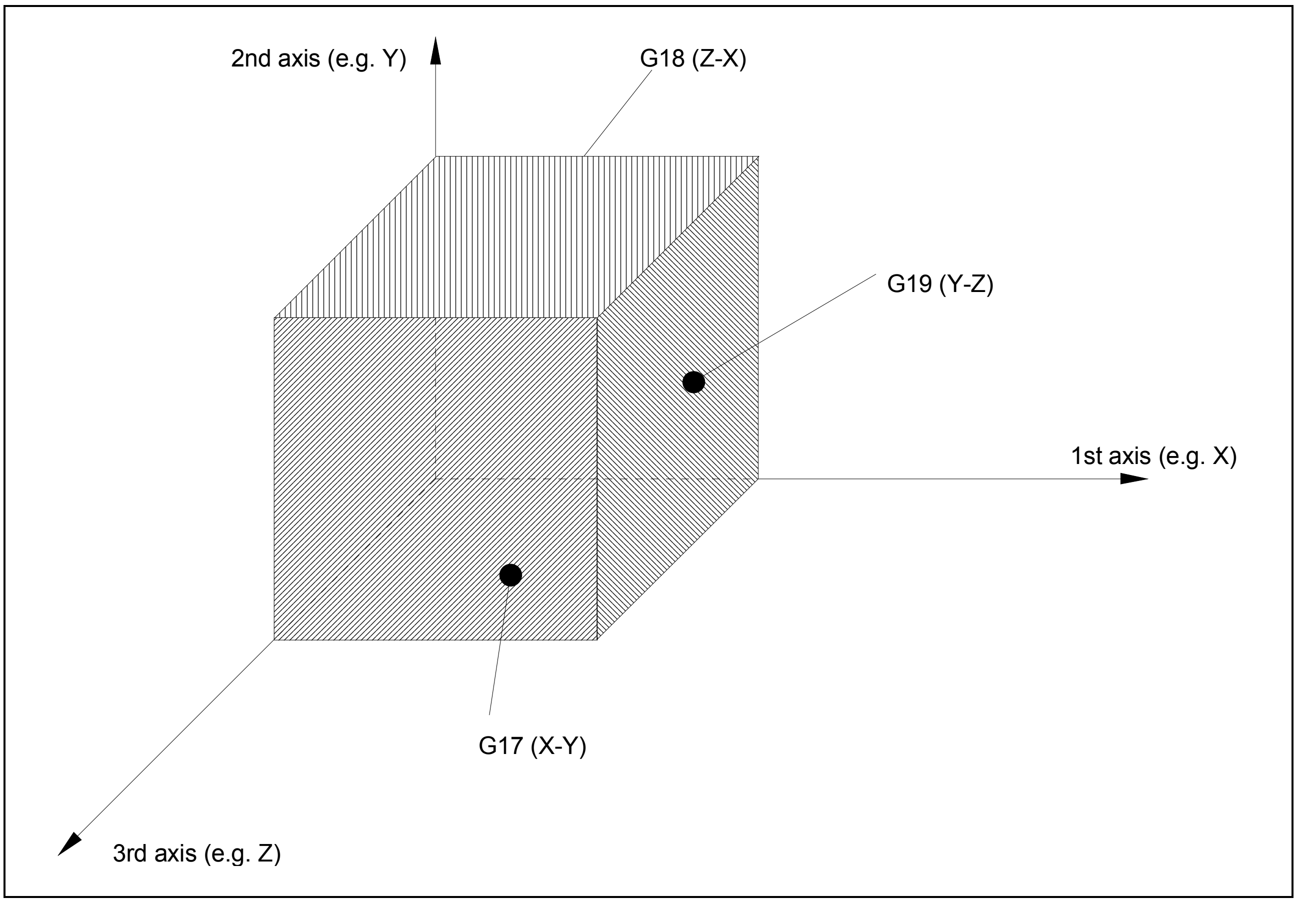


Fig. 3.4: Display of the plane selection

3.5 Mirroring (G20, G21, G22, G23)

G20	Cancel mirroring	(modal, basic state)
G21	Mirroring X on Y axis	(modal)
G22	Mirroring Y on X axis	(modal)
G23	Mirroring X on Y and Y on X	(modal)

The programmed contour is mirrored on the stated axis by programming G21, G22. Point mirroring on the zero point is obtained with G23.

3.6 Dimensional units (G70, G71)

G70	Entries in inches	(modal)
G71	Entries metric	(modal, basic state)

The statement G70 or G71 acts on all distance and coordinate data, but not on feed, tool offsets and zero offsets.

3.7 Dimensioning systems (G90/G91)

G90	Absolute coordinate	(modal, basic state)
G91	Incremental coordinate	(modal)

In the absolute data input (G90), all coordinate data refer to the coordinate zero, i.e. the numerical value of a distance block states the target position in the coordinate system.

In incremental programming, the coordinate data refer to the target position of the preceding distance block, i.e. the numerical value of a distance block states the distance to be driven.

3.8 In-position programming (G60)

G60	In-position programming	(nonmodal)

G60 enables a target position to be moved to accurately within the in-position programming limits. The feed speed is reduced to zero up to the end of the block and the following error is diminished.

In-position programming can be used if corners must be machined exactly or if the target position must be reached exactly in the case of direction reversal. The step enabling condition is reaching a control window (G60 window) which is defined in the axis machine data (see description of NC configuration).

3.9 Tool offset selection (G40, G41, G42)

G40	TRO deselection	(modal, basic state)
G41	Tool radius offset left of contour	(modal)
G42	Tool radius offset right of contour	(modal)

The tool radius offset acts in the plane selected with G17, G18 or G19. The tool length offset acts perpendicular to the selected plane. The tool offset records filed under the D words are used as tool offset values. The D word for the corresponding tool offset record must be stated (see also Chap. 12: "Tool geometry offset").

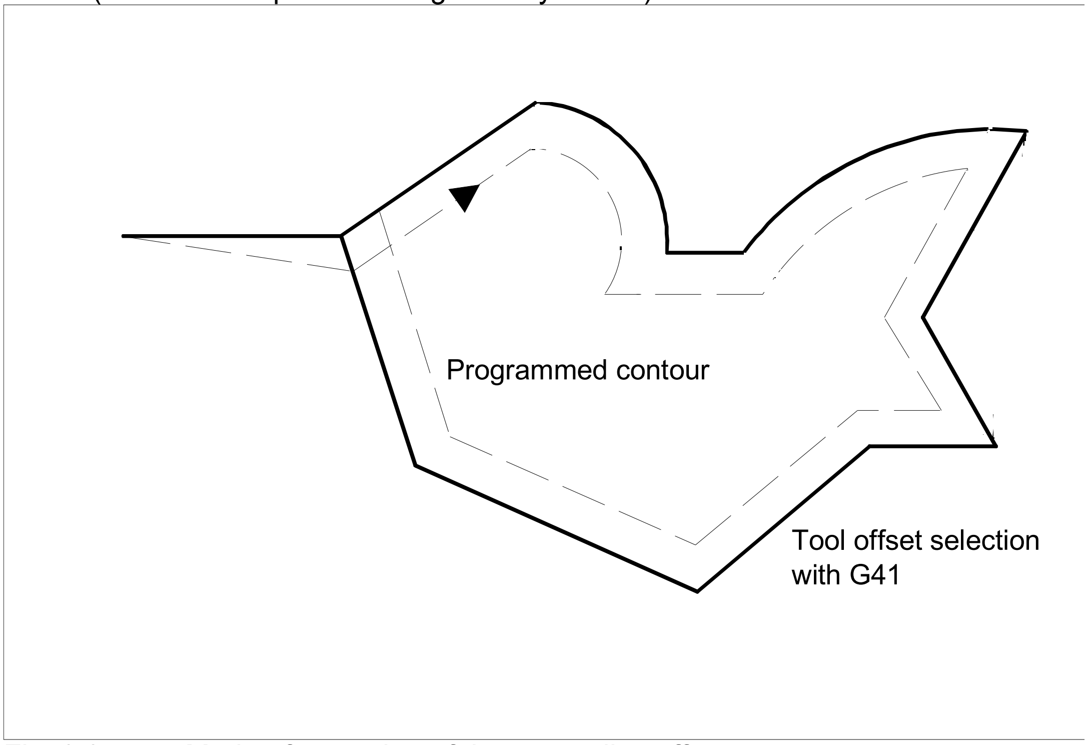


Fig. 3.9: Mode of operation of the tool radius offset

3.10 Tool radius offset (TRO) selection type (G138, G139)

The TRO selection and deselection type can be selected by G functions:

G138 -Direct selection	(modal)
G139 -Selection with transition blocks	(basic state)

Refer to Section 11.3.1 “Selection and approach blocks of the TRO“ for the mode of operation of the different selection/deselection strategies.

3.11 Center statement for the circle definition (G161, G162)

G161 Circle center absolute	(modal)
G162 Circle center relative	(modal, basic state)

With G161 effective, the center is defined relative to the circle start point via I, J, K. With G162 effective, I, J, K are absolute data in the coordinate system of the programmer (see Section 3.1.3 Circular interpolation G02/G03).

3.12 Zero point offsets ZPO (G53/G54/ ... /G59)

G53	Deselection of the ZPO	(modal, basic state)
G54	Selection of 2nd ZPO	(modal)
G55	Selection of 3rd ZPO	(modal)
G56	Selection of 4th ZPO	(modal)
G57	Selection of 5th ZPO	(modal)
G58	Selection of 6th ZPO	(modal)
G59	Selection of 7th ZPO	(modal)

With G54 ... G59, the corresponding zero offset from the zero offset data (zero 02..07) becomes valid. The zero offset is already valid in the block in which G53, G54, ... stands. However, without coordinate statement there is no movement!
No zero offset is effective in the basic position (G53).

3.13 Rounding of linear blocks (G36, G37)

G36	Rounding of linear blocks	(modal)
G37	Rounding deselection	(modal, basic state)

At the end of a NC block, there can be a short-term speed change by up to 50% of the programmed tool path speed because of the residual distance in the last NC interpolation cycle. This effect can be disturbing with very short blocks with high speed.
The "Rounding of linear blocks" function (G36) can be selected to avoid this effect. A somewhat shorter or longer travel is specified so that the speed remains constant in the last interpolation cycle. The consequence is a slight tool path error.
This function is expedient for example when describing a path by many short NC blocks.

3.14 Fix override at 100% (G63)

G63	Fix override at 100%	(nonmodal)

If the G word "G63" is stated in a motion block (G00, G01, G02, G03), then the override is fixed at 100% on execution of this block. An override input has no effect.

4 SWITCHING AND ADDITIONAL FUNCTIONS

4.1 The "M" functions

M functions are determined essentially by programming the "Programmed logic controller" (in general usage also PLC). This controller receives information on the programmed M functions in the current NC block through an interface ("Window to the PLC"). Synchronization conditions which control the synchronization between NC and PLC are provided in addition (see NC-PLC interface description document).

```
        M <int>
```

The following M functions are executed by the NC kernel itself:

M00	Programmed stop,
M01	Optional stop,
M29	Subroutine end,
M30	Program end, standstill of the machine,
M96	Acceleration override

Further preallocated M functions are:

M03	Spindle movement clockwise
M04	Spindle movement counter-clockwise
M05	Stop spindle
M19	Position spindle
M25	Read parameters from the PLC (unsynchronized)
M26	Synchronization of axis groups
M27	Synchronous control (master axis 1)
M28	Synchronous control (master axis 2)
M112	Read parameters from the PLC (synchronized)

4.1.1 Programmed stop (M00)

M00 interrupts the current NC program, in order to perform a measurement or chip removal, for instance. Machining is resumed when the "Cycle start" button is pressed. If M00 is in a block with motion statements, then the interruption takes place after execution of this motion statement.

4.1.2 Optional stop (M01)

M01 acts like M00, only the effectiveness of M01 must be activated on the control panel of the controller.

4.1.3 Program end (M30)

M30 causes in the main program resetting to the main program start and return of the NC into the basic position.
In addition the "Program started" signal in the interface to the machine interface (PLC) is reset with M30.

4.1.4 Subroutine end (M29)

M29 is the correct conclusion both of a local and of a global subroutine.

4.1.5 Axis group synchronization (M26)

The M26 command serves to synchronize different axis groups to one another. For instance if a NC block contains a movement for a path **and** a straight line, then both axis groups are started simultaneously. If the path is finished before the straight line has reached its programmed end position, then the next path block is executed immediately. If this should be prevented, because the next path block may be started only after the straight line end position is reached, then a M26 must be programmed in the next NC block.

4.1.6 Acceleration override (M96)

The acceleration parameterized in the axis data record is changed by a percentage with M96 in combination with a S word. The value in the S word corresponds to the percentage of the parameterized acceleration. The value range goes from 1..1000%.

4.1.7 Spindle M functions

M-Fct.	Meaning	Remarks
M03	Spindle movement clockwise	Spindle 1
M04	Spindle movement counter-clockwise	
M05	Stop spindle	
M17	Set speed limit for Vconst	
M18	Constant cutting speed	
M19	Position spindle in rapid traverse	
M13	Spindle movement clockwise	Spindle 2, if configured
M14	Spindle movement counter-clockwise	
M15	Stop spindle	
M20	Set speed limit for Vconst	
M21	Constant cutting speed	
M22	Position spindle in rapid traverse	

M03, M04, M17, M18 (M13, M14, M20, M21) must be stated in combination with a S word (figures in brackets for 2nd spindle). If no S word is stated, then the last stated S word is taken over. A S word alone produces no spindle movement.

M05 (M15) stops the spindle. The statement is made without S word.

M17 (M20) sets the speed limit for the constant cutting speed to the speed stated in the S word. This needs to be set only once in the program.

M18 (M21) selects the "Constant cutting speed (Vconst)" mode. The value of the cutting speed in m/min stands in the S word. The spindle speed then changes depending upon the position of the reference axis (see configuration of axis data records keyword "Master axis"). If the speed limit is 0, then there is an error message.

M19 (M22) positions the spindle to position 0. The statement is made without S word. An error message is output if M19(M22) was programmed and the spindle position is not known.

```
Example:
N50 S750 M03            (New spindle speed 750)
N60 M19                 (Spindle positioning to position 0)
N70 M03                 (Spindle rotation cw with speed 750)
N80 M05                 (Spindle stop)
N10 S1000               (Spindle speed 1000 rpm, still no movement)
N20 M03                 (Spindle rotation cw with speed 1000 rpm)
N30 M04                 (Spindle rotation ccw with speed 1000)
N40 M05                 (Spindle stop)
N99 M30                 (Program end)
```

If a position axis is also assigned to the spindle axis, the position axis must be homed (G74) after spindle functions and before closed loop position controlled movements. If there should merely be setting down on the current position, then this can be achieved by the special function "G74 ?99". Insert the designation of the position axis for "?". This position axis is set on its modulo value.

```
Example: Spindle axis S, assigned position axis C
N10 M03 S1000           (Spindle axis 1000 rpm)
N20 G04 X2              (2s waiting time)
N30 M05                 (Spindle stop)
N40 G74 C99             (Setting on position)
N50 C500 F1000          (C axis on position 500)
N99 M30
```

4.1.8 Synchronous control, Thread function (M27, M28)

Activating the synchronous control for a slave axis. Assignment to master axes is by the configuration. The system processes 2 such assignments:

M-Fct.	Meaning	Remarks
M27	Synchronous axis follows master 1	e.g. Master1=Spindle 1
M28	Synchronous axis follows master 2	e.g. Master2=Spindle 2

The main application of synchronous control is **thread** drilling or turning.

```
M28 S<int>          Synchronous control to master axis 2        (modal)
```

For a slave axis, the synchronous control is activated according to a configured master axis 2. The synchronous ratio is specified with the S word.

The meaning of the synchronous ratio specified in the S word depends upon configuration. Generally it is stated as follows:

Feed of the slave axis in µm per revolution of the master axis.

The synchronous axis is switched back into the contouring mode by homing the assigned closed loop position controlled axis (G74). If the homing cycle is not wanted, this can be suppressed by entering an index "99". The interpolator is set to the current position of the axis.

Example:

```
%1999 (Thread drilling)
N10  G01 Z-50 C0 F5000       (Z to -50mm                              )
N20  M28 S1000               (Configuration: Master = C axis          )
                             (Slave axis identical with Z axis      )
                             (Master 3600°=10U -> Slave 10.000mm   )
N30  C3600 F10000 G60        (Slave to -40mm with in-position programming    )
N40  C0 G60                  (Turn back master
N50  G74 Z99                 (Synchronous control inactive                   )
N60  G01 C0 Z-10             (C to 0°, Z to -10                    )
N999 M30
```

4.2 Read R parameters (from the PLC)

4.2.1 Parameter transfer without synchronisation (M25)

This M function enables both the complete parameter field of the NC to be overwritten by the contents of a data block of the PLC or only one selected parameter. Execution occurs directly so that the NC program execution is continued and it is possible to work with the new parameters.

Syntax:	M25 S-1	Immediate acceptance of all transfer parameters of the PLC in the R parameter field of the NC of numbers R1000, R1001,..
	M25 Sn	Immediate acceptance of the transfer parameter n in R parameter R1000+n of the NC.

4.2.2 Parameter transfer with synchronisation (M112)

This M function causes the NC block decoder to be stopped in order to transfer parameters from the PLC for further work. This is necessary if, for instance, a program branch has to be performed because of information by the PLC.

Syntax:	M112 S-1	Waiting for acknowledgement of the PLC and subsequent acceptance of all transfer parameters of the PLC in the R parameter field of the NC as from R1000, R1001,..
	M112 Sn	Waiting for acknowledgement of the PLC and subsequent acceptance of the transfer parameter n in R parameter R1000+n of the NC.

4.3 The "H" functions

H <int>	(stopping up to the next H function)

Messages to the machine interface (PLC) can be programmed using H functions. The H function programmed in the current NC block is entered in the window to the PLC (see NC-PLC interface document). If a H function is programmed again in a later NC block, then the previously entered H function is written over.

4.4 S word

S<int, float, expr>	(modal)

In the M function M03 (M13) and M04 (M14), the value following the S word is interpreted as spindle speed. Values can be assigned directly or by parameter to the S word, whereby decimal numbers (REAL64 format) are permissible.

The spindle movement itself is initiated exclusively by M03 (M13) (spindle movement clockwise) or M04 (M14) (spindle movement counter-clockwise). The S word stated in the NC block is evaluated. The statement of only one S word produces no movement.

In the functions M17(M20) and M18(M21), the speed limit or the cutting speed stands in the S word.

In the synchronous control M27 (M28), the contents of the S word are evaluated as synchronous ratio.

The S word is provided converted as integer in the "Window to the PLC". Thus the S word in connection with an arbitrary M function can be used as pure value transfer to the PLC.

4.5 Tool location selection (T word)

T<int> Tool location selection	(modal)

The tool command determines the tool required for the machining section. The tool number is forwarded to the machine interface (PLC), which then provides the selected tool for a tool change in a tool magazine. It should be noted that the T word itself has no influence on the control system-internal calculation of the tool geometry. The D word is used for this purpose.
The T word still does not initiate a tool change. This is done normally with the machine function M06 which must be configured for the machine interface with synchronisation.

5 SPEEDS (F-)

F <int, float, expr>	Feed speed at start of block	(modal)

In the interpolation modes G01, G02, G03, the programmed path of the linear axes is moved through with the path speed agreed in the F word. The statement is made generally in mm/min, but can also be in inch/min by different configuration. The slave axes controlled in the same NC block move with a speed such that they reach their end point at the same time as the path axes.

If only one rotary axis is stated in a NC block with a G01 function, then the entry is in °/min. Here as well another configuration is possible.

Values can be assigned directly or by parameter to the F word, whereby decimal numbers are also permissible.

6 THE "N" FUNCTION

```
N <int, float, expr>
```

The NC block number can be programmed with the address letter "N". This number is entered with corresponding configuration both in the display data and in the error messages (alternatively to this, the offset value can also be used in the file).

The block number is of no importance for the program run. Therefore it also does not have to be used in ascending form.

7 SUBROUTINE TECHNIQUES

The same motion sequences and functional procedures which have to be repeated several times can be implemented as subroutines. A distinction is made between two types of subroutine:

- local subroutines,
- global subroutines.

Several subroutines (local or global) can be called up nested in one another, whereby the maximum nesting depth is 10.

Variables are transferred from the main program to the subroutine through parameters.

7.1 Local subroutines (call LL <int>)

A local subroutine is called from the main program with

```
"LL<int>
```

Local subroutines stand together with the main program in a common data file, whereby firstly all subroutines must be listed before the actual main program.

It should be noted that local subroutines can be called up in the same data file only by the main program.

Local subroutines start with "%L" and a program number. The end of the subroutine is marked by the function M29.

If these program end identifications are missing, then "%" (first character of the following program) ends the subroutine.

A subroutine should be called in a separate NC record.

Example:
Structure of a date file consisting of NC main program and local subroutines:

```
%L007                                   (1st local subroutine)
N1 .....
N2 .....
.
N9 M29                                  (M29 can also be dispensed with)


%L008                                   (2nd local subroutine)
N11 .....
N12 .....
    .
    .
N19 M29                                 (M29 can also be dispensed with)


%100                                    (Main program)
N100 .....
N105 .....
N110 LL008                              (Call of the 2nd local subroutine)
    .
    .
N200 LL007                              (Call of the 1st local subroutine)
    .
N250 LL008                              (Call of the 2nd local subroutine)
    .
    .
N300 M30                                (Main program end)
```

7.2 Global subroutines (call L <int>)

A global subroutine is called with

```
L<int>.
```

Global subroutines stand in a separate data file as independent program units. A global subroutine is not called through the subroutine name, but through the designation of this data file, which in turn can consist of local subroutines and a main program. Stating a name of this main program is required only to indicate the start of the main program after local subroutines. It can therefore be dispensed with if no local subroutines are contained in the data file.

The calling main program is also filed on another data file as independent program unit. Global subroutines can be called from all main programs.

Example: Call of local and global subroutines

```
                              (Main program with local subroutine)
%L007                                 (Local subroutine)
N11 .....
N12 .....
   .
   .
N19 M29                       (M29 can also be dispensed with)


%333  (Main program)
N100 .....
N105 .....
N110 LL007                    (Local call of subroutine %L007)
   .
N200 L100                     (Global call of subroutine %100)
   .
N300 M30
```

```
                              (Global subroutine, File name "100")
%100                          (Line can be dispensed with)
N1 .....
N2 .....
   .
   .
N9 M29                                (M29 must stand here)
```

In the above example the name of the global subroutine can be dispensed with, since the global subroutine is called through the name of the data file and there are no local subroutines in this data file.

8 PARAMETERS

Parameters can be used in the NC programs as wild cards for numerical values. The advantage of parameters is that the values of a parameter can be changed during the program run. This enables flexible NC programs to be produced.
Parameters can be transferred to local and global subroutines.
The R word is used for parameter designation. Apart from these direct parameters, indirect parameters which are designated by the P word also exist.

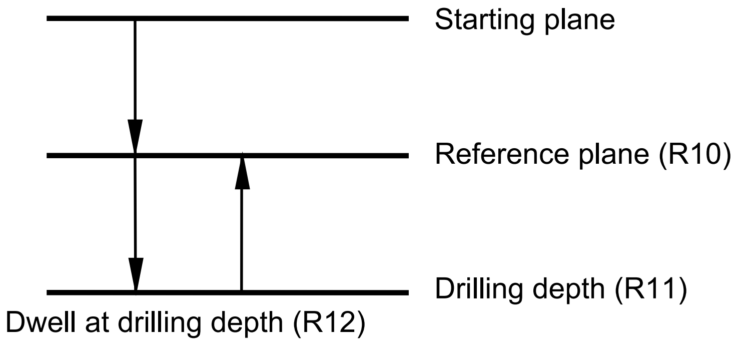
Example:
In a subroutine, e.g. a drilling cycle, parameters are used instead of co-ordinate values (drilling depth, drilling feed, dwell etc.). The parameters are then allocated the final values in the calling main program in each case:

For the global subroutine

%4712 (drilling, end facing)

the following parameters must be defined:

R10	-Reference plane = withdrawal plane
R11	-Drilling depth
R12	-Dwell



The call in the main program then looks as follows:

```
      .
N100 R10=20.5 R11=12.6 R12=1.2
N110 L4712
      .
      .
```

Fig. 8: Application example for parameter calculation

8.1 Value allocation of parameters

Parameters are allocated their values by the NC program, e.g. R12=0.12.

8.2 Parameter calculation

Instead of the direct allocation of numbers, linked arithmetic expressions can also be used (see also Section 2.4.1: Mathematical expressions). On entry the known mathematical rules apply such as

- Dot before dash calculation,
- The brackets rule, whereby however square brackets "[ ]" must be used.

8.3 Allocation of parameters to address letters

The syntax for the allocation of parameters to the axis designations is:

```
        XR1
```

However, mathematical expression can also stand instead of P1

```
        X R1*SIN [R2*30]
```

9 INFLUENCING THE NC PROGRAM RUN

The syntax for control block statements is:

```
        $<Statement>
```

It should be noted that no blanks are allowed between $ and <statement>.

Statements for influencing the program run (control blocks) permit the implementation of:

- Conditional jumps, e.g. to trigger optional machining steps depending upon a parameter (refer to the "NC-PLC description" document for parameter transfer from the PLC).
- Counter loops, to program simply machining steps repeating several times, e.g. in picture frame milling or in drilling bolt hole circles.
- Loops with run condition, to let machining steps repeating several times run on until an abort condition is fulfilled. For instance, infeeding the tool and a roughing process should be performed until a certain coordinate value is reached. Loops can be programmed as endless loop in the case of missing or not fulfilled run condition.

Caution:
If a NC block contains a control block statement, it may contain no further statements, also no further control block statements. If further statements are made except for comments, then this can lead to error messages. Only the block number and/or "/" may stand before a control block.

9.1 Conditional jumps

9.1.1 The IF-ELSE branch

The following control statements are required for the IF-ELSE branch:
$IF, $ELSE, $ELSIF, $ENDIF.

The branch always starts with

```
        $IF <expr>
```

and always ends with

```
        $ENDIF
```

The control statements

```
        $ELSE
```

and

```
        $ELSEIF<expr>
```

are optional and serve for building up multiple branches.

The jump condition in the $IF control block is tested by testing the mathematical expression for "true" or "false". However, in order to be able to use decimal variables as well, the decoder proceeds as follows:

The jump condition is fulfilled if

the absolute value of the mathematical expression is > or = 0.5.

Examples:

```
N10 ...
N20 $IF R1          Only if |R1| is greater than or equal to 0.5 are the statements
N30 ...             N30 to N50 executed.
N40
N50
N60 $ENDIF
```

But the following is also possible:

```
N10 ...
N20 $IF R1 >= 0.5   Only if R1 is greater than or equal to 0.5 are the statements
N30 ...             N30 to N50 executed.
N40
N50
N60 $ENDIF
```

or else:

```
N10 ...
N20 $IF R1 > R2     Only if R1 is greater than R2 are the statements N30 to N50
N30 ...             executed, otherwise the statements N70 to N90.
N40 ...
N50 ...
N60 $ELSE
N70 ...
N80 ...
N90 ...
N100 $ENDIF
```

For the use of ELSEIF:

```
N10 ...
N20 $IF R1 == 0     Only if R1 is equal to 0 are the statements N30
N30 ...             to N50 executed, otherwise it is checked in the $ELSEIF
N40                 condition whether R2 is greater than or equal to 0.5 and
N50                 correspondingly N70 to N90 or N110 to N130 are executed.
N60 $ELSEIF R2>=0.5      The $ELSEIF condition serves for building up
N70 ...                  branches nested in one another.
N80
N90
N100 $ELSE
N110 ...
N120
N130
N140 $ENDIF
```

Note:

1.) Corresponding to the programming language C, here as well there exists the difference in the syntax between the allocation: R5 = 3
and the comparison: $IF R5 == 3.

2.) Because in mathematical expressions the sequence
Operator ➔ Term ➔ Operator ➔ etc.
is expected, a bracket must occur firstly behind a comparison operator in negative expressions.

Example:

```
$IF R1 >= -5  is wrong                  $IF R1 >= [-5]  is right
```

9.1.2 The SWITCH statement

The SWITCH statement enables different NC program versions to be executed depending upon an arithmetical expression.

The control statements

$SWITCH, $CASE, $BREAK, $DEFAULT, $ENDSWITCH

are used for the jump distributor.

The SWITCH statement always starts with

```
$SWITCH <expr1>
```

followed by several

```
$CASE <expr2>
...
$BREAK
```

optionally followed by:

```
$DEFAULT
```

and always ends with:

```
$ENDSWITCH
```

```
Example:
N100 $SWITCH INT [R1*R2/R3]  If the result of the arithmetical expression is
N110 $CASE 1                 equal to 1, then the blocks are executed
N120 ...                     according to $CASE 1 (N120 to N140).
N130
N140 $BREAK
N150 $CASE R2                If the result is equal to R2, then the
N160                         blocks N160 ..N170 are executed.
N170 $BREAK
N300 $CASE 5
N320 ...
N330 $BREAK
N350 $DEFAULT                The $DEFAULT block is optional and serves to
N360 ...                     execute the NC blocks N360 to N380 if the
N370                         result of the $SWITCH block has corresponded
N380                         to none of the $CASE cases.
N390 $ENDSWITCH
```

Note: The comparison of the expression <expr1> with <expr2> is performed with internal REAL notation. Here the two expression are assessed to be equal if the absolute difference is < 0.00.

9.2 Counting loops

Counting loops enable statements to be executed n times. The number of loop passages is controlled by a counting variable.

The syntax of the counting loop starts with:

```
        $FOR Rn = <expr1>, <expr2>, <expr3>
```

and always ends with:

```
        $ENDFOR
```

Here Rn is the counting variable, the start value of which is defined by <expr1>, the end value of which by <expr2> and the counting increment by <expr3>.

```
Example:
N100 $FOR R1= 10, 100, 2     R1 is preallocated with 10 at the loop start. The
                             counting loop is run through until R1 has the value 100
N110 X SIN [R1 * 5]          (or more), whereby R1 is increased by 2 at the
                             end of each loop passage. The NC blocks
N120 Y COS [R1 * 5]          N110 to N140 are executed within the counting loop

N130 ...                     .
N140 ...                     .
N150 $ENDFOR
```

9.3 Loops with run condition

9.3.1 Checking the run condition at the loop start

The syntax of this loop starts with:

```
        $WHILE <expr>
```

and always ends with

```
        $ENDWHILE
```

At the start of each loop passage, the listed parameter is checked. The loop is interrupted if the expression <expr> becomes logically "false" or assumes the value range of FALSE (-0.5 < expr < 0.5).

```
Example:
N90    R1 = 100.0
N100 $WHILE R1 > 0.5           R1 > 0.5 is checked for FALSE at the loop start.
                               The loop is run through until R1 fulfils the
                               break off condition.
N110 R1 = R1 - 1.5 Y R1

N120 $ENDWHILE
N130 ...
```

9.3.2 Checking the run condition at the loop end

The syntax of this loop starts with:

```
        $DO
```

and always ends with

```
        $ENDDO <expr>
```

At the end of each loop passage, the listed parameter is checked. The loop is interrupted if the expression <expr> becomes logically "false" or assumes the value range of FALSE (-0.5 < expr < 0.5).

```
Example:
N90    R1 = 100.0
N100  $DO                  R1 is checked for FALSE at loop end. The loop
N110  R1 = R1 - .6 Y R1    is run through until R 1 fulfils the break off
N120  $ENDDO  R1 >= R3     condition.
N130 ...
```

9.4 Influencing loop processes

9.4.1 The $BREAK statement

```
        $BREAK
```

It is not always expedient to leave a loop with the break off criterion. The keyword $BREAK can also abruptly terminate execution of a loop next to the program execution with the individual $CASE marks of the $SWITCH statement (see also Section 10.2: "The SWITCH statement").
This is expedient for example with highly nested loops, if execution of the innermost loop should be interrupted.

```
Example:
N10   $WHILE <expr1>                  The loop is ended if
N20   ...                             expr1 is   "not valid"       or
N40   $IF <expr2>                     expr2      "valid".
N50      $BREAK
N60   $ENDIF
N70   ...
N90   $ENDWHILE
N100  ...
```

9.4.2 The $CONTINUE statement

```
        $CONTINUE
```

In contrast to $BREAK., the loop is not interrupted with the $CONTINUE statement, but branched to the loop start. Execution of all statements standing after $CONTINUE therefore does not take place.

```
Example:
N10   $FOR <expr1>              The statements of the lines N70 and N80
N20   ...                       are executed only if
N40   $IF <expr2>               expr2 is        "not valid"
N50      $CONTINUE
N60   $ENDIF
N70   ...
N90   $ENDFOR
N100  ...
```

10 SPECIAL FUNCTIONS

The syntax for special functions is: $<statement>.
Note here that no blanks are allowed between $ and <statement>.

10.1 Waiting function

$TIME <expr>	(non modal)

The time must be stated decimal in seconds as value. Mode of operation as for G04.
Please refer to Section 3.1.4: Dwell

11 TOOL GEOMETRY OFFSET

Tool geometries are corrected in length, radius and axial offset. The corresponding tool geometry offset data must be made available as data structure.

Tool offset data are selected with the D word:

D <int>	Tool offset selection	(modal)

The tool offset data are calculated for each axis only if a travel has been programmed with this axis.

If a tool change is therefore selected without travel, then no axial offset becomes effective.

11.1 Format of the tool geometry offset data

The D word defines a tool data record containing the following values
- Tool length (perpendicular to the main plane)
- Tool radius (in the main plane)
- Axial offset coordinates (in all axis directions)

In the next motion block, the tool length becomes effective in the perpendicular to the main plane and the axial offset coordinates in the axis directions. The tool radius is transferred to the tool radius offset and here influences the offset path calculation.

Special case:

D00	Tool offset deselection

11.2 Tool length offset (TLO)

The tool length offset is selected only for an effective motion block (i.e. for G00, G01, G02, G03) and if a tool offset value has been selected with Dnn or automatically with Tnn. The tool length offset is deselected with D00 (basic position). With TLO selected, the controller performs with the first motion block the compensating movement for taking account of the tool length with the correct sign perpendicular to the plane selected with G17, G18 or G19.

In the following example, the tool length offset is performed in the Z direction on the basis of the X-Y plane selected with G17. On selection of the offset data record D16 in block N30, the compensating movement takes place in the Z direction jointly with the motion block in N30, whereby the tool length is calculated taking account of the sign, i.e. "negative" tool lengths can also be input.

Example:

```
N10  G01  F900  G17         (X-Y plane; length offset in Z direction      )
N20  X150  Y10  Z10
N30  D16  Y40  Z15          (Selection of the length offset D16.)
 .       (The compensating movement is performed.       )
 .
 .
N100  D20     (Selection of the length offset D20.)
 .       (The compensating movement takes place          )
 .       (only with the next motion block in Z direction.    )
 .
N200  G0  D0  X0  Y0  Z0   (Deselection of TLO )
```

Offset data record	D0 :	Length = 0	Radius = 0
	.		
	.		
	D16 :	Length = 5	Radius = 5
	.		
	.		
	D20 :	Length = 12.5	Radius = 5

Compensating movement in Z direction with N30

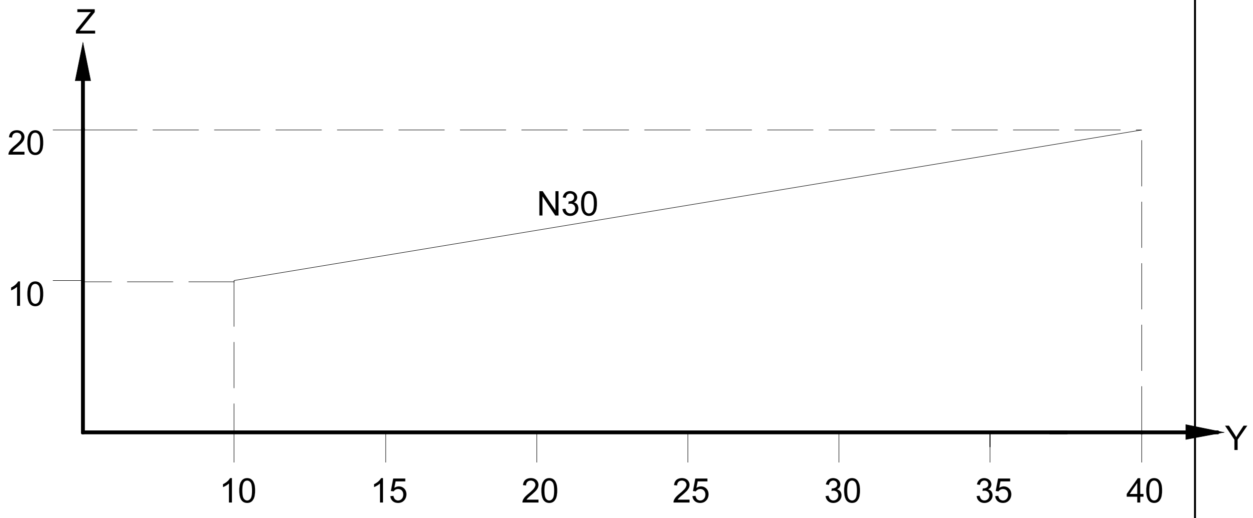


Fig. 12.1: Example of tool length offset

11.3 Tool radius offset (TRO)

With selected TRO (G41, G42), an offset tool path is calculated for a programmed workpiece contour at the "Tool radius" distance. For the calculation of intermediate blocks, which are necessary at contour transitions between the programmed NC blocks, the next NC block to the just current block is used in each case.

The following contour transitions are possible in this case:

Straight line	---	Straight line
Straight line	---	Circle
Circle	---	Straight line
Circle	---	Circle

Intermediate blocks, which are necessary for the calculation of the tool mid point path at contour transitions, are executed as standard as straight line pieces.

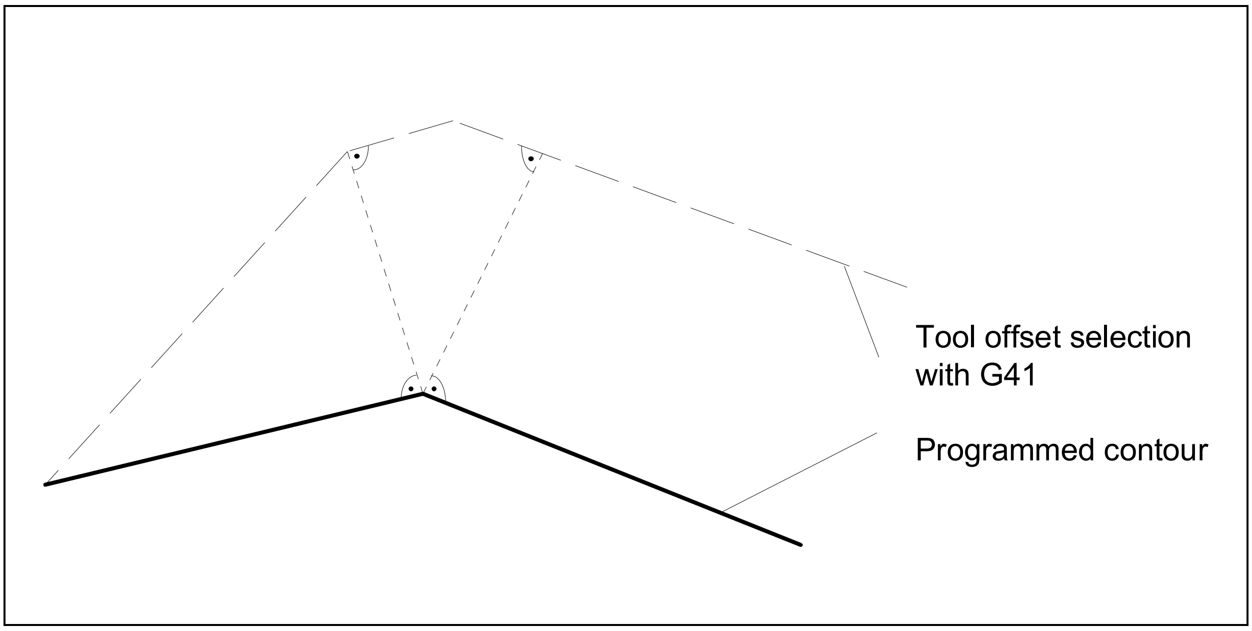


Fig. 12.2: Example for the insertion of straight intermediate blocks for a Straight line-Straight line contour transition.

In contrast to the TLO, the selection and deselection of the radius offset as well as the change of the offset direction (offset to the left or to the right of the contour viewed in machining direction) is stated explicitly with the command G40, G41, G42.

The tool radius of the currently valid D word is used for the tool radius offset.

Should the value of the tool radius be changed during a NC program run with TRO switched on, then selection of a new D word is sufficient; a renewed G41 or G42 command is not required.

On deselection of the TRO (G40) ensure that a travel takes place in the deselection block.

```
Example:
N30   G0    D0    X0    Y0    Z0    G17   (X-Y plane                  )
N40   G0    D10   X10   Y10               (Selection of the TLO       )
N50   G1    Z100                          (Compensating movement)
N60   G41                                 (Selection of TRO with data record D10)
N70   G1    Z0
N80   G2    X10   Y10   I-15              (Full circle with radius 15    )
N90   G0    Z100
N100  D2    D20                           (Other offset data record, i.e. )
                                          (other values for TLO and TRO )
N110  G1    Z0                            (The compensating movement of   )
                                          (the TLO takes place here       )
N120  G1    X20   Y20                     (Compensating movement of the TRO   )
N130  G02   X10   Y10   I-15              (The TRO is now performed with )
                                          (data record D20          )
N140  G0    Z100
N150  G40   G0. ...                       (Deselection of the TRO              )
```

11.3.1 Selection and approach blocks of the TRO

With selected TRO, an approach block is generated in the selected offset plane with the following motion block in the NC program.

Important: Approach blocks can be generated only for linear motion blocks, i.e. for the contour combinations of
Straight line - Straight line
Straight line - Circle.

Selection of the TRO with a contour combination
Circle - Straight line
Circle - Circle
leads to an error message.

The approach and withdrawal strategy can be set using the G functions G138 and G139. If G138 is selected, then the tool is moved with the approach block to the offset path (=tool radius) of the following block, this being to the point which lies perpendicular to the programmed path.
Without G138 or with G139, correction is made in the selected plane within the approach block.
The following diagrams (Fig. 11.3.1.1) show all possible approach blocks for the approved contour transitions. In each case two NC blocks N10 and N20 are shown for the three relevant contour transition angles for the approach strategy according to G139. The two selection/deselection strategies (G138/G139) are compared in 11.3.1.2.

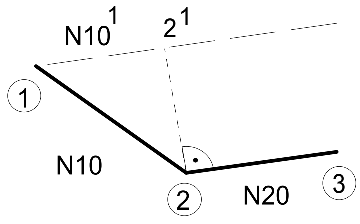
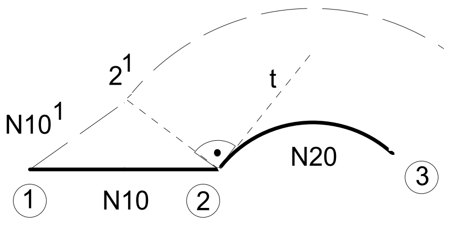
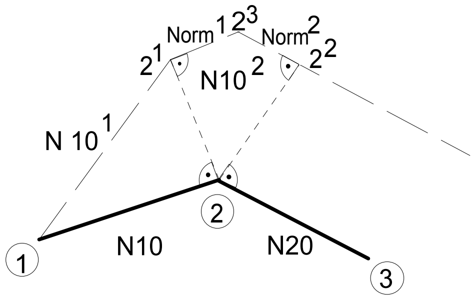
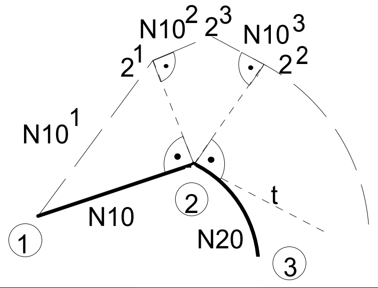
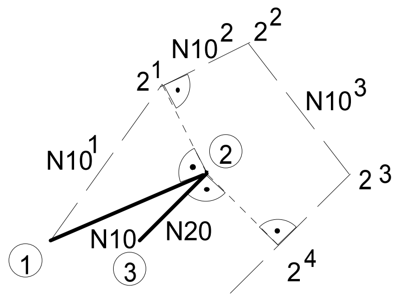
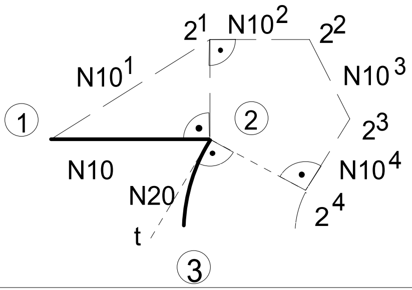
Kombinationstyp 1: 0° < ß ≤ 180°	**Kombinationstyp 2: 0° < ß ≤ 180°**
	
Kombinationstyp 1: 180° < ß ≤ 270°	**Kombinationstyp 2: 180° < ß ≤ 270°**
 Norm1 2^3 Norm2 2^1 $N10^2$ 2^2 N 10^1 2 1 N10 N20 3	
Kombinationstyp 1: 270° < ß ≤ 360°	**Kombinationstyp 2: 270° < ß ≤ 360°**
	 2^1 $N10^2$ 2^2 $N10^1$ $N10^3$ 1 2 2^3 N10 $N10^4$ N20 2^4 t 3

Fig. 11.3.1.1: Examples of approach blocks for the radius offset

Kombinationstyp = Combination type

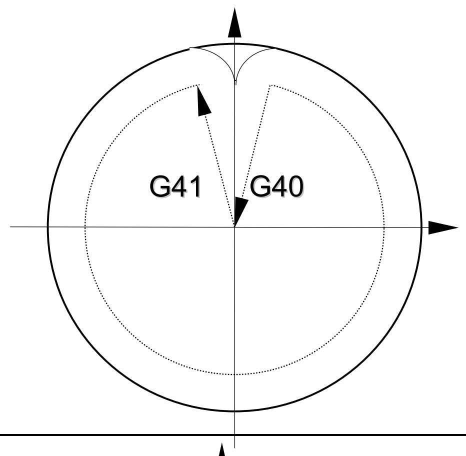
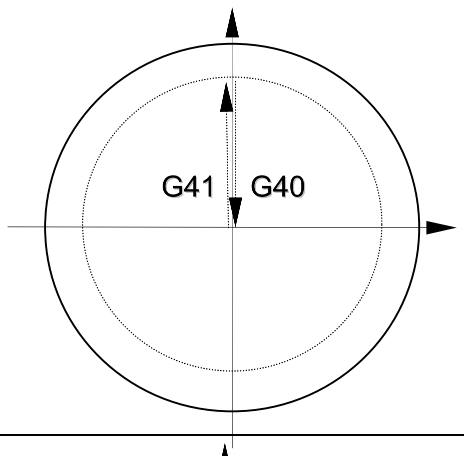
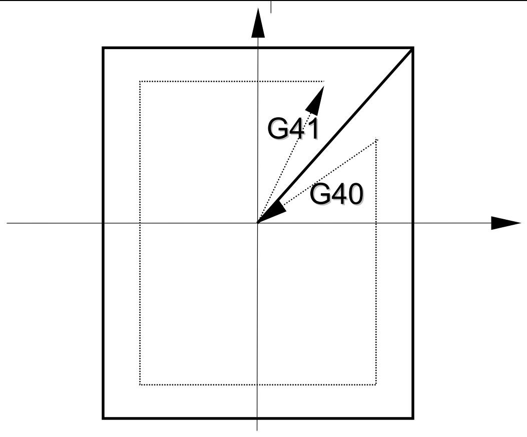
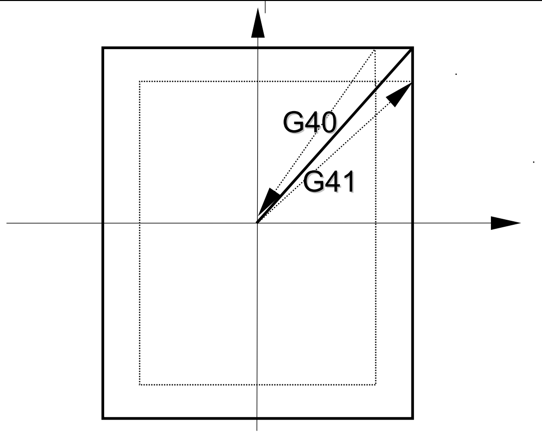
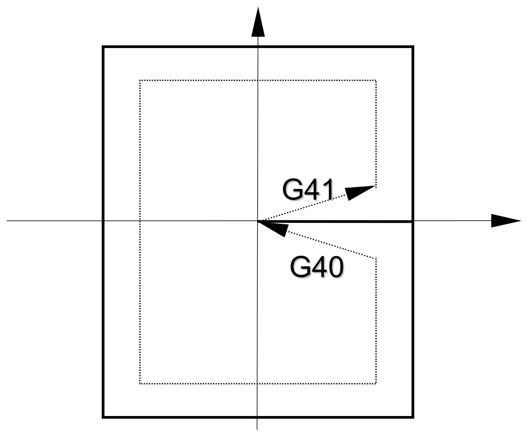
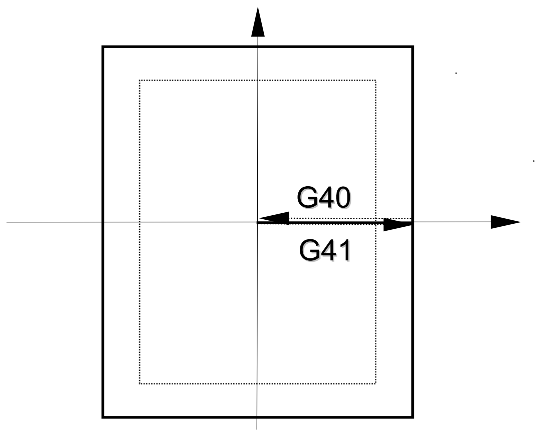
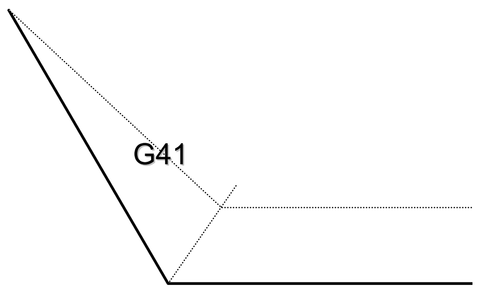
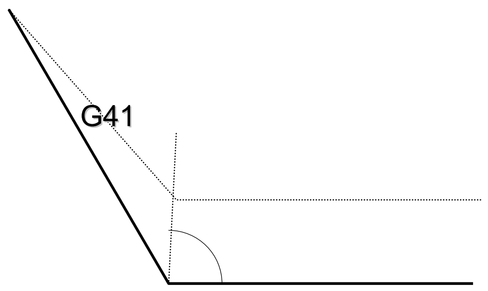
G139 Indirect selection (basic setting)	G138 Direct selection
	
	
	
	
Key: Programmed contour ——— Corrected contour ········	

Fig. 11.3.1.2: Examples of the selection/deselection strategy according to G139 or G138

11.3.2 Generation of offset blocks

The generation of offset blocks is subject to no restrictions regarding the NC block sequence, as apply for the TRO selection. The following diagrams show all possible contour transitions.

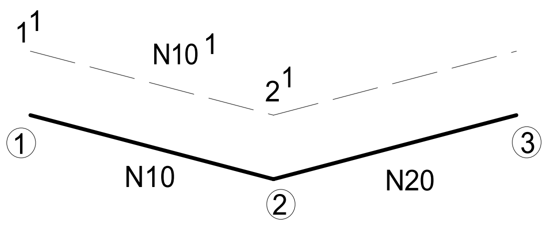
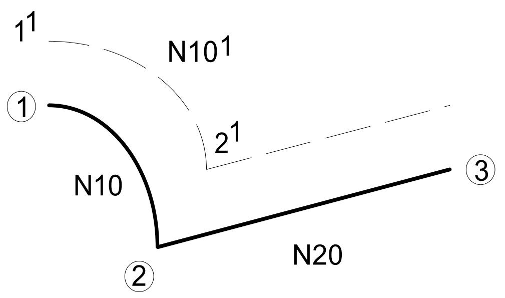
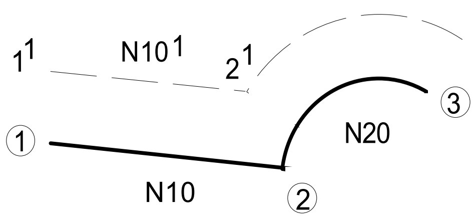
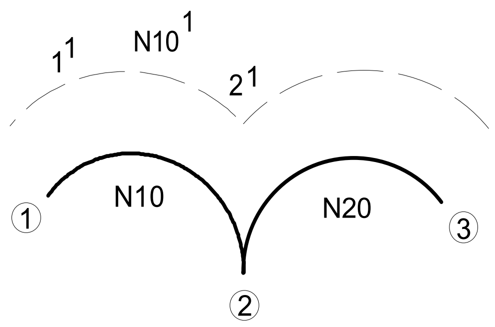
Kombinationstyp 1: 0° ≤ ß ≤ 180°, G41	**Kombinationstyp 3: 0° ≤ ß ≤ 180°, G41**
	
Kombinationstyp 2: 0° ≤ ß ≤ 180°, G41	**Kombinationstyp 4: 0° ≤ ß ≤ 180°, G41**
	

Fig. 11.3.2.1: TRO transitions: Combination cases for 0° ≤ ß ≤ 180 °

Kombinationstyp = Combination type

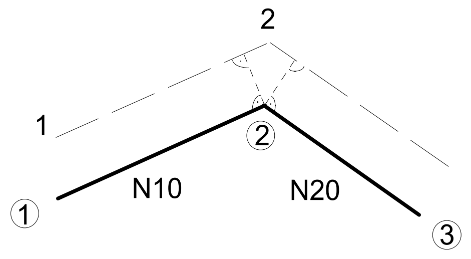
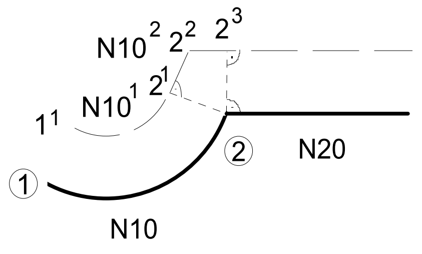
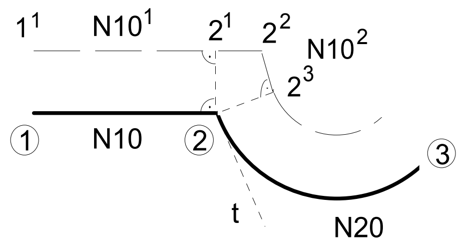
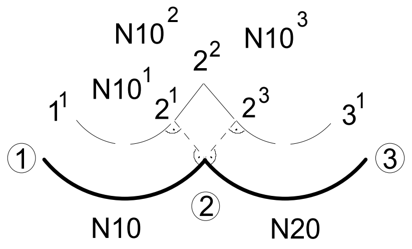
Kombinationstyp 1: 180° ≤ ß ≤ 270°, G41	**Kombinationstyp 1: 180°≤ ß ≤270°, G41**
	
Kombinationstyp 1: 180° ≤ ß ≤ 270°, G41	**Kombinationstyp 1: 180° ≤ ß ≤ 270°, G41**
	

Fig. 11.3.2.2: TRO transitions: Combination cases for 180° ≤ ß ≤ 270°

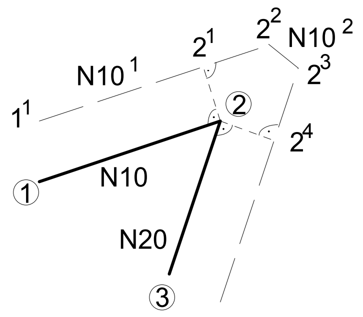
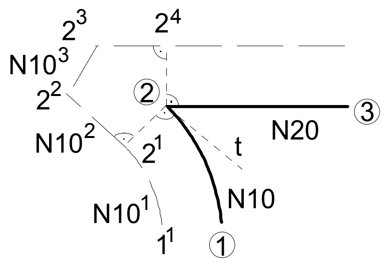
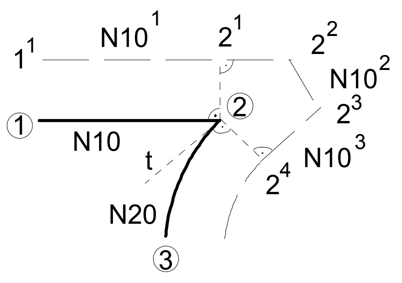
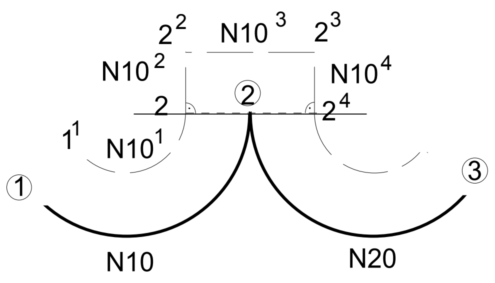
Kombinationstyp 1: 270° ≤ ß ≤ 360°, G41	**Kombinationstyp 3: 270° ≤ ß ≤ 360°, G41**
	
Kombinationstyp 2: 270° ≤ ß ≤ 360°, G41	**Kombinationstyp 4: 270° ≤ ß ≤ 360°, G41**
	

Fig. 11.3.2.3: TRO transitions: Combination cases for 270° ≤ ß ≤ 360°

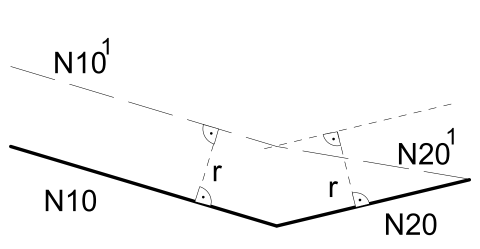
11.3.3 Behaviour on contour change

A direct contour change, e.g. from G41 to G42, is not permissible without explicit deselection (G40) of the TRO.

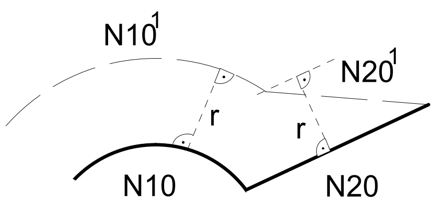
11.3.4 Withdrawal block on deselection of the TRO

Withdrawal blocks like approach blocks are possible only for linear motion blocks, i.e. in the Straight line - Straight line and Circle - Straight line contour transitions. Two different withdrawal strategies are possible for generation of the approach blocks. The standard approach blocks (not selected G138 function) are shown in the following diagram.

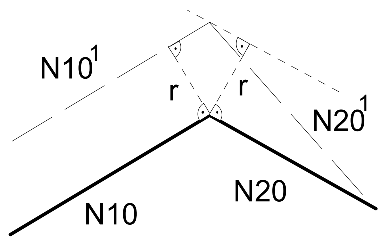
Kombinationstyp 1: 0° < ß ≤ 180°



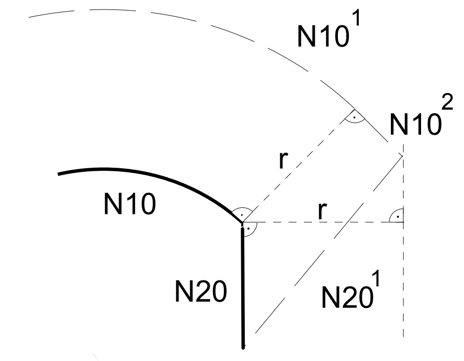
Kombinationstyp 2: 0° < ß ≤ 180°



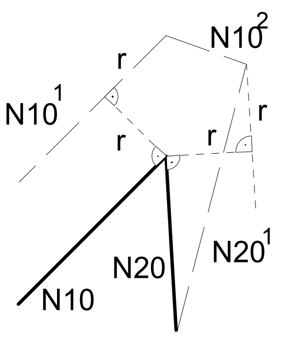
Kombinationstyp 1: 180° < ß ≤ 270°



Kombinationstyp 2: 180° < ß ≤ 270°



Kombinationstyp 1: 270° < ß ≤ 360°



Kombinationstyp 2: 270° < ß ≤ 360°

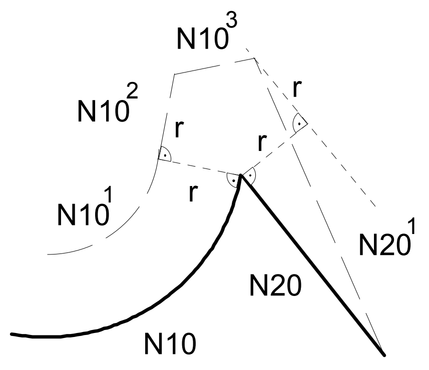


Fig. 11.3.4: Withdrawal blocks of the TRO

11.3.5 Limits of the TRO

For the calculation of intermediate blocks, the TRO considers the current and two further relevant NC blocks. "Relevant" here means: Motion blocks in the selected main plane in which the tool radius offset takes place. "Not relevant" would be, for instance, blocks with M functions, time delays (G04) or travels with one axis perpendicular to the main plane. If within these 3 blocks an intermediate block which would run in **opposite direction** to the programmed contour is calculated, then a contour violation is recognised by this. Examples of this are shown in Fig. 11.3.4.1.

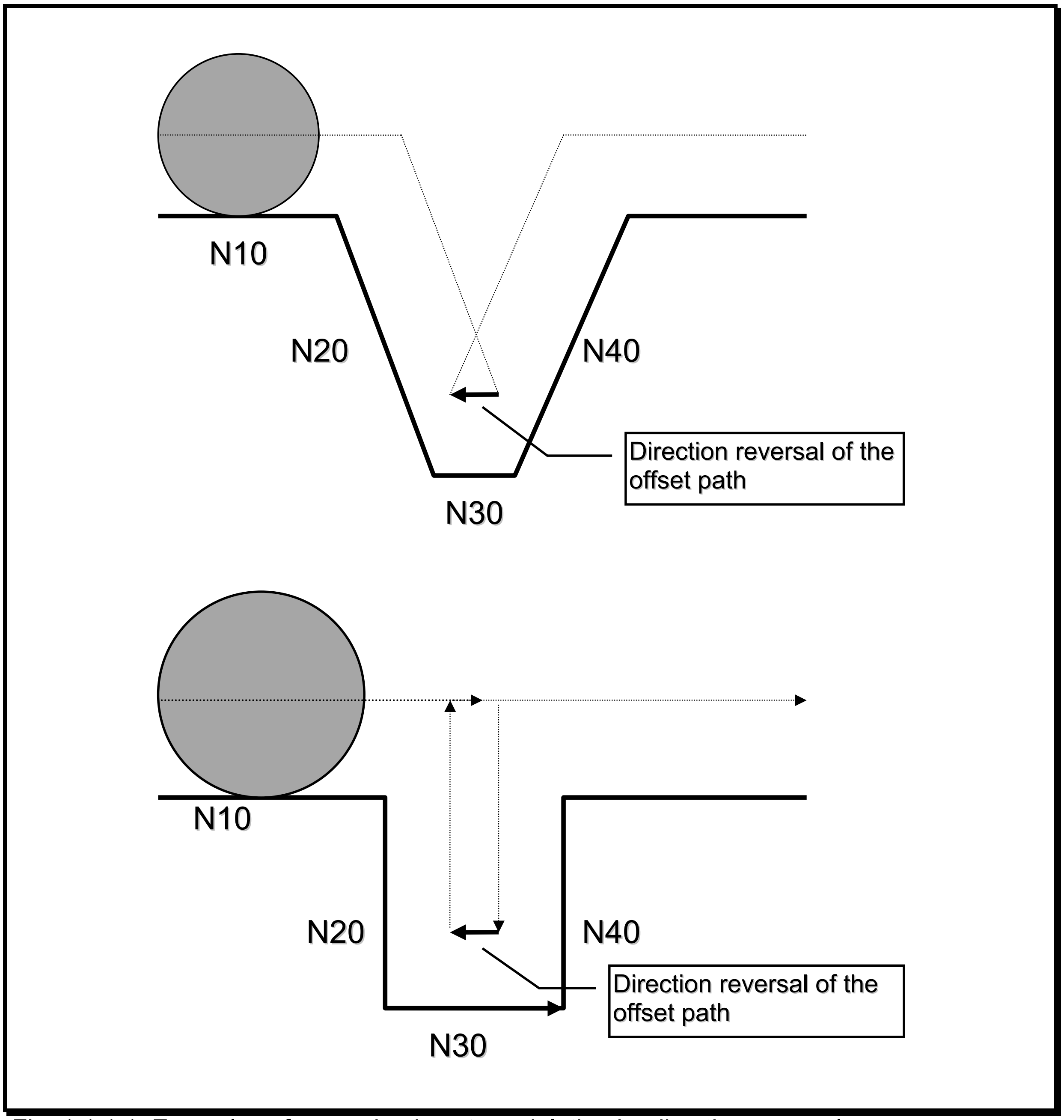

N10
N20
N40
Direction reversal of the offset path
N30
N10
N20
N40
Direction reversal of the offset path
N30

Fig. 1.1.1.1: Examples of recognised contour violation by direction reversal

If a constriction is produced by one or several blocks which are more than 3 blocks away from the current block, then this contour violation is not recognised by the TRO. An example of this is shown in Fig. 1.1.1.2.

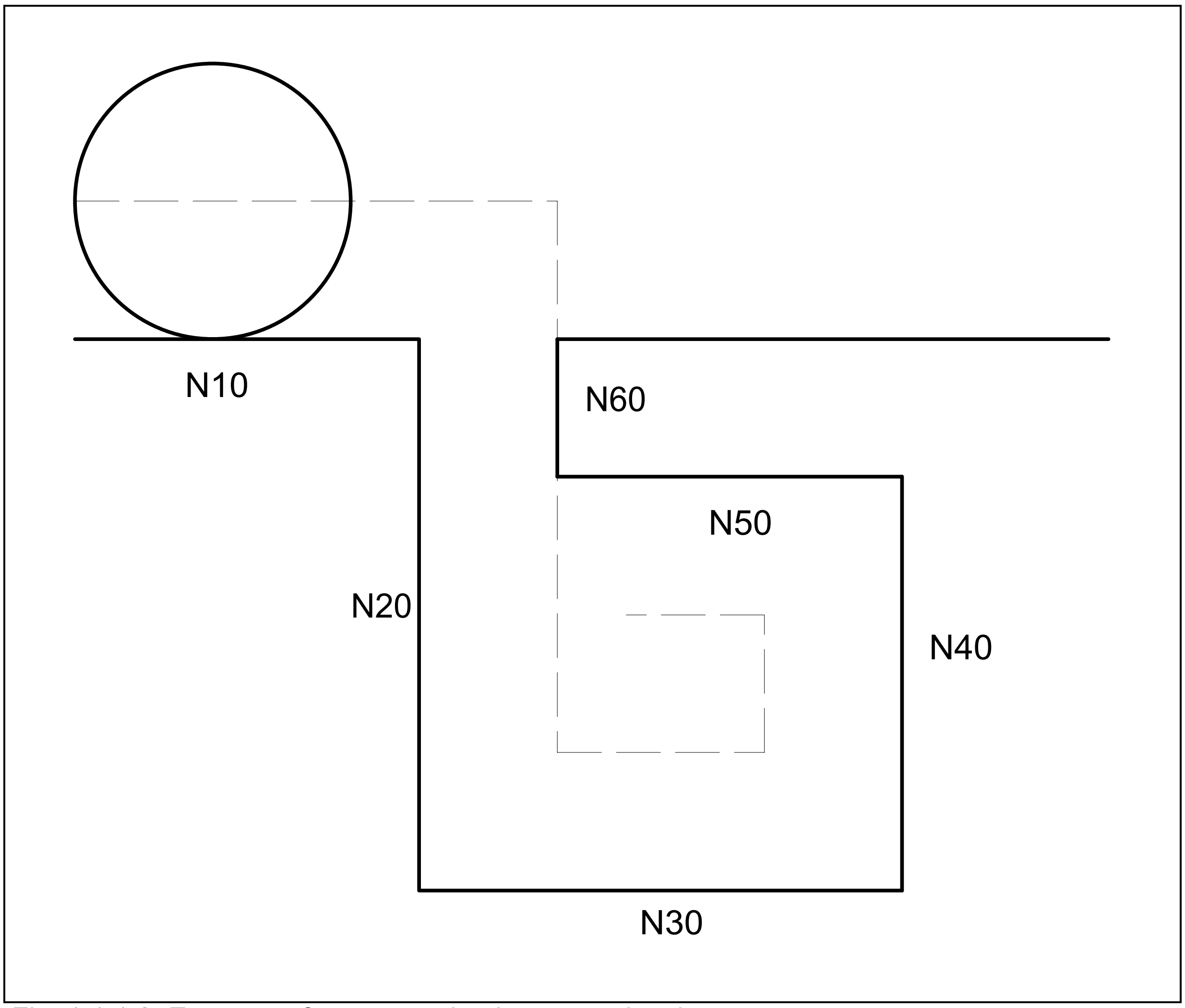


Fig. 1.1.1.2: Example of not recognised contour violation

12 TOOL MANAGEMENT

The NC decoder offers commands allowing access to the extended tool data for monitoring the tool service life.

12.1 Tool life monitoring

Commands for reading extended tool data:

TLIFE[n]	Reading the maximum tool life
TWARN[n]	Reading the WARN limit
TRES[n]	Reading the remaining tool life (RESidual time)
TLOC[n]	Reading the tool location number (tool LOCation)
TALT[n]	Reading the D number of the sister tool (ALTernative tool) with n = required D number

Corresponding values from the tool data record can be assigned to R parameters by use of these commands. The commands must always be used with a register assignment or an address letter.

```
Example:    R1=TALT[R2]Register assignment
            M06 TTLOC[R1]       Address letter „T“ with command
            M06 TTLOC[4]
```

12.2 Tool breakage

Command for writing tool data:

TRESDEL[n] Deletion of the residual tool life on tool breakage, which was recognised by the PLC and transferred to the NC with M112 (PLC parameter transfer).

This command has no register assignment. The current time is set internally to the maximum value (=TLIFE) by deleting the residual tool life (TRES[n]). This can be checked in the tool data record.

```
Example:    M112  S1001
            $IF R1001 ==1             If tool breakage:
              TRESDEL[R1]             Reset residual tool life
            $ENDIF
```

12.3 Programming

Example of programming of an alternative tool:

```
%1000                           (Main program)
...
R1=2                            (Selection tool D=2)
L999999
...
M30                             (Main program end)

%999999                         (Subroutine for WZn)
$IF TRES[R1] <= 0               (Query whether residual time has elapsed)
   R1=TALT[R1]                  (Determine alternative tool)
   $IF R1=0                     (No alternative tool defined)
     M80                        (Message to PLC)
     M30                        (Subroutine end)
  $ENDIF
   $IF TRES[R1] <=0             (Query whether residual time has elapsed)
         M80                    (Message to PLC that WZe has expired)
         M30                    (Program end)
  $ENDIF
$ENDIF
DR1 M06 T[TLOC[R1]]             (Change determined tool with location statement)
M29                             (Subroutine end)
```

13 APPENDIX

13.1 Command overview

13.1.1 "G" functions

G00 Linear interpolation in rapid traverse	17
G01 Linear interpolation	18
G02 Circle clockwise (cw)	19
G03 Circle counter-clockwise (ccw)	19
G161 Circle center absolute	20
G162 Circle center relative	20
G04 Dwell	21
G74 Homing cycle	22
G92 Datum point offset	22
G98 Set negative software limit switch	23
G99 Set positive software limit switch	23
G09 Delay at end of block	23
G94 Revolution feed OFF	26
G95 Revolution feed ON	26
G17 X-Y plane	26
G18 Z-X plane	26
G19 Y-Z plane	26
G20 Cancel mirroring	27
G21 Mirroring X on Y axis	27
G22 Mirroring Y on X axis	27
G23 Mirroring X on Y and Y on X	27
G70 Entries in inches	27
G71 Entries metric	27
G90 Absolute coordinate	27
G91 Incremental coordinate	27
G60 In-position programming	27
G40 Tool radius offset deselection	27
G41 Tool radius offset left of contour	27
G42 Tool radius offset right of contour	28
G138 Tool radius offset selection direct	28
G139 Tool radius offset selection indirect	28
G55 Selecting the 3rd zero offset	29
G56 Selecting the 4th zero offset	29
G58 Selecting the 6th zero offset	29
G59 Selecting the 7th zero offset	29
G36 Rounding selection	29
G37 Rounding deselection	29
G63 Fix override at 100%	29

13.1.2 "M" functions

13.1.3 Functions reserved according to DIN

13.1.4 Control block statements

13.1.5 Additional functions

14 IMPRESSUM

Title	**CNC ISO programming**
Objective	**Instruction for programming the CNC**
Part-Number	**27875**

History

Date
1998/07

Copyright

© AMK GmbH & Co. KG
No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose without the express written permission of AMK GmbH + Co. KG. Violations are subject to legal action. All rights in case of patent filings or user-sample registrations are reserved.

Disclaimer

We reserve the right to change the contents of the documentation and the availability of products at any time without prior notice.

Service

Tel.: **+49/(0)7021 / 5005-191, Fax –193**

Business Hours:
Mo-Fr 7.30 - 16.30, On weekends and holidays calls are forwarded to an emergency response number by the automated answering system.

To assure a fast and accurate response to solve customer problems we ask for your cooperation in providing us with the following information:

- Nameplate data
- Software version
- System configuration and application
- Description of problem and presumed cause of failure
- Diagnostic message (error code)

Publisher

AMK Arnold Müller Antriebs- und Steuerungstechnik GmbH & Co. KG
Gaußstraße 37 – 39, 73230 Kirchheim/Teck

Tel.: 07021/5005-0, Fax: 07021/5005-176
E-Mail: info@amk-antriebe.de

For further information www.amk-antriebe.de