



# DOCUMENTATION ISG-kernel

## Functional description Block search

Short Description:  
FCT-C6

© Copyright  
ISG Industrielle Steuerungstechnik GmbH  
STEP, Gropiusplatz 10  
D-70563 Stuttgart  
All rights reserved  
[www.isg-stuttgart.de](http://www.isg-stuttgart.de)  
[support@isg-stuttgart.de](mailto:support@isg-stuttgart.de)

Documentation version: 1.05  
07/11/2024

# Contents

<b>1</b>	<b>Overview</b>	<b>5</b>
<b>2</b>	<b>Description</b>	<b>6</b>
<b>3</b>	<b>Block search types</b>	<b>8</b>
3.1	Type 1: Continuation position defined by file offset	8
3.2	Type 3: Continuation position defined by block counter	10
3.3	Type 4: Continuation position defined by block number	11
3.4	Type 5: Block search at program end	13
3.5	All types: Block search with additional breakpoint	14
3.5.1	Type 6: Breakpoint without block search	15
3.6	All types: Block search as of a specific program position (file offset)	16
<b>4</b>	<b>General parameters</b>	<b>18</b>
4.1	Continuation position within a block	18
4.1.1	Covered distance in current block in per thousand	18
4.1.2	Distance covered from program start (#DISTANCE PROG START)	20
4.2	Restart to the contour after block search	26
4.2.1	Manual restart during block search	29
4.2.2	Track the C axis with block search (#CAX TRACK)	31
4.2.3	Path-dependent angle compensation at block search (#VECTOR OFFSET)	33
4.3	Backward motion after block search	34
<b>5</b>	<b>Block search interface and parameters</b>	<b>35</b>
5.1	General block search parameters	35
5.1.1	Covered distance	39
5.2	Block search type 1 parameters: File start/end position	41
5.2.1	Start position	41
5.2.2	End position	43
5.3	Block search type 3 parameters: Block counter	45
5.4	Block search type 4 parameters: Block number	45
5.5	Status data: Access via CNC objects	47
5.6	Status data: Access via HLI	47
5.6.1	HL access with CNC Version < V2.11.28xx	53
<b>6</b>	<b>Lock program areas for block search (#BLOCKSEARCH)</b>	<b>56</b>
<b>7</b>	<b>On/off handshake with PLC</b>	<b>58</b>
<b>8</b>	<b>Known restrictions</b>	<b>61</b>
<b>9</b>	<b>Examples</b>	<b>63</b>
9.1	Block search type 4	63
9.1.1	Specify block number and pass counter	63
9.1.2	Specify the block number and covered distance in the block	64
9.2	Block search type 3	65
9.2.1	Specify block counter	65
<b>10</b>	<b>Exceptions and errors</b>	<b>67</b>



---

<b>11 Appendix .....</b>	<b>68</b>
11.1 Suggestions, corrections and the latest documentation.....	68
<b>Keyword index .....</b>	<b>69</b>

## List of figures

Fig. 1:	Interactions and interfaces with block search .....	5
Fig. 2:	Block search position by start and end .....	9
Fig. 3:	Continuation position by block counter .....	10
Fig. 4:	Continuation position by block number .....	12
Fig. 5:	Continuation position by block number and pass counter .....	12
Fig. 6:	Continuation position at program end .....	13
Fig. 7:	Block search and additional breakpoint .....	14
Fig. 8:	Additional breakpoint .....	15
Fig. 9:	Jump point with block search .....	16
Fig. 10:	Continuation position with current block split by per thousand .....	18
Fig. 11:	Per thousand display with two inserted polynomial blocks .....	19
Fig. 12:	Per thousand display with one inserted polynomial block .....	19
Fig. 13:	Distance from program start .....	20
Fig. 14:	Influence of starting movement is prevented by NC commands .....	21
Fig. 15:	Continuation position with current block split by distance from program start .....	22
Fig. 16:	Search for continuation position by distance from program start over several blocks .....	23
Fig. 17:	G74, G100, #FLUSH WAIT during search for continuation position .....	24
Fig. 18:	Continuation position by distance from program start before current block .....	25
Fig. 19:	Use of different tool radii .....	26
Fig. 20:	Use of block search to restore the program context .....	27
Fig. 21:	Manual restart to the contour .....	28
Fig. 22:	Restart to the contour .....	29
Fig. 23:	Sequence of block search and completion of manual restart .....	30
Fig. 24:	Example of continuous alignment of the C axis to the contour .....	31
Fig. 25:	Backward motion after block search .....	34
Fig. 26:	Block search states .....	48
Fig. 27:	Interaction between BOOLEAN-LC control unit and PLC .....	60

# 1 Overview

## Task

The operator can start machining at what is called the continuation position at any point in the program. After a program is interrupted (e.g. tool breakage), this is a quick method to reactivate machining at the point of interruption.

The continuation position can be defined using a number of different block search types (file offset, block counter, block number, etc.).

It is imperative to restore the entire program context at the starting point specified here (program parameters, axis positions, etc.). This is ensured by processing the program up to this continuation position **without** any axis motion (simulation). Technology functions are signalled to the PLC, even during the simulation. All the vital machine functions for the machining process are then activated at the continuation position (e.g. coolant, velocity).

When the program reaches the continuation position, the axes can be moved to their current positions at this program position either manually or automatically.

The operator can then start the continued execution of the program.

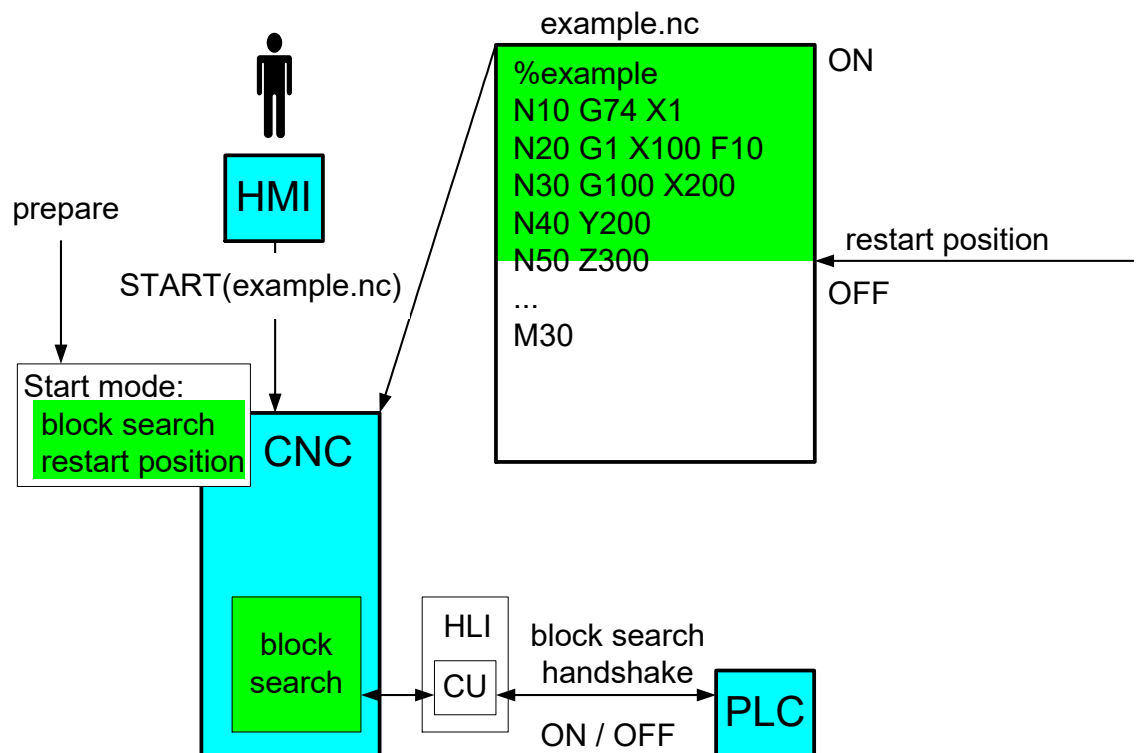


Fig. 1: Interactions and interfaces with block search



### Notice

Block search type 2 is no longer available.

## 2 Description

### Program start in block search mode

In block search, a specific program point, which is called the continuation position, is controlled without real axis motions. After this program point is reached, the axes are positioned at the contour either manually or automatically and the machining process is continued.

### Continuation position

The continuation position is the position at which processing the NC program ends in block search and real machining starts on the workpiece.

Normally this is the interrupt position of a previous machining process. The reason for an interrupt may be to measure a workpiece or a tool breakage.

### Continue in a motion block

The continuation position may also be located within an NC block. For this reason, one option with all block search types is to define the covered distance in the NC block as of which actual machining should continue.



#### Notice

Block search must be selected on the control panel (HMI) or via the PLC **before** the main program starts.

Block search selection is modal, i.e. it remains active after the NC program ends. Block search must then be deselected explicitly, e.g. via the HMI.

### Simulation and axis motions

No physical axis motion takes place in block search mode up to the continuation position. However, the NC program is completely decoded to produce the required program context at the continuation position.

Since no real interpolation takes place, the continuation position is usually reached very much faster than with normal program processing.

### Simulation and technology functions

In block search mode, the PLC receives all technology functions as in normal mode so that machine functions can be switched correctly. These functions must also be acknowledged.

The PLC synchronises the selection/deselection of block search mode. This may result in the specific handling of certain technology commands if this is supported by the PLC (group handling, activating certain functions before block search is deselected).



#### Release Note

This extension is available as of Builds V2.11.2018.09, V2.11.2804.10 and V3.1.3030.2.

## Move to continuation position

---

When the continuation position is reached in block search mode, ensure that the axes were moved from their current actual positions to the restored command positions before actual machining starts (Restart to the contour)

At the same time, the spindles must reach their last commanded speeds.

## Real mode

---

After switching over to real machining, the NC program continues as if it had been started without block search mode. A number of methods are available for restart to the contour (see Restart to the contour after block search [▶ 26]).

## 3 Block search types

### 3.1 Type 1: Continuation position defined by file offset

#### Block search type 1:

---

##### Continuation position and end position by file offset

This block search type defines a processing range by specifying a continuation/end position (start/end mark) using file offset. Processing starts at the continuation point and ends after the end position is reached. The NC program is then ended immediately. If no end position is specified, processing continues until NC program end M30.

File offset defines the continuation/end position as the distance to file start of the NC program.

It is the operator's responsibility to define the file offset. File offset is always determined in relation to the start of the NC row of **each** NC program (main program (HP), global subroutine (UP)).

File offset and other data to determine the continuation/end position are transferred as parameters when block search is selected.

##### Continuation position

Parameter:

- file offset,
- file name,
- identifier whether file is in the HP-(0) or UP path (1),
- pass counter (optional),
- covered distance within the block (optional)

##### end position (optional)

Parameter:

- file offset,
- file name,
- identifier whether file is in the HP-(0) or UP path (1),
- pass counter (optional)





## Example

Start in the main program in 1st pass at file offset 100 at block start, end at file offset 239

	File offset		
	7	N10 ...	No axis movement
	18	N20 ...	
	31	N30 ...	
	55	N40 ...	
	82	N50 ...	
Start of execution →	100	N60 ...	Axis movement
	121	N70 ...	
	145	N80 ...	
	152	N90 ...	
	176	N100 ...	
	194	N110 ...	
End of execution →	210	N120 ...	
	239	N130 ...	Program end
	...	...	
	500	M30	

Fig. 2: Block search position by start and end

## 3.2 Type 3: Continuation position defined by block counter

### Block search type 3:

#### Continuation position by block counter

At program start the block counter (controller-internal) is incremented by 1 for each NC row decoded. The block counter is also incremented continuously for comment lines, empty lines, in loops and during subroutine calls. The block counter is displayed to the PLC or the HMI during normal program execution. To determine the continuation position, the block counter is transferred as one of the parameters when block search is selected.

Parameter:

- block counter,
- covered distance within the block (optional)



#### Example

##### With block counter

Start at block counter 12

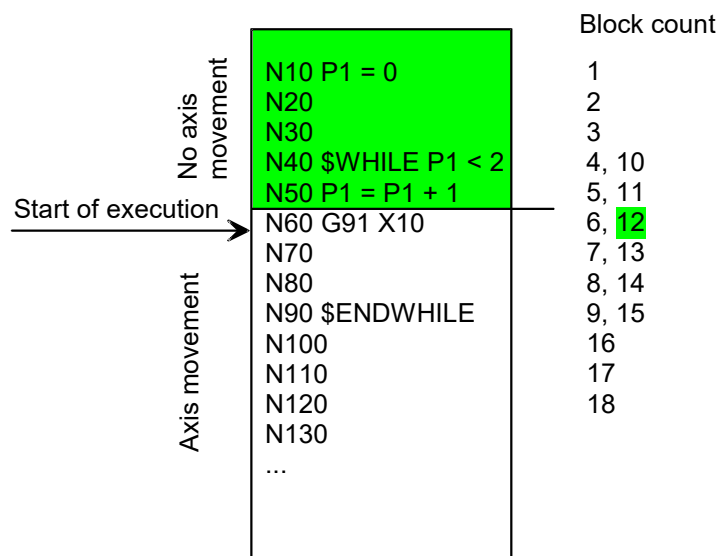


Fig. 3: Continuation position by block counter

## 3.3 Type 4: Continuation position defined by block number

### Block search type 4:

#### Continuation position by block number and program name

The block number is the number (N word) of an NC row in the NC program. To generate the NC program, the operator or the system is responsible for providing every row which is relevant later, even only once, with a unique block number. However, block number ambiguities may occur due to local and global subroutines. Therefore, to determine the exact continuation position, the program name (%...) can also be specified as an option to conduct a search for the block number.

Parameter:

- block number,
- Program name [▶ 46] (%...) via CNC object (optional),
- covered distance within the block (optional)



#### Release Note

**The optional specification of a program name is available as of Build V3.01.3000.00.**

#### Continuation position by block number and pass counter

Since the block number is passed several times, e.g. in loops, it is not always unique on its own. In this case, machining start may be optionally triggered by the additional specification of a pass counter. The program name (%...) can also be specified as an additional option here.

Parameter:

- block number,
- pass counter,
- program name (%...) (optional),
- covered distance within the block (optional)



#### Notice

**It is the user's responsibility to supply the pass counter.**



#### Example

##### With block number

Start at block number 60

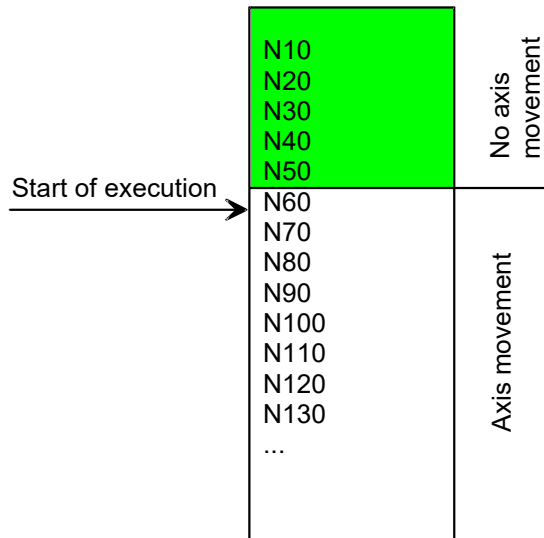


Fig. 4: Continuation position by block number



### Example

#### With block number and pass counter

Start at block number 60 in 5th pass

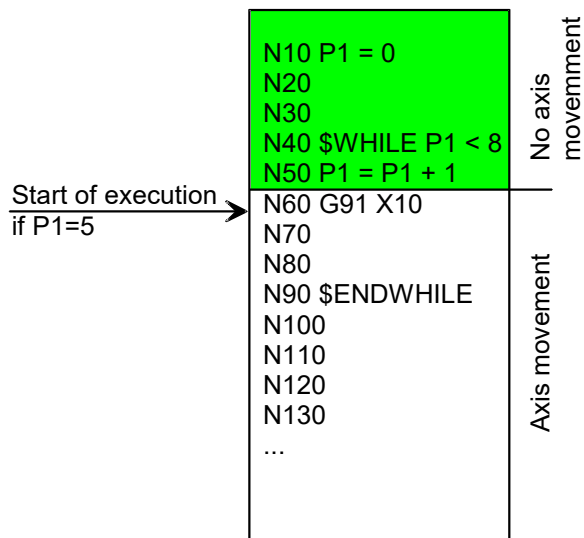


Fig. 5: Continuation position by block number and pass counter

## 3.4 Type 5: Block search at program end

### Block search type 5:

#### Continuation position at program end

This special block search type is used in particular in job planning on a simulation system for a rapid test of NC programs. The continuation position is placed implicitly at program end (M17, M30). The NC program is only decoded but no axis positions are interpolated. This permits long NC program to run through quickly.

When program end is reached, the complete NC program is terminated in block search mode.

#### Distinction from “Dry Run”

In Dry Run [FCT-C17] mode the processing speed is identical to execution on the machine. Axis positions are interpolated but axis motions are not executed. This operation mode is only useful if it is executed directly on the machine controller.



### Example

#### Block search at program end M30

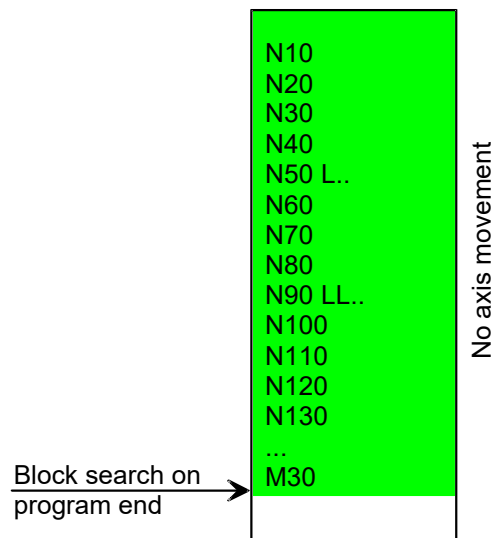


Fig. 6: Continuation position at program end

### 3.5 All types: Block search with additional breakpoint

#### Set a breakpoint with block search

An automatic stop is inserted by specifying a breakpoint by the **distance from program start** (cf. explicitly programmed M0). This permits the automatic instrumentation of an NC program with an M0.

The breakpoint can be specified in addition to the continuation position of the block search. The breakpoint must be placed **after** the continuation position.

During block search, the NC program is processed up to the specified continuation position without axis motion (green section). The axes are then moved for real (grey and white section).

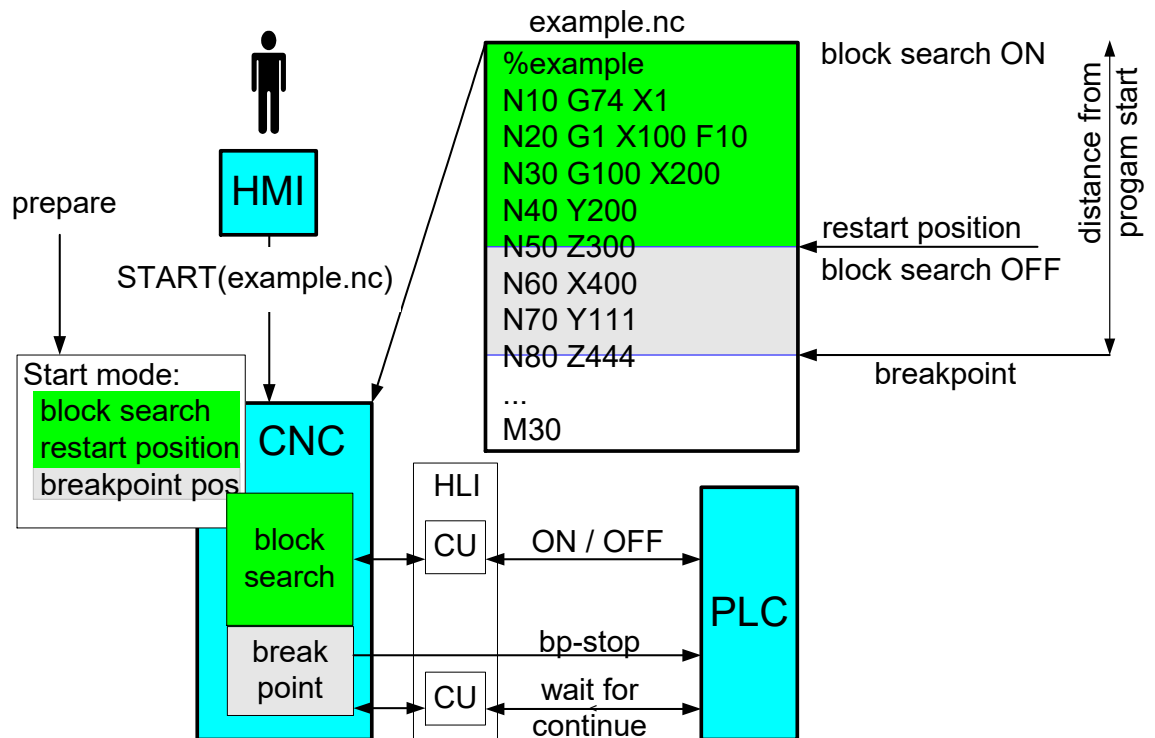


Fig. 7: Block search and additional breakpoint

#### Definition of breakpoint

The **breakpoint** is defined in a similar way to the continuation position of block search before the start of an NC program by the distance from program start by

mc\_cmd\_bs\_breakpoint\_position\_w

(see section Additional breakpoint).

## Interaction with PLC

The stop caused by the breakpoint is displayed at the breakpoint (see section HLI: Stop conditions). The block search state then signals “Wait for continue motion” (see section HLI: Block search state). This is displayed until the PLC requests release for continuation of machining (see [HLI//Continue motion]).



### Notice

The breakpoint is only evaluated the first time it is reached in forward direction. If the contour is then moved in backward/forward direction, no stop is executed at the breakpoint.

### 3.5.1

## Type 6: Breakpoint without block search

### Set a breakpoint without block search

If a breakpoint is to be set without a previous block search, this can be done by specifying the block search type `SIMULATION = 6`.

In this case the program is processed normally with an additional breakpoint (grey and white section).

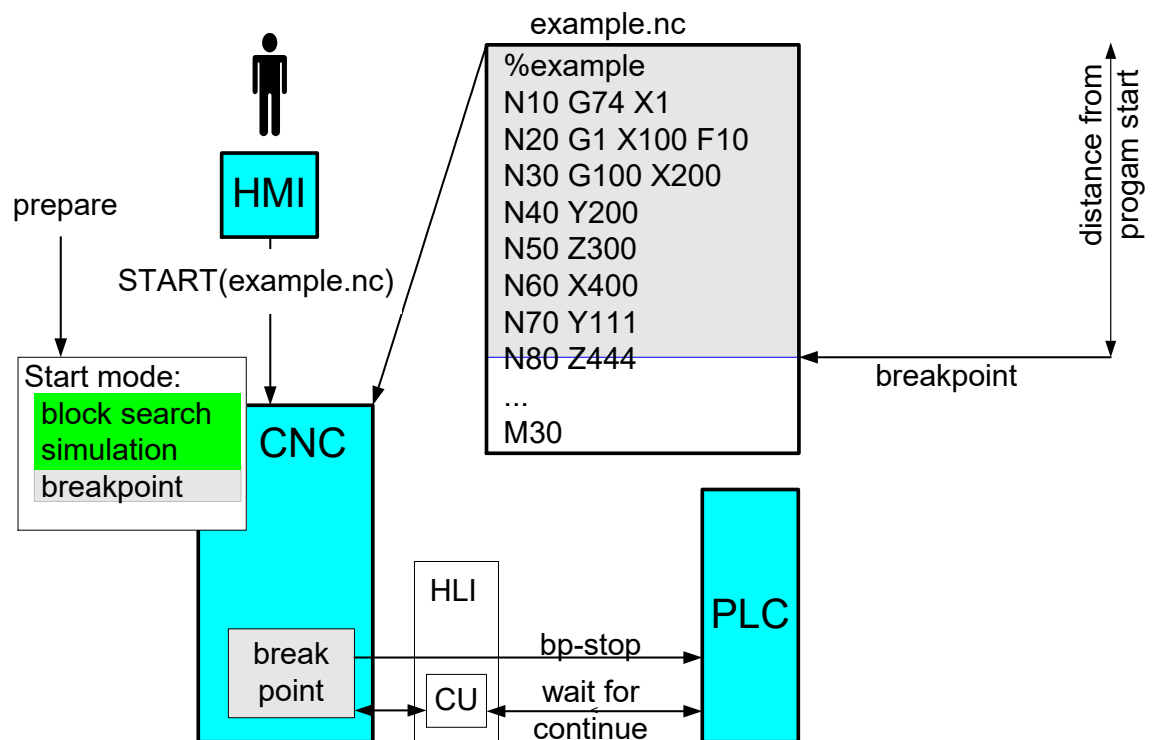


Fig. 8: Additional breakpoint

### 3.6 All types: Block search as of a specific program position (file offset)

#### Set a jump point with block search

The NC program can be started with a file offset in combination with any block search type. The file offset defines a jump to a known position in the NC program.

The program part before the jump point is ignored. Evaluation starts at the jump point as for a program shortened by file offset.

The jump point can be specified in addition to the continuation position of the block search. The jump point must be placed **before** the continuation position.

The full technology scope must be reproduced at the jump point so that machining can continue. The NC program then runs as usual up to a continuation position in the commanded block search type.

This method saves time in the block search with large NC programs.

During block search, the NC program is processed up to the specified continuation position (green section) without axis motion after the jump point (grey section). The axes are then moved for real (white section).

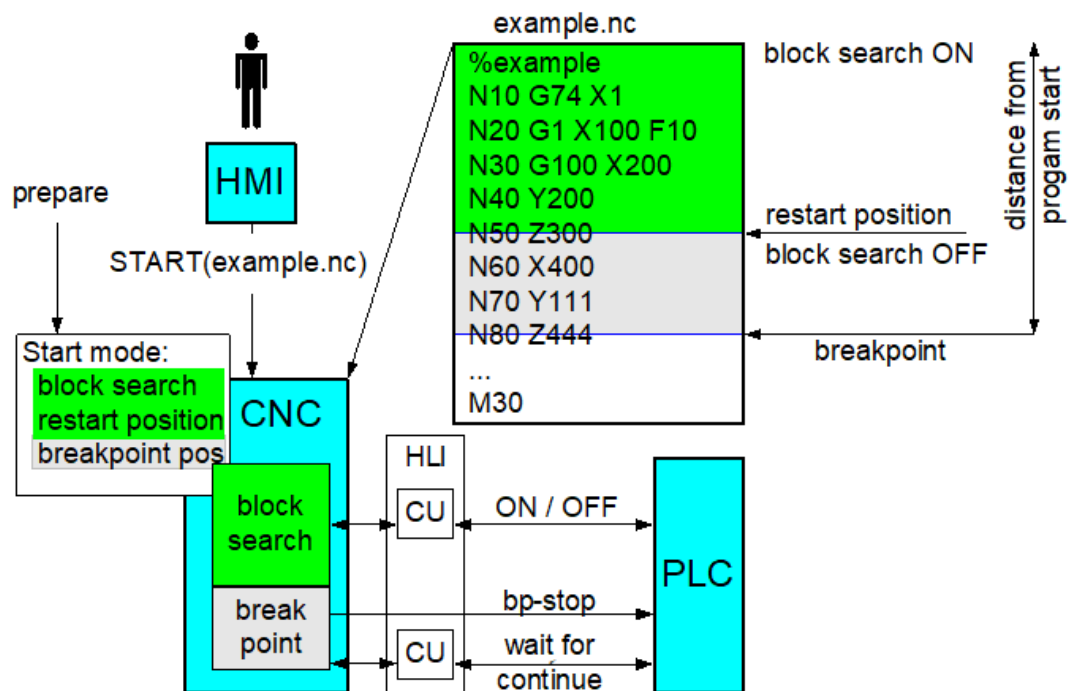


Fig. 9: Jump point with block search



## Definition of jump point

---

The jump point is defined in a similar way to the continuation position of block search before the start of an NC program by

`mc_command_file_offset_w`

(see section Program start at file offset).

File offset to define a jump point can also be used without block search. Processing then starts directly at the jump point as for a program shortened by file offset.

## 4 General parameters

### 4.1 Continuation position within a block

#### Position within motion block

Normally block search is disabled between two NC rows. However, if this is a motion block, it may also be required to specify the continuation position **within the motion block** with greater precision. Therefore, there is an option for motion blocks to define a path distance within the block in addition to the block specification. This path distance can be specified in two ways:

Specify the covered distance in the block in per thousand referred to the current distance length of the block (see section Covered distance in current block in per thousand [► 18]).

Specify the covered distance from program start or from the last #DISTANCE PROG START CLEAR (see section Covered distance from program start).

#### 4.1.1 Covered distance in current block in per thousand

##### Per thousand

The position within a motion block is defined in per thousand.

The per thousand display can be read as a display datum on the PLC interface (see Chapter HLI: Covered distance in the block – per thousand) when the current motion is interrupted.

Alternatively, the per thousand display can be specified as a purely required value without being previously read.

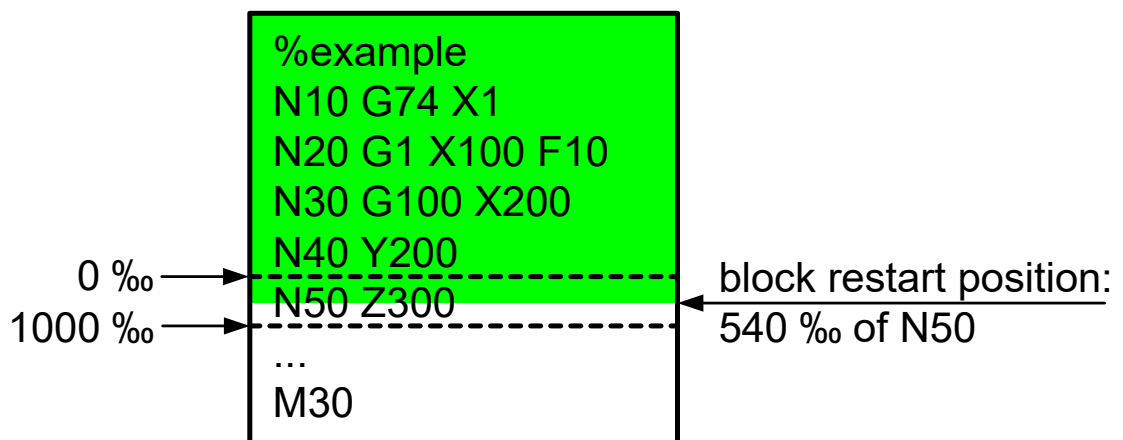


Fig. 10: Continuation position with current block split by per thousand



#### Notice

If the restart position is defined by the relative display in relation to the motion block (per thousand), this position changes – within the range of resolution accuracy – even for restart when tool radius compensation is active and another tool radius is not.

This means that, if tool radius compensation is active, a tool with a different radius can be replaced in block search.

## Value range of per thousand display

The per thousand display of a block is normally within the range [0, 1000].

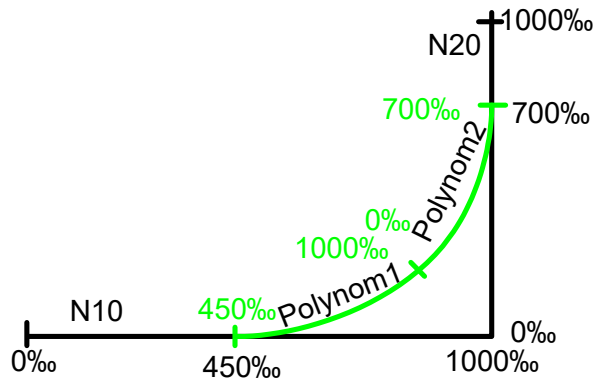


Fig. 11: Per thousand display with two inserted polynomial blocks



### Notice

If only one block is inserted by the CNC between two original blocks (e.g. when #HSC [OPMODE = 1]), its distance may be between [0,2000]

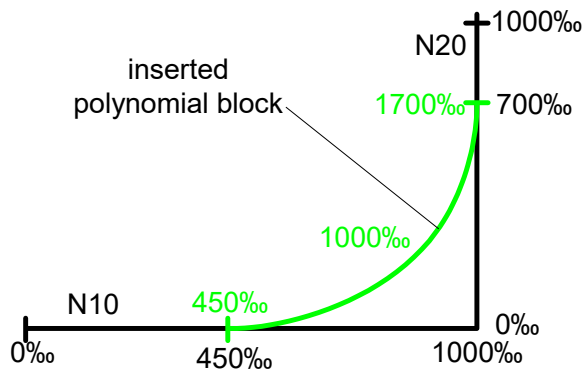


Fig. 12: Per thousand display with one inserted polynomial block

## 4.1.2 Distance covered from program start (#DISTANCE PROG START)

### Distance from program start

Every motion block can be identified by the covered distance from program start. The distance is displayed on the PLC interface during the process (see section HLI: Covered distance - path increments, section Covered distance - path increments). The distance is formed as the sum of the main axis motions of all previous motion blocks. If a motion block contains no main axis motion, the distance of the tracking axis that moves at its dynamic limit is added.

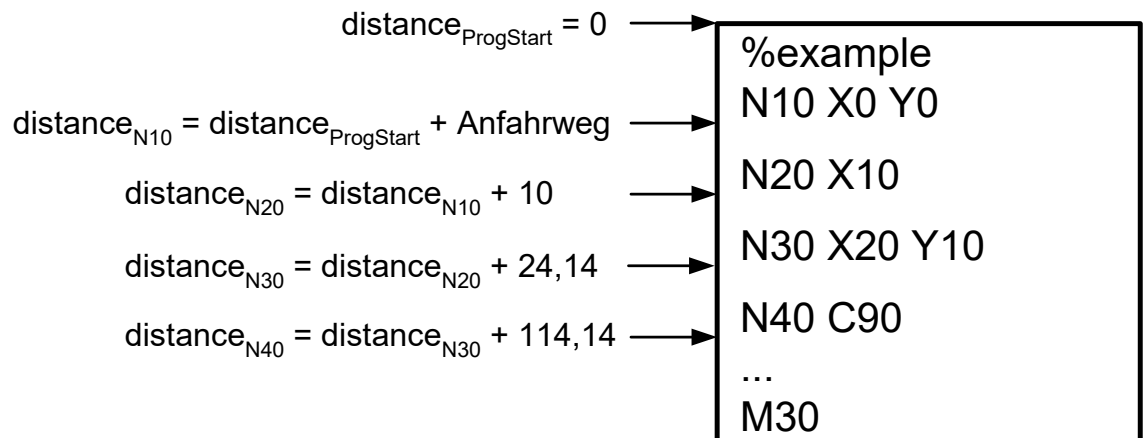


Fig. 13: Distance from program start



#### Notice

To approach the identical position in the block search, the original contour may not be changed in the definition of position by distance. This means that, if tool radius compensation is active, an identical tool must be replaced in block search.

However, if starting is executed with different tool geometries, the “Distance from program start” changes.

### Distance display, NC commands

To render this independent of the starting axis position, the distance display can be controlled by the following commands in the NC program.

#DISTANCE PROG START ON	modal
#DISTANCE PROG START OFF	modal
#DISTANCE PROG START CLEAR	non-modal

ON	Distance of following motion blocks is evaluated for display (default after program start).
OFF	Distance of following motion blocks is not evaluated for display.
CLEAR	The current distance is set to <b>0</b> (default at program start)

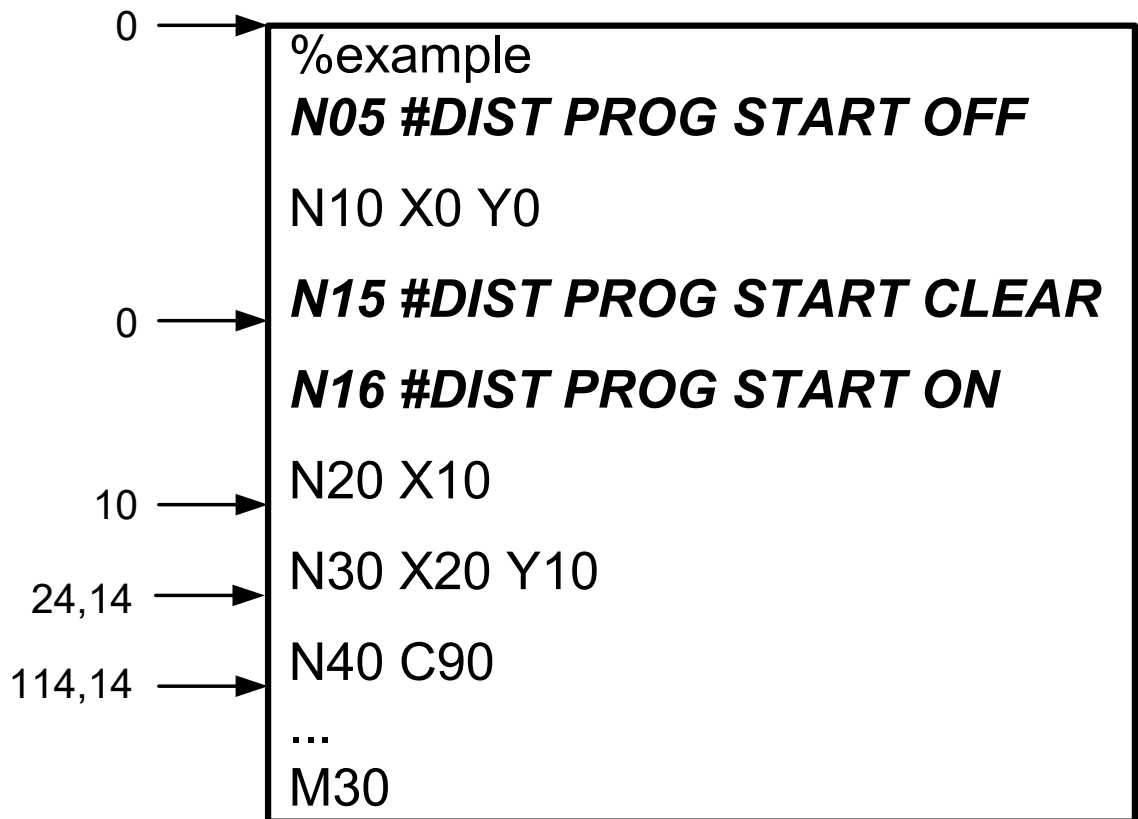


Fig. 14: Influence of starting movement is prevented by NC commands



## Programming Example

### #DISTANCE PROG START

```

%example
N10 #DISTANCE PROG START OFF
N20 G01 G90 X0 Y0 Z0 F1000

N100 G92 X33 Y55          ; Offset -> position of part
N110 X0 Y0 Z0            ; Starting movement of geometry
N120 #DISTANCE PROG START ON
N130 X100                ; distance = [ 0, 100]
N140 Y100                ; distance = [ 100, 200]
N150 X0                  ; distance = [ 200, 300]
N160 Y0                  ; distance = [ 300, 400]
N170 #DISTANCE PROG START OFF
...
N200 G92 X600 Y700       ; Offset -> position of part
N210 X0 Y0 Z0            ; Starting movement of geometry
N220 #DISTANCE PROG START ON
N230 X100                ; distance = [ 400, 500]
N240 Y100                ; distance = [ 500, 600]
N250 X0                  ; distance = [ 600, 700]
N260 Y0                  ; distance = [ 700, 800]
N270 #DISTANCE PROG START OFF
M30
    
```

## Position in block by distance

If a block is stopped or interrupted, the actual distance from program start can be read on the PLC interface (see section HLI: Covered distance - path increments) and the continuation position can be specified more exactly by this distance display.



### Example

#### Continuation position with current block split by distance from program start

Block search type                    4  
Block number                        30  
Distance since program start    16 mm  
Continuation position is within block N30

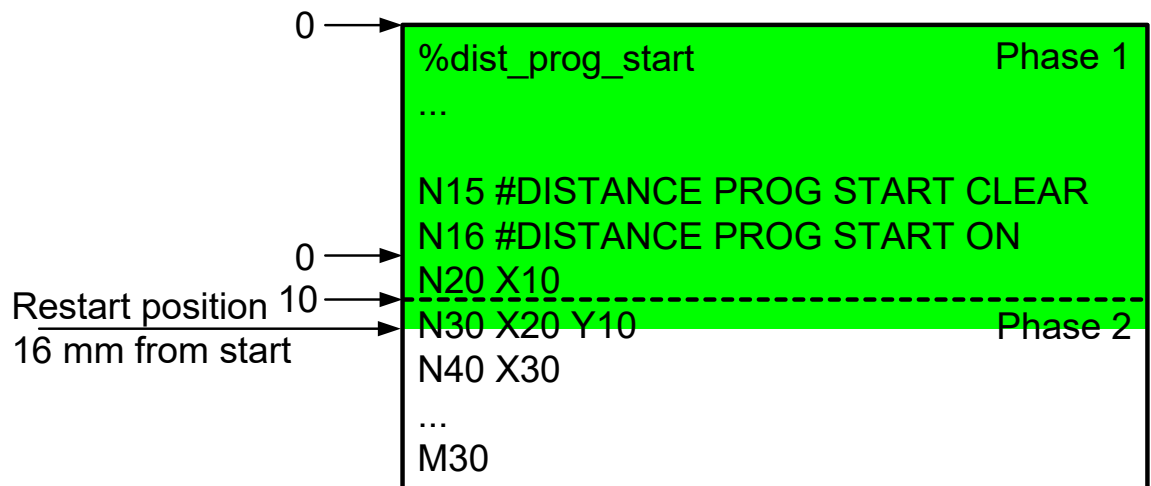


Fig. 15: Continuation position with current block split by distance from program start



## Example

### Search for continuation position by distance from program start over several blocks

Block search type                4  
 Block number                     30  
 Covered distance            234.79 mm  
 Continuation position is within block N50

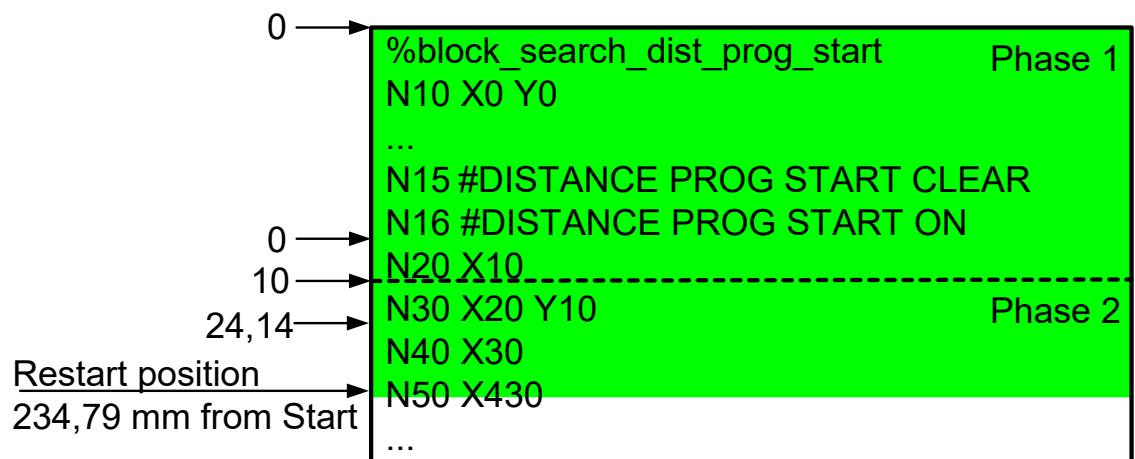


Fig. 16: Search for continuation position by distance from program start over several blocks



## Attention

A homing run G74 or a measurement run G100 are movements which are ended by an external signal. The signal is simulated during block search up to the specified block limit (Phase 1).

If block search is additionally extended by specifying the covered distance (Phase 2), no commands such as G74 or G100 may occur in this section of the NC program since they cannot be simulated there. However, if this type of command occurs, an error message is output.



## Example

### G74, G100, #FLUSH WAIT during search for continuation position

Block search type            4  
 Block number                30  
 Covered distance        495.12 mm  
 Continuation position is within block N100

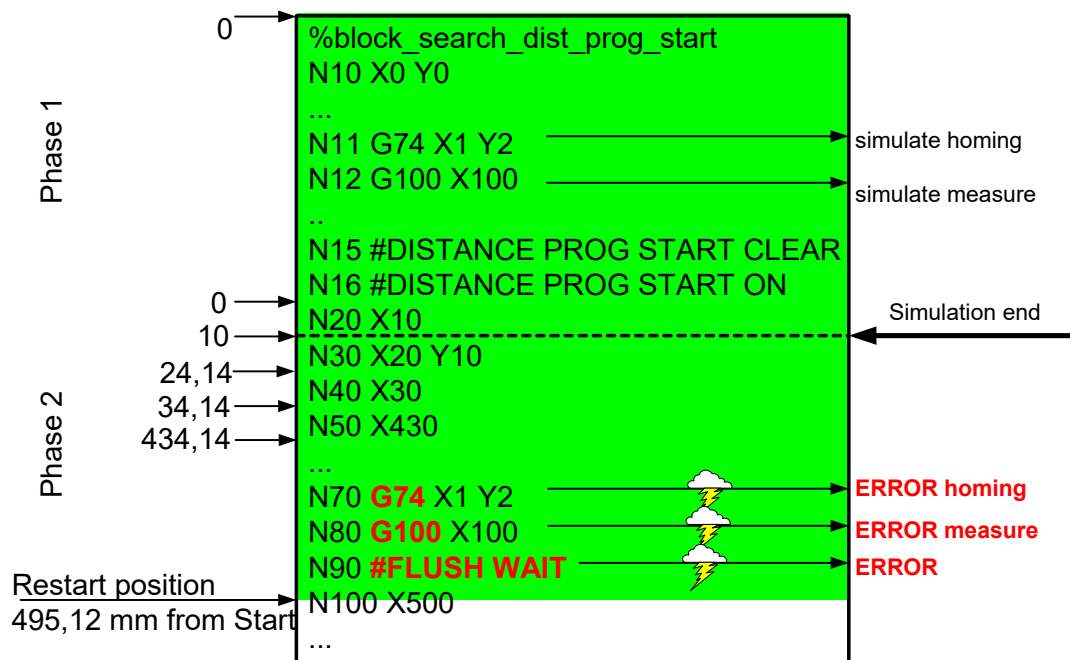


Fig. 17: G74, G100, #FLUSH WAIT during search for continuation position



## Notice

The continuation position can only be moved by tracking, i.e. towards program end, beyond the specified block limit (solid green section).

Moving the continuation position to a position already skipped in block search is not possible. This is prevented and a warning is output.





## Example

### Continuation position by distance from program start before current block

Block search type            4  
Block number                30  
Covered distance        2 mm  
Continuation position is within block N20  
Warning is output

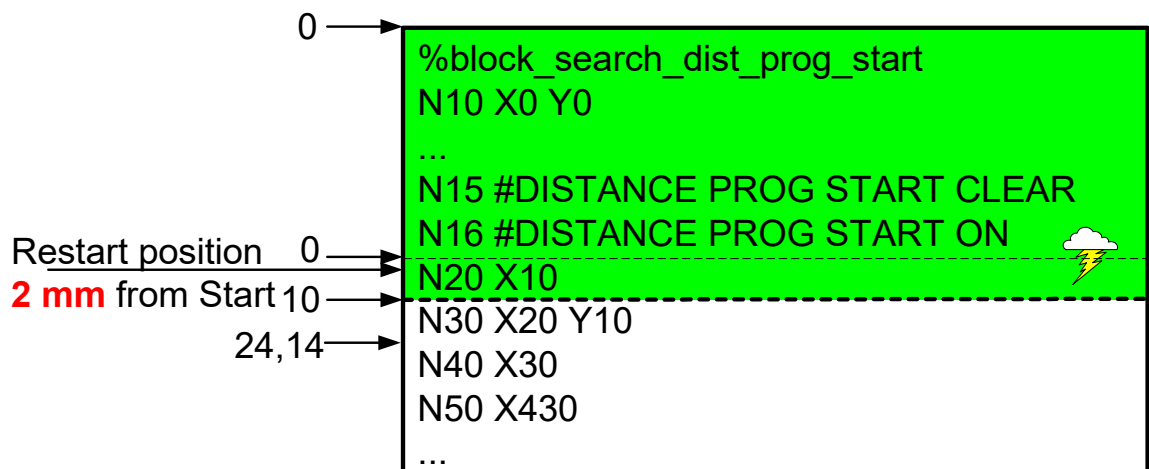


Fig. 18: Continuation position by distance from program start before current block

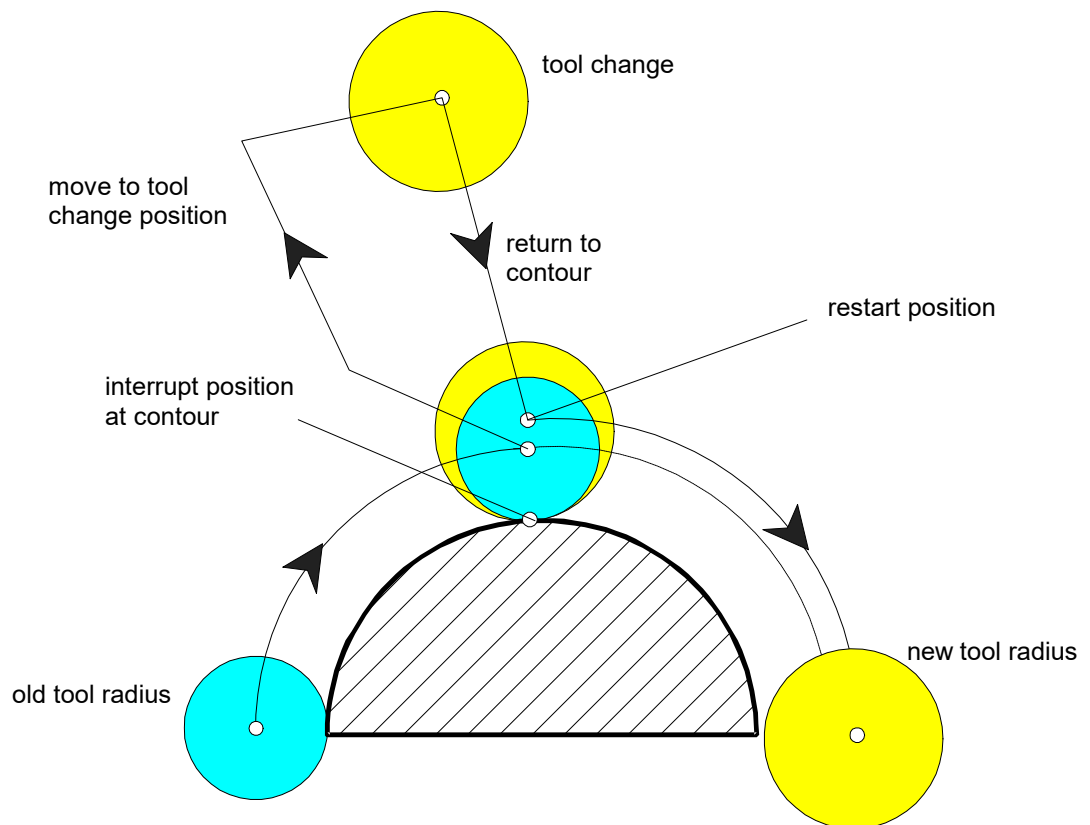
## 4.2 Restart to the contour after block search

### Move back to command position

Before normal machining can be resumed, the axes must be located back at the command positions of the NC program (restart to the contour).

This starting movement is executed either

- by an NC program specified by the operator or
- manually before start of block search or
- automatically in a straight line at the end of block search.



**Fig. 19: Use of different tool radii**

As the above figure shows, the covered distance in the current block must be specified in per thousand relative to the entire block length since the absolute block lengths are different for circular blocks with different tool radii after block search.

### Automatic restart to the contour

The automatically generated motion is moved in rapid traverse (G00).



## Example

- Operator starts normal program at position 1.
- Program interrupt at position 2 in block N20.
- Axis is moved to position 3 and the tool may be replaced (tool radius may change).
- The program context is restored at the end of block search to movement to position 4. Automatic restart occurs in a straight line.
- The program can be continued with the new tool radius in block N20.

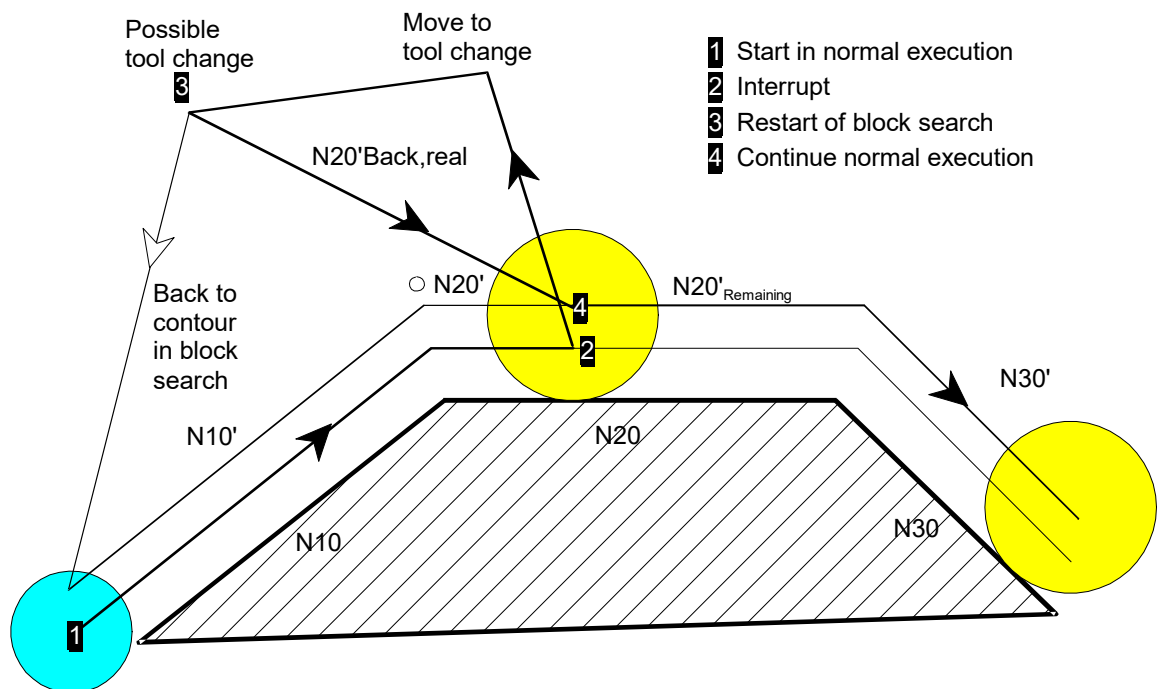


Fig. 20: Use of block search to restore the program context

### Parameterisation

Automatic restart = TRUE

mc\_cmd\_bs\_auto\_return\_w

### Manual restart to the contour

If restart to the contour is executed manually and the axes are not repositioned exactly on the contour, an offset occurs between the command positions of the NC program and the actual positions. Here, the operator can specify the maximum permitted three-dimensional offset.

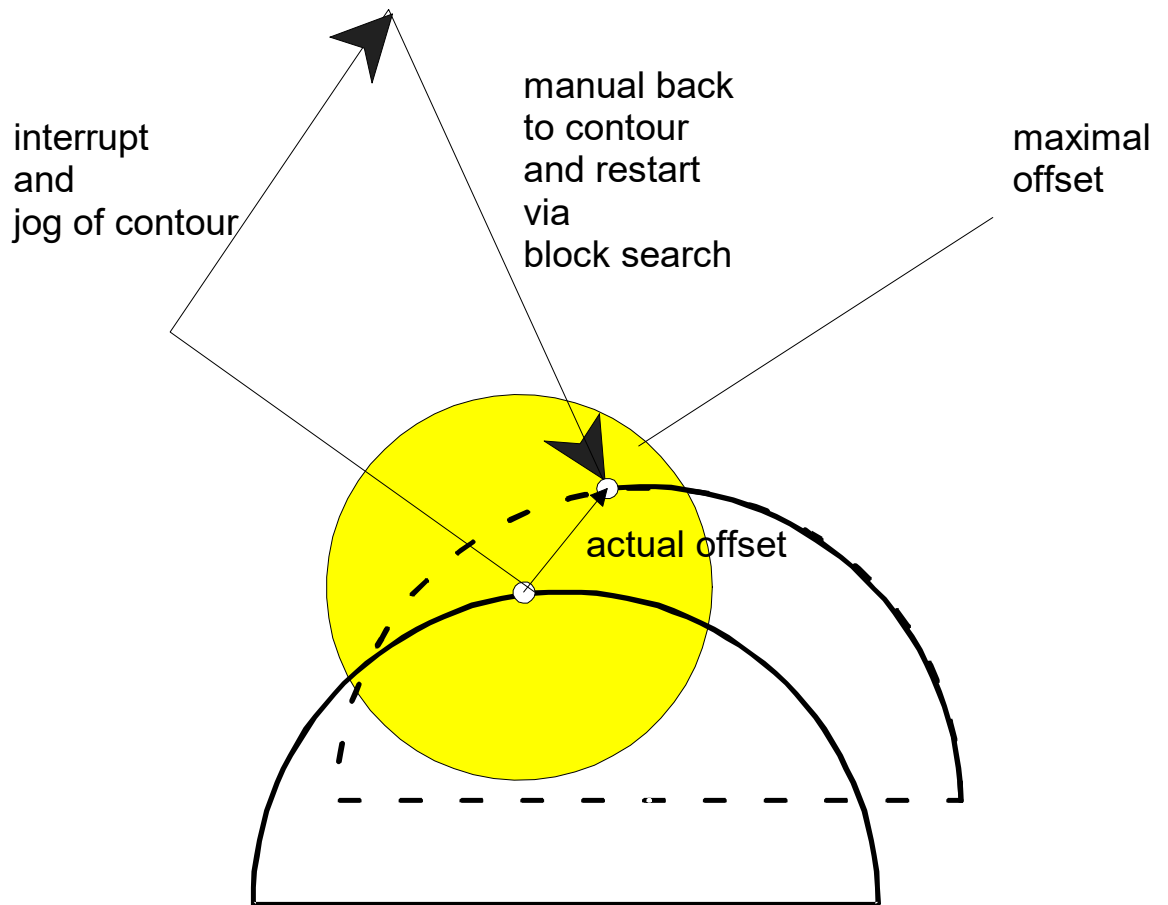


Fig. 21: Manual restart to the contour

### Parameterisation

Maximum permitted offset after block search between command position and actual position  
`mc_cmd_bs_deviation_max_w`



#### Notice

Backward motion is not possible with an offset > 0.



#### Notice

If an offset is active, please note the following in connection with software limit switch monitoring or collision monitoring:

The offset between the command positions and the actual position may result in SLS errors or collision errors although the actual positions are still within the valid range and no collision has occurred. Similarly, an SLS overshoot can not be detected in path preparation, only in the position controller. In this case, a collision cannot be detected in path preparation.

## 4.2.1 Manual restart during block search

### JogOfPath

Manual restart to the contour (see Phase (4) "Pre-return to contour") by means of an NC program can also be executed by a JogOfPath channel. Switchover between restart channel (JogOfPath) and block search channel always takes place when the axes are at standstill (see FCT-C15 Jog of path).

Block search may take longer if the program is rather large. Manual restart using a JogOfPath channel has the advantage that it can still be executed after block search has already started. In this case, it must be completed by the PLC before automatic restart to the contour is enabled (see Phase (6)).

Restart using any NC program can be used to start the contour via any strategy. By contrast, automatic restart only takes places by direct linear motion.

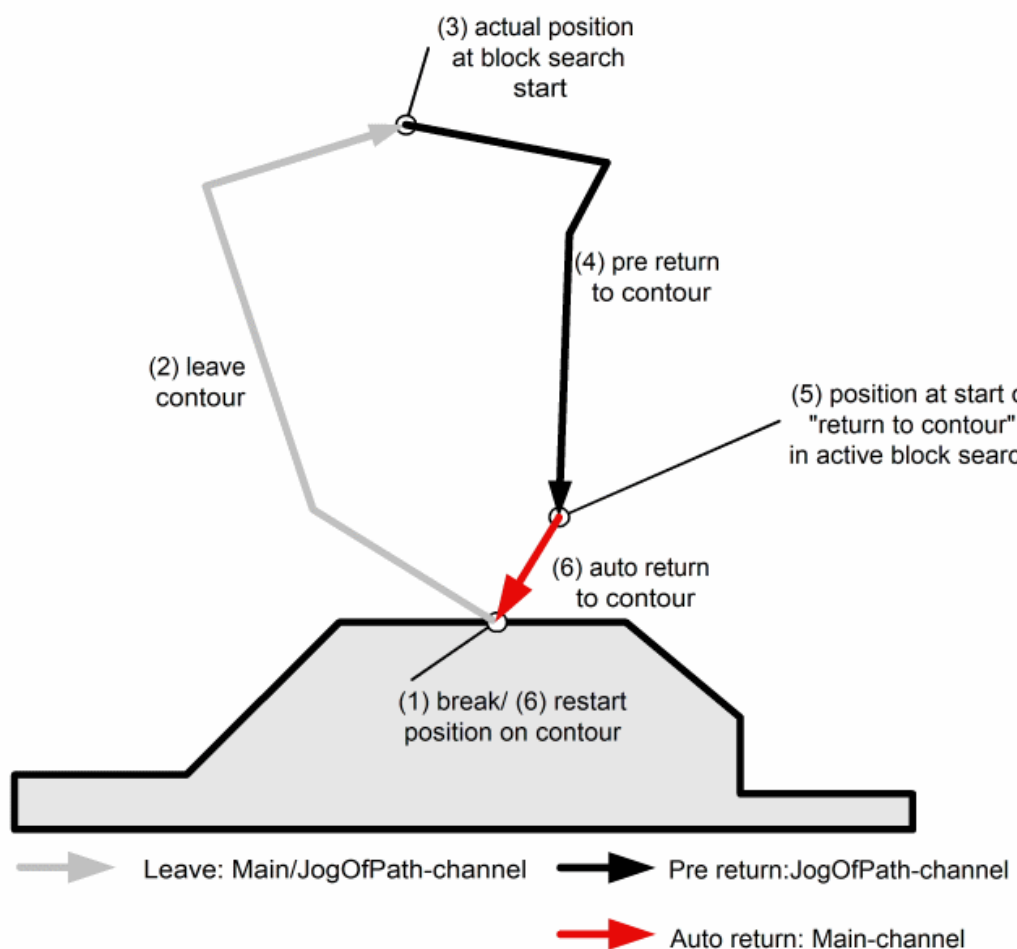


Fig. 22: Restart to the contour

### Sequence

1. Normal program start
2. Program interrupt and channel reset
3. Move to any position
4. Switchover to JogOfPath and start program in block search channel
5. Start manual start program in JogOfPath channel
6. Switchover to block search channel and enable restart by PLC
7. Automatic restart of residual path difference

**Remark:** As opposed to starting on a JogOfPath channel, manual restart in a block search channel can only take place before the block search starts.



### Notice

Channel block search can be started even though the channel axes are still released to the JogOfPath channel (PLC sets HLI.channel.SuspendAxisOutput).

The flow diagram below shows a manual restart to the contour executed by a JogOfPath channel before the PLC enables START. Before the block search channel continues, channel output is switched back to the block search channel (HLI.BlockSearchChannel.SuspendOutput = FALSE, HLI.JogOfPathChannel.SuspendOutput=TRUE).

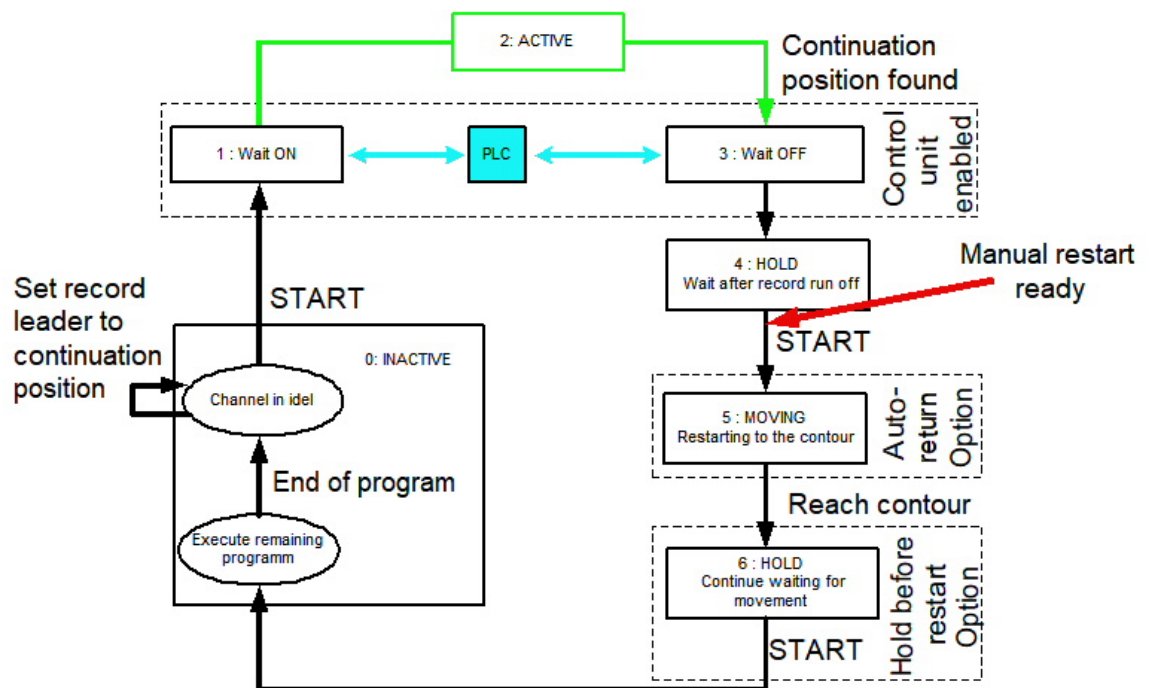


Fig. 23: Sequence of block search and completion of manual restart

## 4.2.2 Track the C axis with block search (#CAX TRACK)

### #CAX TRACK

If automatic tracking of the C axis (#CAX TRACK) is already active at the restart position of the NC program, the C axis position can be restored before automatic restart.

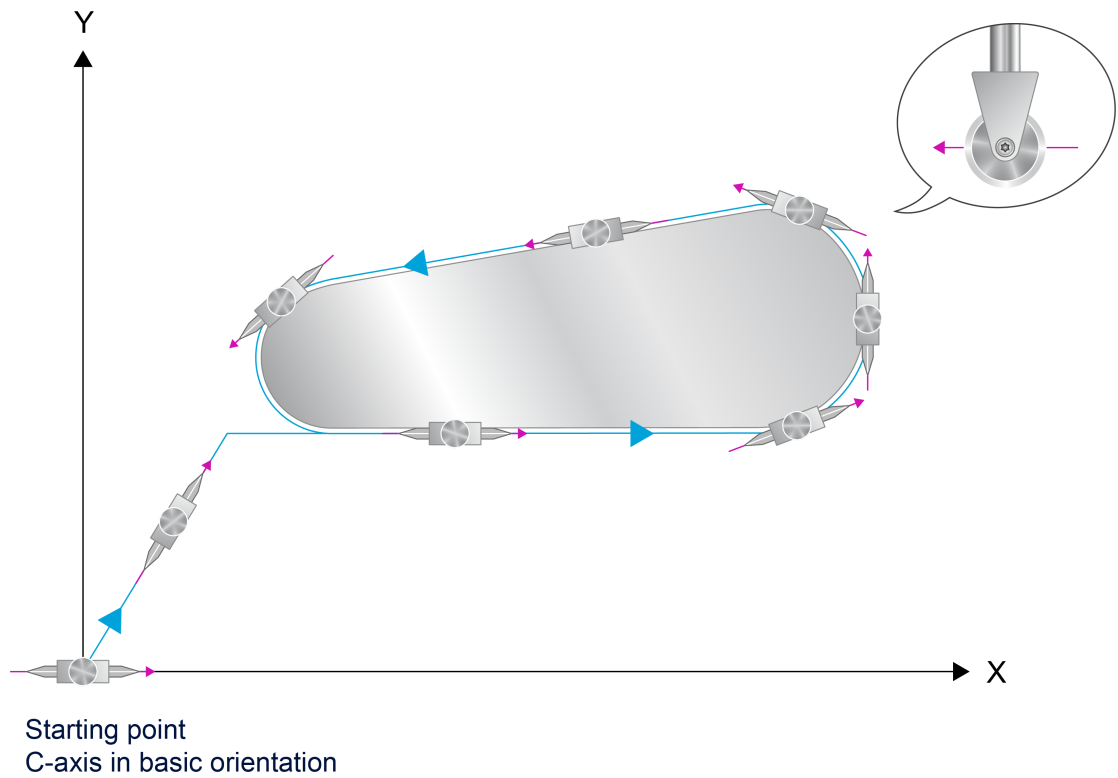


Fig. 24: Example of continuous alignment of the C axis to the contour

### The following sequence runs at restart:

1. Align the C axis according to the tangent at the restart position
2. Restart according to the block search position
3. Reactivate automatic C axis tracking
4. Wait for continue from operator/HMI/PLC



## Programing Example

### Block search to block N40 at C axis = 0°

In the example below, the C axis is first aligned in block 40 according to the tangent to C = 0 at re-start

```
%block-search-cax-track
N10 G00 G90 X0 Y0 Z0 C0
N20 X5 Y5 C45 ;straight line 45° to X axis, tracking axis
;C aligned parallel to the contour

N20 #CAXTRACK ON [ANGLIMIT 3, OFFSET 0] ;Activate axis
;tracking, limit angle 3°,
;Angular offset 0°
N30 X10 Y10 ;Primary motion block, C axis is
;already aligned
N40 X20 ;Angle to previous block: -45° >
;Limit angle -> Block is inserted:
;End position of C = 0
N60 X40 ;C axis angle 0°
N70 X30 ;C axis angle 180°
N80 Y0 ;C axis angle -90°
N90 #CAXTRACK OFF ;Deactivate axis tracking

M30
```



## Notice

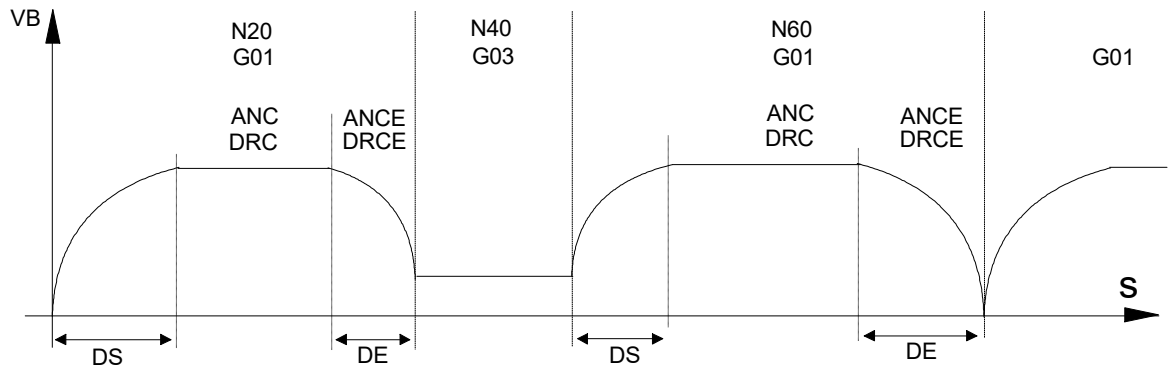
The C axis is only restarted if automatic restart to the contour is selected (see also mc\_cmd\_bs\_auto\_return\_w).



## 4.2.3 Path-dependent angle compensation at block search (#VECTOR OFFSET)

### #VECTOR OFFSET

If path-dependent angle compensation (#VECTOR OFFSET) is active at the restart position of the NC program, the compensation is produced separately at automatic restart (see also #CAX TRACK).



The following sequence runs at restart:

- (Align the C axis according to the tangent at the restart position)
- Align the angle vertically and tangentially to the contour (ANC, DRC)
- Restart according to the block search position
- Reactivate (automatic C axis tracking and) path-dependent angle compensation
- Wait for continue from operator/HMI/PLC



### Programing Example

#### Block search at N40 at ANC/DRC angle offset

In the example below, the vector offset at restart to block N40 is first set according to the active setting of block N30.

```
%block-search-vector-offset
N10 #VECTOR OFFSET ON [ DS=20 DE=15 ANC=2 DRC=3
                        ANCE=0.5 DRCE=0.2 ]

N20 G01 G90 X100 Y0 F200
N30 #VECTOR OFFSET ON [ DS=1 DE=1 ANC=0.5 DRC=0.2]
N40 G03 X110 Y10 J10 F75
N50 #VECTOR OFFSET ON [ DS=15 DE=20 ANC=2 DRC=3
                        ANCE=0.1 DRCE=0.1 ]

N60 G01 Y115
N70 G01 Y0
...
N100 #VECTOR OFFSET OFF ALL
M30
```



### Notice

The vector offset is only restarted if automatic restart to the contour is selected (see also mc\_cmd\_bs\_auto\_return\_w).

### 4.3

## Backward motion after block search

If a backward motion is executed after block search is used, the actually programmed NC program is interpolated in backward direction as of the start position from the block search.

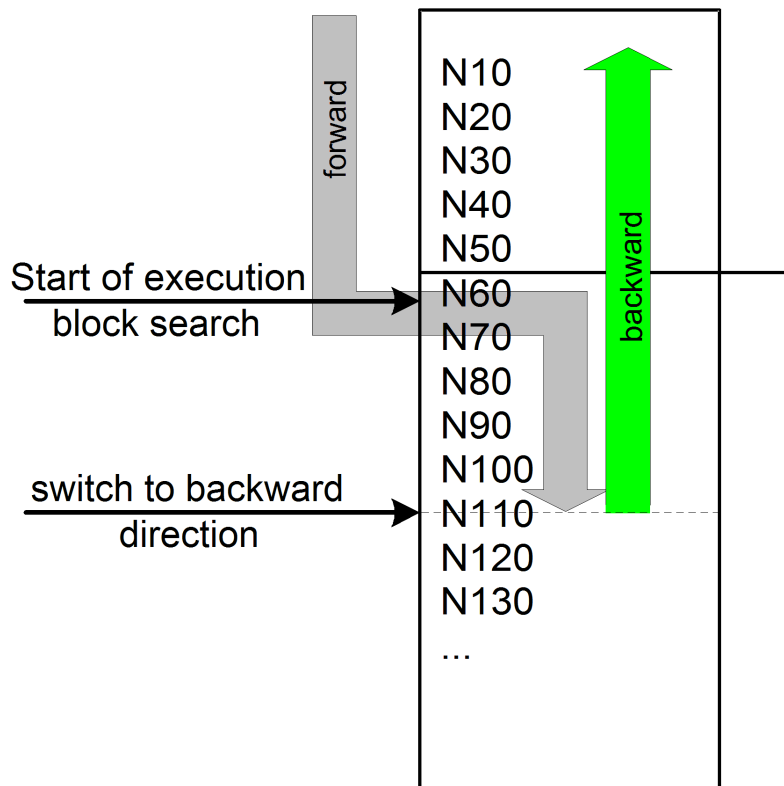


Fig. 25: Backward motion after block search

## 5 Block search interface and parameters

### Validity of block search parameters

The block search parameters (block search type, continuation position display) are defined before the start of the NC program. At program start, the current data is adopted in the start parameters. Any further change in block search parameters after program start has no more influence on the running program.

### Availability on the HMI

Assigning parameter before program start avoids time-critical situations. The parameters are available on the HMI and access is made via the PLC by read/write requests with CNC objects.

## 5.1 General block search parameters

### Block search type

<b>Name</b>	mc_cmd_bs_type_w		
<b>Description</b>	This object defines the block search type. Permissible values: 0 – no block search (default) 1 – file offset 3 – block counter 4 – block number		
<b>Task</b>	COM (Port 553)		
<b>Index group</b>	0x12010<C <sub>ID</sub> >	<b>Index offset</b>	0x49
<b>Data type</b>	UN16	<b>Length/byte</b>	2
<b>Attributes</b>	read/ write	<b>Unit</b>	-
<b>Remarks</b>			

<b>Name</b>	mc_cmd_bs_type_r		
<b>Description</b>	This object reads the block search type. Possible values: 0 – no block search (default) 1 – file offset 3 – block counter 4 – block number 5 - Program end		
<b>Task</b>	COM (Port 553)		
<b>Index group</b>	0x12010<C <sub>ID</sub> >	<b>Index offset</b>	0x61
<b>Data type</b>	UN16	<b>Length/byte</b>	2
<b>Attributes</b>	read	<b>Unit</b>	-
<b>Remarks</b>			

### Automatic resumption

<b>Name</b>	mc_cmd_bs_auto_return_w		
<b>Description</b>	This object defines that resumption of motion on the contour occurs automatically.		
<b>Task</b>	COM (Port 553)		
<b>Index group</b>	0x12010<C <sub>ID</sub> >	<b>Index offset</b>	0x4B
<b>Data type</b>	BOOLEAN	<b>Length/byte</b>	1
<b>Attributes</b>	read/ write	<b>Unit</b>	-
<b>Remarks</b>			

<b>Name</b>	mc_cmd_bs_auto_return_r		
<b>Description</b>	This object reads whether resumption of motion on the contour should occur automatically.		
<b>Task</b>	COM (Port 553)		
<b>Index group</b>	0x12010<C <sub>ID</sub> >	<b>Index offset</b>	0x4B
<b>Data type</b>	BOOLEAN	<b>Length/byte</b>	1
<b>Attributes</b>	read	<b>Unit</b>	-
<b>Remarks</b>			

## No stop on resumption

<b>Name</b>	mc_cmd_bs_no_hold_at_restart_w		
<b>Description</b>	This object defines whether resumption of motion on the contour should occur directly without any operator input.		
<b>Task</b>	COM (Port 553)		
<b>Index group</b>	0x12010<C <sub>ID</sub> >	<b>Index offset</b>	0x79
<b>Data type</b>	BOOLEAN	<b>Length/byte</b>	1
<b>Attributes</b>	read/ write	<b>Unit</b>	-
<b>Remarks</b>			

<b>Name</b>	mc_cmd_bs_no_hold_at_restart_r		
<b>Description</b>	This object reads whether resumption of motion on the contour should occur directly without any operator input.		
<b>Task</b>	COM (Port 553)		
<b>Index group</b>	0x12010<C <sub>ID</sub> >	<b>Index offset</b>	0x7A
<b>Data type</b>	BOOLEAN	<b>Length/byte</b>	1
<b>Attributes</b>	read	<b>Unit</b>	-
<b>Remarks</b>			

## Maximum path deviation

<b>Name</b>	mc_cmd_bs_deviation_max_w		
<b>Description</b>	<p>This object defines the maximum deviation of the axes between actual position and continuation position when machining is resumed after block search.</p> <p>If resumption of motion on the contour is automatic, the maximum path deviation is not considered since the exact continuation position has already been reached. (Default= 0)</p>		
<b>Task</b>	COM (Port 553)		
<b>Index group</b>	0x12010<C <sub>ID</sub> >	<b>Index offset</b>	0x4C
<b>Data type</b>	UN32	<b>Length/byte</b>	4
<b>Attributes</b>	read/ write	<b>Unit</b>	[0.1 µm]
<b>Remarks</b>			

<b>Name</b>	mc_cmd_bs_deviation_max_r		
<b>Description</b>	<p>This object reads the maximum deviation of the axes between actual position and continuation position when machining is resumed after block search.</p> <p>If restart to the contour is automatic, the maximum path deviation is not considered since reaching the exact continuation position is ensured.</p>		
<b>Task</b>	COM (Port 553)		
<b>Index group</b>	0x12010<C <sub>ID</sub> >	<b>Index offset</b>	0x64
<b>Data type</b>	UN32	<b>Length/byte</b>	4
<b>Attributes</b>	read	<b>Unit</b>	[0.1 µm]
<b>Remarks</b>			

### Program start as of file offset

<b>Name</b>	mc_command_file_offset_w		
<b>Description</b>	<p>The file offset defines a jump to a known position in the NC program. The program part before the jump point is not evaluated. Processing starts at the jump point as for a program shortened by file offset.</p> <p>This object defines the file offset. (Default value= 0)</p>		
<b>Task</b>	COM (Port 553)		
<b>Index group</b>	0x12010<C <sub>ID</sub> >	<b>Index offset</b>	0x11
<b>Data type</b>	SGN32	<b>Length/byte</b>	4
<b>Attributes</b>	write	<b>Unit</b>	-
<b>Remarks</b>	Can also be used in combination with all block search types.		

### Breakpoint

<b>Name</b>	mc_cmd_bs_breakpoint_position_w		
<b>Description</b>	This object defines an automatic breakpoint by specifying the distance from program start.		
<b>Task</b>	COM (Port 553)		
<b>Index group</b>	0x12010<C <sub>ID</sub> >	<b>Index offset</b>	0x7B
<b>Data type</b>	REAL64	<b>Length/byte</b>	8
<b>Attributes</b>	read/ write	<b>Unit</b>	[0.1 µm]
<b>Remarks</b>	Can also be used in combination with all block search types.		

<b>Name</b>	mc_cmd_bs_breakpoint_position_r		
<b>Description</b>	This object reads an automatic breakpoint by specifying the distance from program start.		
<b>Task</b>	COM (Port 553)		
<b>Index group</b>	0x12010<C <sub>ID</sub> >	<b>Index offset</b>	0x11
<b>Data type</b>	REAL64	<b>Length/byte</b>	8
<b>Attributes</b>	read	<b>Unit</b>	[0.1 µm]
<b>Remarks</b>	Can also be used in combination with all block search types.		

## 5.1.1 Covered distance

### Motion block in per mil

<b>Name</b>	mc_cmd_bs_distance_prog_start_w		
<b>Description</b>	This object defines the distance from program start or #DISTANCE PROG START CLEAR at which machining is actually supposed to start.		
<b>Task</b>	COM (Port 553)		
<b>Index group</b>	0x12010<C <sub>ID</sub> >	<b>Index offset</b>	0x44
<b>Data type</b>	REAL64	<b>Length/byte</b>	8
<b>Attributes</b>	read/ write	<b>Unit</b>	[0.1 µm]
<b>Remarks</b>	.		

<b>Name</b>	mc_cmd_bs_distance_prog_start_r		
<b>Description</b>	This object defines the distance from program start or DISTANCE PROG START CLEAR at which machining is actually supposed to start.		
<b>Task</b>	COM (Port 553)		
<b>Index group</b>	0x12010<C <sub>ID</sub> >	<b>Index offset</b>	0x45
<b>Data type</b>	REAL64	<b>Length/byte</b>	8
<b>Attributes</b>	read	<b>Unit</b>	[0.1 µm]
<b>Remarks</b>	.		

### From program start - path increments

<b>Name</b>	mc_cmd_bs_covered_distance_w		
<b>Description</b>	This object defines the distance covered in the NC block in <i>per mil</i> at which machining is actually supposed to continue. The first part of the block in the block search is then executed without motion and only the remaining part is executed with moved axes. Value range: [0.0 to 1000.0]; default value= 0.0		
<b>Task</b>	COM (Port 553)		
<b>Index group</b>	0x12010<C <sub>ID</sub> >	<b>Index offset</b>	0x4A
<b>Data type</b>	REAL64	<b>Length/byte</b>	8
<b>Attributes</b>	read/ write	<b>Unit</b>	[0.1%]
<b>Remarks</b>	.		

<b>Name</b>	mc_cmd_bs_covered_distance_r		
<b>Description</b>	This object defines the distance covered in the NC block in <i>per mil</i> at which machining is actually supposed to continue. The first part of the block in the block search is then executed without motion and only the remaining part is executed with moved axes. Value range: [0.0 to 1000.0]; default value= 0.0		
<b>Task</b>	COM (Port 553)		
<b>Index group</b>	0x12010<C <sub>ID</sub> >	<b>Index offset</b>	0x62
<b>Data type</b>	REAL64	<b>Length/byte</b>	8
<b>Attributes</b>	read	<b>Unit</b>	[0.1%]
<b>Remarks</b>	.		



## 5.2 Block search type 1 parameters: File start/end position

### 5.2.1 Start position

The following parameters can be accessed via CNC objects to define the start position.

- File offset
- File name
- Program path ID
- Pass counter

They are described below.

File offset	
Description	Defines the file offset at which point processing is to continue.
Type	SGN32
Value range	[0, MAX_SGN32]
HMI elements	mc_cmd_bs_pos_start_offset_w (write) mc_cmd_bs_pos_start_offset_r (read)
IndexOffset	0x50 (write) 0x68 (read) (IndexGroup = 0x000201<ii> where <ii> = channel)

File name	
Description	Specifies the file which contains the start position specified by file offset. This means that actual processing continues at this point.
Type	STRING
Value range	[0, 83] – maximum 84 characters
HMI elements	mc_cmd_bs_pos_start_name_w (write) mc_cmd_bs_pos_start_name_r (read)
IndexOffset	0x4F (write) 0x67 (read) (IndexGroup = 0x000201<ii> where <ii> = channel)



#### Attention

The specified block search filename must be identical with the file started in automatic mode. It also contains any additional specified path.

If an explicit path is specified at program start, the path must also be included in the block search when the filename is specified.

<b>Program path ID</b>	
Description	The identifier indicates whether the NC program defined by the filename is located on the main program path (HP) or the subroutine path (UP).
Type	UNS16
Value range	0 - HP (default); 1 - UP
HMI elements	mc_cmd_bs_pos_start_type_w (write) mc_cmd_bs_pos_start_type_r (read)
IndexOffset	0x4E (write) 0x66 (read) (IndexGroup = 0x000201<ii> where <ii> = channel)

<b>Pass counter start position</b>	
Description	The pass counter determines the number of times the start position should be passed up to the final start, e.g. in loops.
Type	SGN16
Value range	[ 1; MAX_SGN16 ], default 1
HMI elements	mc_cmd_bs_pos_start_count_w (write) mc_cmd_bs_pos_start_count_r (read)
IndexOffset	0x51 (write) 0x69 (read) (IndexGroup = 0x000201<ii> where <ii> = channel)

## 5.2.2 End position

The following parameters can be accessed via CNC objects to define the end position.

- File offset
- File name
- Program path ID
- Pass counter

They are described below.

File offset	
Description	Defines the file offset at which point processing is to end.
Type	SGN32
Value range	[0, MAX_SGN32]
HMI elements	mc_cmd_bs_pos_end_offset_w (write) mc_cmd_bs_pos_end_offset_r (read)
IndexOffset	0x54 (write) 0x6C (read) (IndexGroup = 0x000201<ii> where <ii> = channel)

File name	
Description	Specifies the file which contains the end position specified by file offset. This means that actual processing end at this point.
Type	STRING
Value range	[0, 83] – maximum 84 characters
HMI elements	mc_cmd_bs_pos_end_name_w (write) mc_cmd_bs_pos_end_name_r (read)
IndexOffset	0x53 (write) 0x6B (read) (IndexGroup = 0x000201<ii> where <ii> = channel)



### Attention

The specified block search filename must be identical with the file started in automatic mode. It also contains any additional specified path.

If an explicit path is specified at program start, the path must also be included in the block search when the filename is specified.

<b>Program path ID</b>	
Description	The identifier indicates whether the NC program defined by the filename is located on the main program path (HP) or the subroutine path (UP).
Type	UNS16
Value range	0 - HP (default); 1 - UP
HMI elements	mc_cmd_bs_pos_end_type_w (write) mc_cmd_bs_pos_end_type_r (read)
IndexOffset	0x52 (write) 0x6A (read) (IndexGroup = 0x000201<ii> where <ii> = channel)

<b>Pass counter end position</b>	
Description	The pass counter determines the number of times the end position should be passed up to the end of machining, e.g. in loops.
Type	SGN16
Value range	[ 1; MAX_SGN16 ], default 1
HMI elements	mc_cmd_bs_pos_end_count_w (write) mc_cmd_bs_pos_end_count_r (read)
IndexOffset	0x55 (write) 0x6D (read) (IndexGroup = 0x000201<ii> where <ii> = channel)

### 5.3 Block search type 3 parameters: Block counter

<b>Block counter</b>	
Description	Defines the block counter at which point actual machining is to continue.
Type	UNS32
Value range	[ 1; MAX_UN32 ]
HMI elements	mc_cmd_bs_intern_block_count_w (write) mc_cmd_bs_intern_block_count_r (read)
IndexOffset	0x5e (write) 0x76 (read) (IndexGroup = 0x000201<ij> where <ij> = channel)

### 5.4 Block search type 4 parameters: Block number

<b>Block number</b>	
Description	Defines the block number at which point actual machining is to continue.
Type	UNS32
Value range	[0, MAX_UN32]
HMI elements	mc_cmd_bs_block_number_w (write) mc_cmd_bs_block_number_r (read)
IndexOffset	0x5F (write) 0x77 (read) (IndexGroup = 0x000201<ij> where <ij> = channel)

<b>Pass counter block number</b>	
Description	If the identical block number is passed several times (e.g. in loops), the operator can specify in the pass counter the number of passes after which machining should start.
Type	UNS32
Value range	[ 1; MAX_UN32 ], default 1
HMI elements	mc_cmd_bs_block_number_pass_w (write) mc_cmd_bs_block_number_pass_r (read)
IndexOffset	0x60 (write) 0x78 (read) (IndexGroup = 0x000201<ij> where <ij> = channel)

<b>Name</b>	mc_cmd_bs_block_nbr_prog_name_w		
<b>Description</b>	<p>This CNC object defines the NC programming name where the specified block number should lie when block search type 4 is used (block search for block number).</p> <p>When using subroutine techniques and the block number also used there, specifying the NC program name can be used to uniquely identify the block search position.</p>		
<b>Task</b>	COM (Port 553)		
<b>Index group</b>	0x12010<C <sub>ID</sub> >	<b>Index offset</b>	0xAF
<b>Data type</b>	String	<b>Length/byte</b>	84
<b>Attributes</b>	write	<b>Unit</b>	-
<b>Remarks</b>			

<b>Name</b>	mc_cmd_bs_block_nbr_prog_name_r		
<b>Description</b>	<p>This CNC object reads the NC program name assigned by the CNC object mc_cmd_bs_block_nbr_prog_name_w [▶ 46].</p>		
<b>Task</b>	COM (Port 553)		
<b>Index group</b>	0x12010<C <sub>ID</sub> >	<b>Index offset</b>	0xB0
<b>Data type</b>	String	<b>Length/byte</b>	84
<b>Attributes</b>	read	<b>Unit</b>	-
<b>Remarks</b>			

## 5.5 Status data: Access via CNC objects:

<b>Name</b>	distance from program start		
<b>Description</b>	Distance covered from program start or from the last NC command #DISTANCE PROG START CLEAR.		
<b>Task</b>	GEO (Port 551)		
<b>Index group</b>	0x12130<C <sub>ID</sub> >	<b>Index offset</b>	0x56
<b>Data type</b>	REAL64	<b>Length</b>	8
<b>Attributes</b>	read	<b>Unit</b>	[-]
<b>Remarks</b>	The value is specified in increments.		

## 5.6 Status data: Access via HLI

The PLC can access the data listed below via the HLI.

<b>Block search active</b>	
<b>Description</b>	The interpolator works in block search mode. No axis motion occurs. As long the block search process in the interpolator remains in the HLI_BS_ACTIVE or HLI_BS_WAIT_FOR_PLC_OFF state, this value indicates TRUE.
<b>Signal flow</b>	CNC → PLC
<b>ST path</b>	gpCh[channel_idx]^bahn_state.block_search_active_r
<b>Data type</b>	BOOL
<b>Value range</b>	[TRUE = active - Interpolator works in block search mode., FALSE]
<b>Access</b>	PLC is reading

Block search, state		
Description	Indicates the current state of the block search mode in the interpolator.	
Signal flow	CNC → PLC	
ST path	gpCh[channel_idx]^bahn_state.block_search_state_r	
Data type	INT	
Value range	<b>Constant</b>	<b>Value</b>
	HLI_BS_INACTIVE	0
	HLI_BS_WAIT_FOR_PLC_ON	1
	HLI_BS_ACTIVE	2
	HLI_BS_WAIT_FOR_PLC_OFF	3
	HLI_BS_WAIT_RETURN_TO_CONTOUR	4
	HLI_BS_RETURNING_TO_CONTOUR	5
	HLI_BS_WAIT_FOR_CONTINUE_CONTOUR	6
Access	PLC is reading	

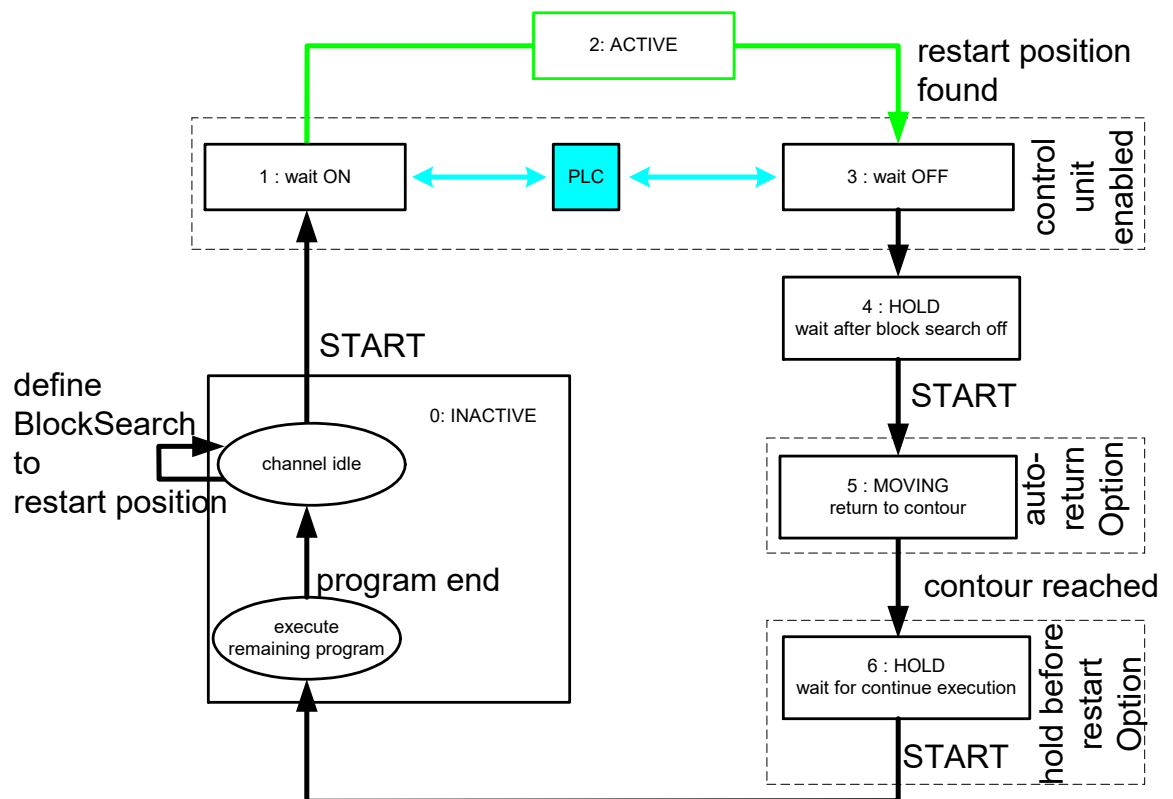
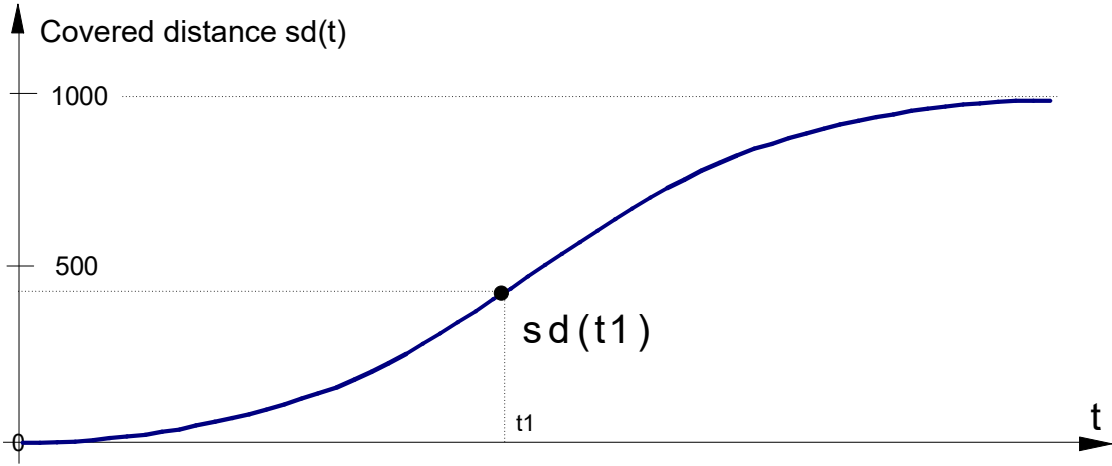


Fig. 26: Block search states



<b>Covered block motion path</b>	
Description	<p>Part of the path motion traversed in the current block in relation to the total path.</p> <p>This status datum contains the current block position referred to the path distance in space in the motion block in per mil <math>sd(t)</math>.</p> 
Signal flow	CNC → PLC
ST path	<code>gpCh[channel_idx]^bahn_state.covered_distance_r</code>
Data type	DINT
Unit	0.1 %
Access	PLC is reading
Special features	If a main axis participates in the motion, the covered path motion is in relation to the block path of the first three axes. If no main axis participates in the motion, the covered path motion is the position lag with the longest motion time in relation to the block path.

<b>Currently covered path in the NC program (PCS)</b>	
Description	Reads the current distance covered in the NC program since program start or since the last # DISTANCE PROG START CLEAR NC command. The calculation is based on the current position in the current NC block.
Signal flow	CNC → PLC
ST path	<code>gpCh[channel_idx]^bahn_state.dist_prog_start</code>
Data type	UDINT (* LREAL)
Unit	0.1 $\mu\text{m}$
Access	PLC is reading
Special features	* As of CNC Build V3.1.3104.01 the data element is provided in LREAL forma.

<b>Line counter, NC program</b>	
Description	<p>The datum indicates the NC program line which is the source of the command just processed by the interpolator.</p> <p>The value is derived from the number of NC program lines which the decoder has read since the NC program started. All the lines read the decoder are counted, i.e. repeatedly read lines, empty and comment lines. All commands to the interpolator resulting from decoding a NC program line are assigned to the associated line counter.</p>
Signal flow	CNC → PLC
ST path	gpCh[channel_idx]^bahn_state.block_count_r
Data type	UDINT
Access	PLC is reading

<b>Block search, distance from continuation position</b>	
Description	<p>If a NC program is started in block search modus, the NC program is processed in simulative mode (without axis motion) until the continuation position is reached. Block search is then in the HLI_BS_WAIT_FOR_PLC_OFF state and calculates the distance between the actual positions of the axis and the continuation position. If block search is in the HLI_BS_RETURNING_TO_CONTOUR state, this value is refreshed cyclically.</p>
Signal flow	CNC → PLC
ST path	gpCh[channel_idx]^bahn_state.block_search_path_deviation_r
Data type	UDINT
Unit	0.1 µm
Value range	[0, MAX_SGN32]
Access	PLC is reading

<b>Stop condition</b>	
Description	Displays the condition why the current motion was stopped.
Signal flow	CNC → PLC
ST path	gpCh[channel_idx]^bahn_state.stop_conditions_r
Data type	DINT
Value range	See Value range of stop conditions [▶ 51] with explanations.
Access	PLC is reading

**Value range of stop conditions**

Constant in PLC	Value	Explanation
HLI_SC_FEEDHOLD	0x0001	Path feed stop
HLI_SC_VFG	0x0002	No axis-specific feed enable.
HLI_SC_SINGLE_BLOCK	0x0004	Single step mode active.
HLI_SC_M00_OR_M01	0x0010	M00 (programmed stop), M01 (optional stop) is active.
HLI_SC_PLC_ACKNOWLEDGE	0x0020	Stop occurs due to waiting for an acknowledgement from the SPS. This may occur as a result of the output of M or H technology functions but is not restricted to them alone.
HLI_SC_OVERRIDE_ZERO	0x0040	Override = 0.
HLI_SC_OVERRIDE_RAPID_ZERO	0x0080	Override = 0 with rapid traverse blocks
HLI_SC_DELAY_TIME	0x0200	Dwell time.
HLI_SC_CHANNEL_SYNC	0x0800	Channel synchronisation is active.
HLI_SC_IPO_INPUT_EMPTY	0x1000	Input FIFO of the interpolation is empty.
HLI_SC_IPO_INPUT_DISABLED	0x2000	Input of function blocks (e.g. motion blocks etc.) disabled.
HLI_SC_WAIT_FOR_AXES	0x8000	Stop occurs due to waiting until a commanded axis swap is completed.
HLI_SC_CHANNEL_ERROR	0x00010000	An error occurred in the channel.
HLI_SC_WAIT_TECHNO_ACK	0x00020000	Waiting for acknowledgement of M/H/ST technology functions.
HLI_SC_W_C_AFTER_COLLISION	0x00040000	After a detected collision, waiting for motion resumption.
HLI_SC_SLOPE_SUPPLY_PROBLEM	0x00080000	Block supply problem (only occurs in conjunction with HSC slope).

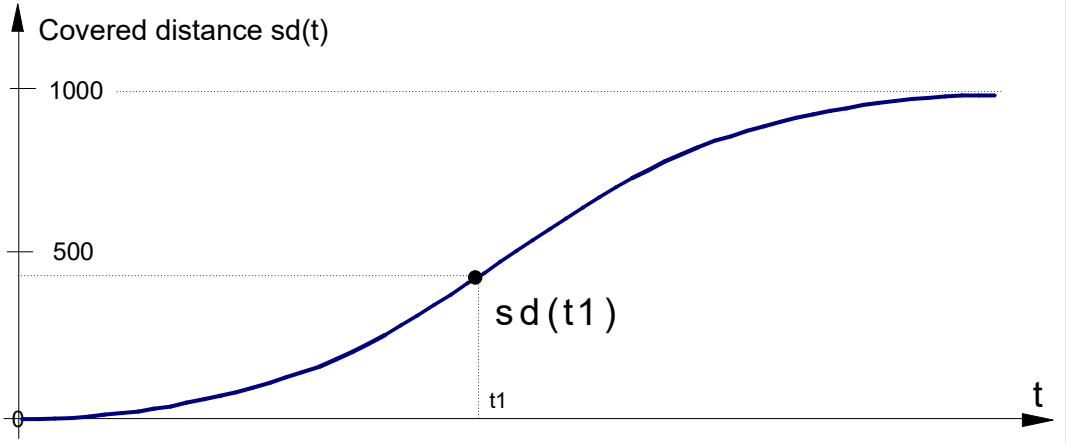
HLI_SC_BACK_INTERPOLATION	0x00100000	Back interpolation after tracking mode is active.
HLI_SC_STOP_REVERSIBLE	0x00200000	Stop since M00 (programmed stop) is active. However, the NC program can be processed backwards despite M00 (available as of V3.1.3039.01).
HLI_SC_BREAKPOINT_STOP	0x00400000	Stop after a breakpoint (stop point) is reached; available as of V3.1.3039.01.
HLI_SC_M0_STOP	0x02000000	Stop after an M00 function is reached
HLI_SC_M1_STOP	0x04000000	Stop after an M01 function is reached
HLI_SC_INSERT_STOP_AT_DIST	0x08000000	Stop after an M function inserted by the Control Unit "Inserting stop marks" is reached.
HLI_SC_DEC_SYN_CHAN_EMPTY	0x10000000	Decoder is waiting for synchronisation. NC channel has no jobs.



## 5.6.1 HL access with CNC Version < V2.11.28xx

<b>Block search active</b>	
Description	The interpolator works in block search mode.
Signal flow	CNC → PLC
ST path	pMC[channel_idx]^addr^.StateBahn_Data.X_BlockSearchActive
Data type	BOOL
Value range	[TRUE = active - Interpolator works in block search mode., FALSE]
Access	PLC is reading

<b>Block search, state</b>		
Description	Indicates the current state of the block search mode in the interpolator.	
Signal flow	CNC → PLC	
ST path	pMC[channel_idx]^addr^.StateBahn_Data.W_BlockSearchState	
Data type	INT	
Value range	<b>Constant</b>	<b>Value</b>
	HLI_BS_INACTIVE	0
	HLI_BS_WAIT_FOR_PLC_ON	1
	HLI_BS_ACTIVE	2
	HLI_BS_WAIT_FOR_PLC_OFF	3
	HLI_BS_WAIT_RETURN_TO_CONTOUR	4
	HLI_BS_RETURNING_TO_CONTOUR	5
	HLI_BS_WAIT_FOR_CONTINUE_CONTOUR	6
Access	PLC is reading	

<b>Covered block motion path</b>	
Description	<p>Part of the path motion traversed in the current block in relation to the total path.</p> <p>This status datum contains the current block position referred to the path distance in space in the motion block in per mil <math>sd(t)</math>.</p> 
Signal flow	CNC → PLC
Unit	0.1 %
ST path	<code>pMC[channel_idx]^addr^.StateBahn_Data.D_CoveredDistance</code>
Data type	DINT
Access	PLC is reading
Special features	If a main axis participates in the motion, the covered path motion is in relation to the block path of the first three axes. If no main axis participates in the motion, the covered path motion is the position lag with the longest motion time in relation to the block path.

<b>Currently covered path in the NC program (PCS)</b>	
Description	<p>Reads the currently covered path in the NC program since program start or since the last <code>#DISTANCE PROG START CLEAR NC</code> command. The calculation is based on the current position in the current NC block.</p>
Signal flow	CNC → PLC
Unit	0.1 $\mu\text{m}$
ST path	<code>pMC[channel_idx]^addr^.StateBahn_Data.D_DistProgStartHigh</code> <code>pMC[channel_idx]^addr^.StateBahn_Data.D_DistProgStartLow</code>
Data type	UDINT
Access	PLC is reading
Special features	<p>In the NC this is an integer number which occupies 8 bytes in the memory. At the HLI the number is provided in the form of two 4-byte wide values. The value in <code>D_DistProgStartLow</code> contains the 4 lower bytes 0 to 3 and the value in <code>D_DistProgStartHigh</code> contains the higher bytes 4 to 7 of the 8-byte value present in the NC kernel.</p> <p>The read value can be used to command the block search to define the covered path in the NC program from where actual machining should effectively start.</p>

<b>Line counter, NC program</b>	
Description	<p>The datum indicates the NC program line which is the source of the command just processed by the interpolator.</p> <p>The value is derived from the number of NC program lines which the decoder has read since the NC program started. All the lines read the decoder are counted, i.e. repeatedly read lines, empty and comment lines. All commands to the interpolator resulting from decoding a NC program line are assigned to the associated line counter.</p>
Signal flow	CNC → PLC
ST path	pMC[channel_idx]^addr^.StateBahn_Data. <b>D_BlockCount</b>
Data type	UDINT
Access	PLC is reading

<b>Block search, distance from continuation position</b>	
Description	<p>If a NC program is started with block search mode, the NC program is processed in simulative mode (with no path motion) up to the specified continuation position. Block search is then in the HLI_BS_WAIT_FOR_PLC_OFF state and calculates the distance between the actual positions of the axis and the continuation position. If block search is in the HLI_BS_RETURNING_TO_CONTOUR state, this value is refreshed cyclically.</p>
Signal flow	CNC → PLC
ST path	pMC[channel_idx]^addr^.StateBahn_Data. <b>D_BlockSearchPathDeviation</b>
Unit	0.1 µm
Data type	UDINT
Value range	[0, MAX_SGN32]
Access	PLC is reading

<b>Stop condition</b>	
Description	Displays the condition why the current motion was stopped.
Signal flow	CNC → PLC
ST path	pMC[channel_idx]^addr^.StateBahn_Data. <b>D_StopConditions</b>
Data type	DINT
Value range	See table: Value range of stop conditions
Access	PLC is reading

## 6 Lock program areas for block search (#BLOCKSEARCH)

The command

#BLOCKSEARCH LOCKED/RELEASED can lock any program areas for block search in the NC program. If the start position of block search is then located in one of these locked areas, the error message P-ERR-21399 is output.

The block search lock also includes all local and global subroutine called in the corresponding area.

If locked areas are nested, the block search lock includes the area starting at the first activation up to the first deactivation (see example 2).

#BLOCKSEARCH LOCKED | RELEASED

modal



### Programming Example

#### Lock program areas for block search

##### Example 1:

No start position may be selected for block search in the range of NC blocks N40–N100 including the subroutines called in it.

```
%BLOCKSEARCH
N10 X0 Y0 Z0
N20 X10
N30 Y10
N40 #BLOCKSEARCH LOCKED
N50 X20
N60 Y20
N65 L GSP.nc
N70 Z20
N80 X30
N90 Z30
N100 #BLOCKSEARCH RELEASED
N110 Y30
N120 X40
N130 Z40
N999 M30
```



**Example 2:**

The area of the block search lock when nested covers N40-N75.

```
%BLOCKSEARCH
N10 X0 Y0 Z0
N20 X10
N30 Y10
N40 #BLOCKSEARCH LOCKED
N50 X20
N55 #BLOCKSEARCH LOCKED
N60 Y20
N65 L GSP.nc
N70 Z20
N75 #BLOCKSEARCH RELEASED
N80 X30
N90 Z30
N100 #BLOCKSEARCH RELEASED
N110 Y30
N120 X40
N130 Z40
N999 M30
```

## 7 On/off handshake with PLC

While block search is active, the PLC is notified of every technology M function (e.g. torch on/off). As opposed to normal mode, however, this is not executed. Instead this is handled in the PLC by a special module. The PLC is then notified of every change in status of block search mode (on/off). The PLC then acknowledges the change and sends it to the NC: The NC waits for the acknowledgement from the PLC in the same way as for synchronised M functions.

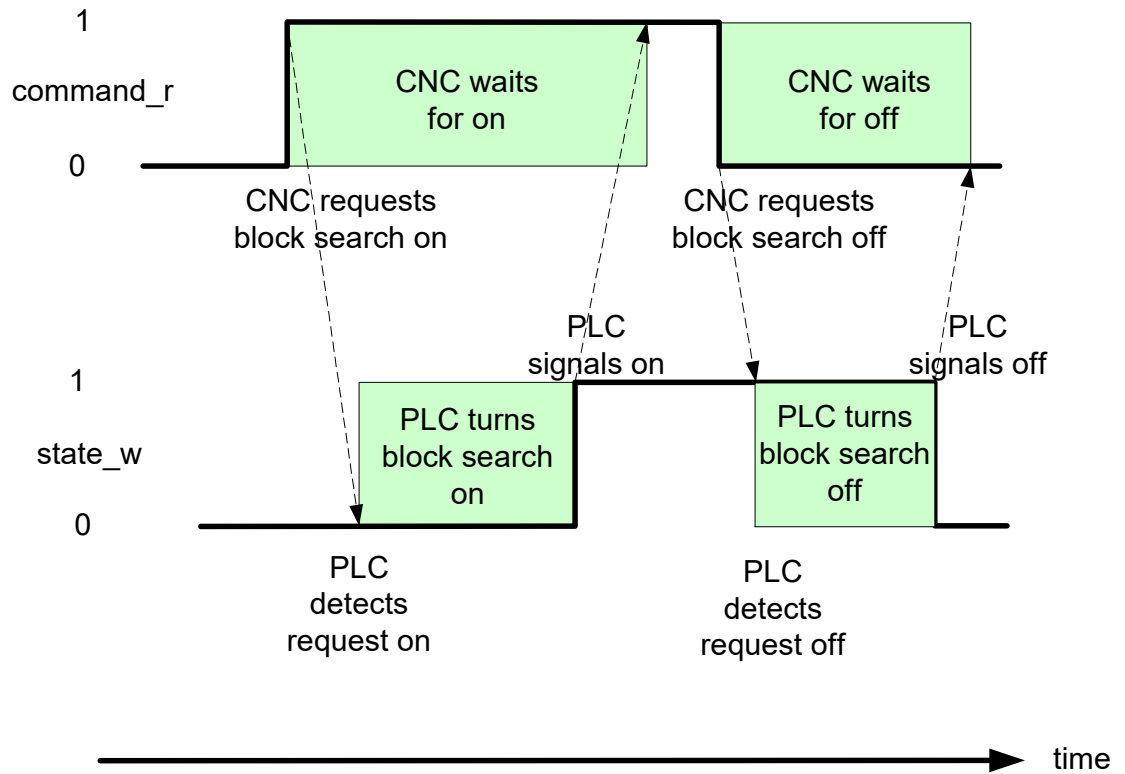
If block search is active during an NC reset, block search without handshake is deselected (the PLC should then be notified of the reset by the reset-specific control unit).

<b>Block search on/off to PLC</b>	
Description	<p>At every block search on/off request, the CNC initiates a handshake with the PLC;</p> <p>As long as the request element [▶ 58] has the value TRUE, the PLC is notified of every change in block search mode.</p>
Access	<p>If an NC program starts with block search mode activated, the CNC sets the CNC signal [▶ 58] to TRUE and waits for the PLC signal [▶ 58], indicating that the PLC is ready for block search.</p> <p>When the PLC has executed the necessary actions to prepare for block search, it notifies this to the CNC by setting the PLC signal [▶ 58] to TRUE.</p> <p>After this signal, the NC program can be processed in block search mode. This takes place either by the "Continue motion" or "Program start" commands.</p> <p>If the re-engagement position is reached while the NC program is processed, the CNC signals this by setting the CNC signal [▶ 58] to FALSE.</p> <p>The PLC detects this, completes its preparations for operation with real axis motions and then sets the PLC signal [▶ 58] to FALSE.</p>

<b>ST path, ST element for CNC Build &gt;= 2800</b>	
ST path	gpCh[channel_idx]^bahn_lc_control.block_search with channel_idx = [0, HLI_SYS_CH_MAXIDX]
Data type	LC_CONTROL_BOOL_UNIT
CNC signal	
ST element	<b>.command_r</b>
Data type	BOOL
Value range	[TRUE = NC program was started in block search mode, FALSE = block search OFF]
PLC signal	
ST element	<b>.state_r</b>
Data type	BOOL
Value range	[TRUE = PLC acknowledged the notification about block search activation, FALSE = PLC acknowledged the notification of block search deactivation]
Request	
ST element	<b>.enable_w</b>
Data type	BOOL
Value range	[TRUE = PLC wants to be notified about activation of block search, FALSE]

<b>ST path, ST element for CNC Build &gt; 2800</b>	
ST path	pMC[channel_idx]^addr^.LCControlBahn_Data.LCControlBoolUnit_BlockSearch with channel_idx = [1, HLI_SYS_CHNMAX]
Data type	LCControlBoolUnit
CNC signal	
ST element	<b>.X_Command</b>
PLC signal	
ST element	<b>.X_State</b>
Request	
ST element	<b>.X_Enable</b>

## wait for block search on / off



**Fig. 27: Interaction between BOOLEAN-LC control unit and PLC**



### Notice

When the CNC is reset, the CNC resets the CNC signal and the PLC signal.

## 8 Known restrictions

### Offset

If the program continues with an offset after block search since it was not fully restarted on the contour, this offset remains valid up to program end and reset. When the program is restarted, the motion continues without offset.

### Motion and technology in starting block

If an NC block intended for restart receives motion and technology commands (e.g. N100), only the remaining motion commands are actually executed. All technology commands are only simulated in block N100 (block search mode).



#### Programing Example

##### Motion and technology in starting block

```
N90 X90  
N100 X100 S1000 M3 M111  
N110 ...
```

If technology commands also require to be executed in real motion, the operator may only continue to move in block search up to the end of block N90 with a covered distance of 100%. This refers to the above example.

### Automatic restart

Please note the points below for automatic restart to the contour:

1. Axes are moved backward along a straight line. The starting movement is executed with rapid traverse values (G00). If obstacles occur on the straight line, the operator must carry out a manual repositioning before automatic restart to the contour.
2. If soft gantry is active, the starting movement is executed decoupled for slave axes, i.e. the axes only move in coupled mode after the approach block.
3. An axis swap is then permitted in block search and continues to permit return to the contour if the axes are requested by a request for positions by the interpolator (default setting, not with #CALL AX FAST).
4. Axes which are moved during block search and which are released before switch-over to read mode may not start automatically at the last known position.

In the example below, the B axis is not moved back to position 45 if automatic restart to the contour is selected. The Z axis is not moved either. The C axis is moved correctly to the contour.



#### Programing Example

##### Automatic restart

```

N00 X10 Y20
N10 #CALL AX [B, 4, 3]
N20 B45      (B axis is not considered)
N30 #PUT AX [B]

N40 Z100
N50 #PUT AX [Z]      (Z axis released, cannot be)
                        (restored)

N60 #PUT AX [X,Y]
N70 #CALL AX [X, 1, 1] [Y, 2, 0]      (exchange X/Y)
                                        (considered)

(-Continuation position-)
N100 X100 Y200      (Continue normal processing)
                    (after block search)

M30
  
```

If program processing is to continue to the continuation position after an abort using block search, the parameters can be obtained directly from the status data in the PLC (see Section Status data: Access via HLI) [► 47]. If the block search parameters are specified directly based on the NC program, it must be taken into consideration that contour influencing functions, e.g. tool radius compensation and polynomial smoothing, affect the original target positions and block limits.



### Attention

If a program only contains relative G91 positions in the block search section, different positions may be approached in the block search despite a restart of the same program.

It is recommended that at least one absolute position occurs in the block search section of the NC program.



### Attention

A homing run G74 is omitted during block search. This may result in different follow-on positions or axis offsets compared to real processing.



### Attention

A measurement block G100 {<axis><destination>} is replaced by a linear motion to the target position during block search.

G01 {<axis><destination>}

Therefore, the specified target position is approached as if the probe signal had not occurred during the measurement.

This may result in different follow-on positions of the axes compared to real processing.

## 9 Examples

### 9.1 Block search type 4

#### 9.1.1 Specify block number and pass counter

##### Type 4 : Block number and number of passes

Block search type = 4, block number = 100

Set number of passes to 1 and start program

Program stops at continuation position X = 10 Y = 12. After continuation, the residual part of the square is traversed.



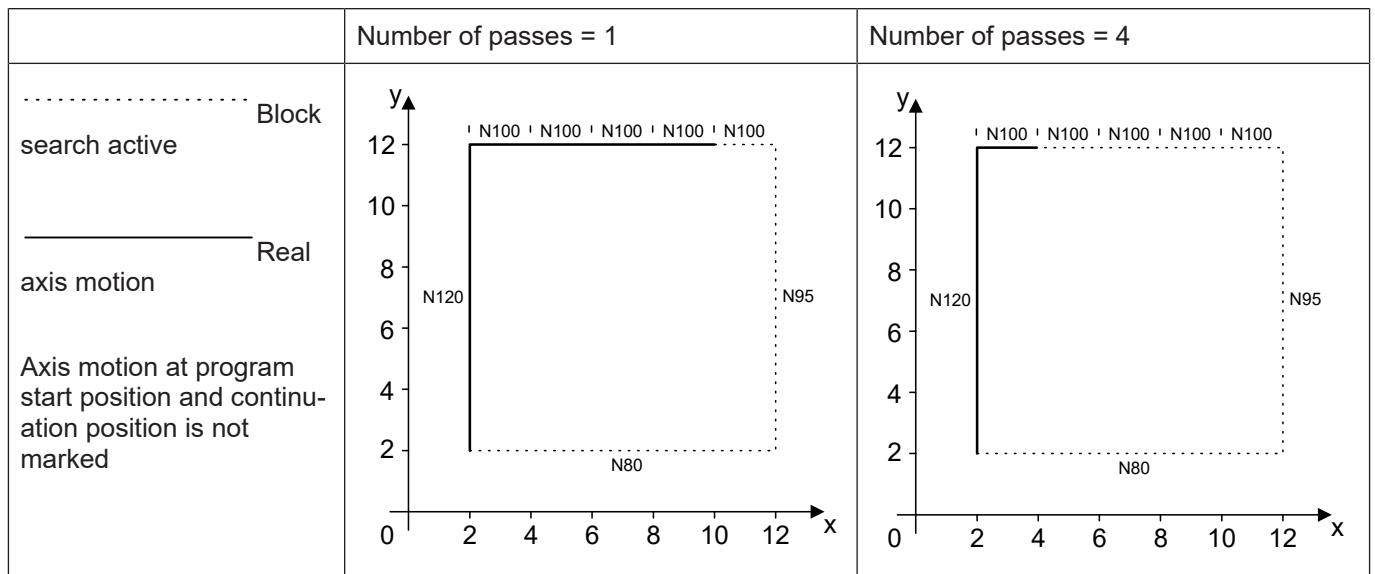
#### Programming Example

##### Block search type 4

```
%t_sv_number.nc
N00 G00 G90 X2 Y2
P1 = 0
N80 G01 G91 X10 F500
N95 Y10

$FOR P1 = 1, 5, 1
N100 X-2
$ENDFOR

N120 Y-10
N130 M30
```



## 9.1.2 Specify the block number and covered distance in the block

### Distance covered in the block

Block search type = 4, block number = 100

Set covered distance and start program.

After continuation, the residual part of the square is traversed.



### Programming Example

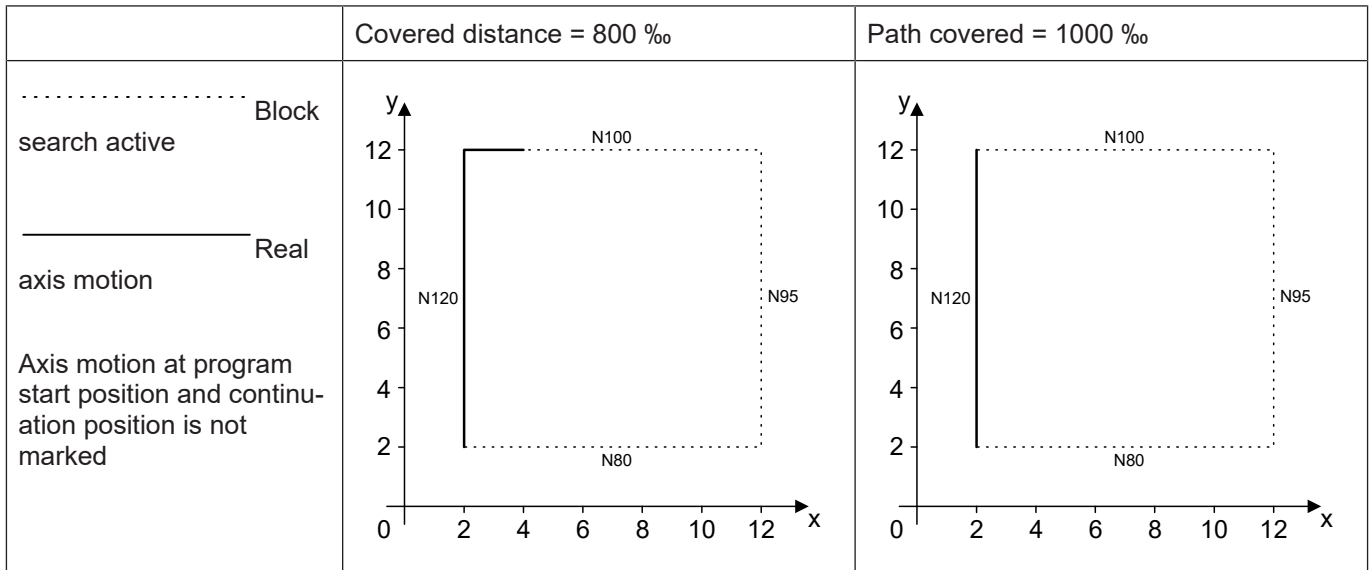
#### Block search type 4

```

N00 G00 G90 X2 Y2
P1 = 0
N80 G01 G91 X10 F500
N095 Y10
N100 X-10
N120 Y-10
N130 M30
    
```

	Covered distance = 0 ‰	Covered distance = 300 ‰
<p>----- Block search active</p> <p>----- Real axis motion</p> <p>Axis motion at program start position and continu- ation position is not marked</p>		





## 9.2 Block search type 3

### 9.2.1 Specify block counter

#### Block counter

Block search type = 3, block counter = 100

Start program

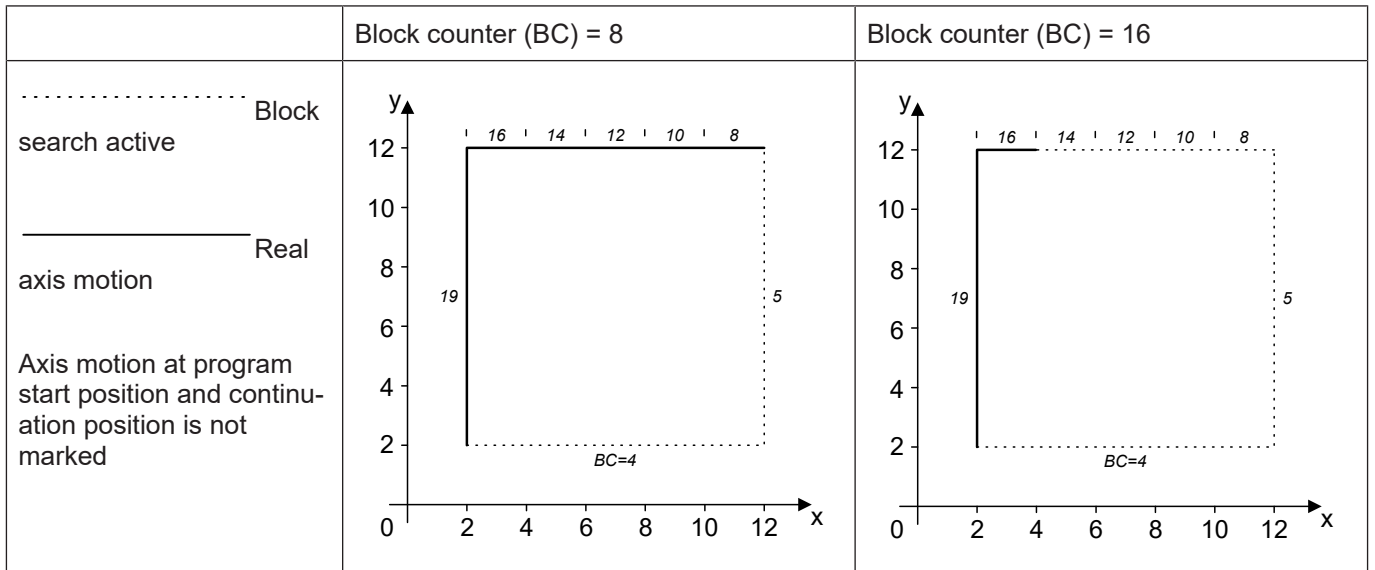
Program is executed up to continuation position. NBC waits for continue. After continuation, the residual part of the square is traversed.



#### Programing Example

```

                                Block counter = BC
%t_sv_count.nc                  1
N00 G00 G90 X2 Y2              2
P1 = 0                          3
N80 G01 G91 X10 F500           4
N095     Y10                   5
                                6
$FOR P1 = 1, 5, 1              7, 9, 11, 13, 15
N100     X-2                   8, 10, 12, 14, 16
$ENDFOR                        17
                                18
N120 Y-10                      19
N130 M30                       20
    
```



## 10 Exceptions and errors

### Block search to a block without motion

Start program and continue in N100 at 10% of covered distance.

Since N100 contains no motion, the block is not split at 10%.



#### Programming Example

##### Block search to a block without motion

```
%t_sv_p.nc
N907090 G01 X0 Y0 Z0 F1000
N070 Y10
N095 X2.1 Y2.2 Z2.3
N100 P100 = 1
N110 G00 G91 X10
N120 G90 X3.1 Y3.2 Z3.3
N907091 M30
```

### Continuation position not found

Block search type = 4, start program

Set number of passes too high /too low. Set unknown block number.

If the number of passes is smaller than 2, the first occurrence of the block number is taken as the continuation position.

If the number of passes is greater than 5, the continuation position is not found and the message P-ERR-20704 is output.

If the block number is not found for continuation, a message (warning) is also output and the entire NC program is terminated in block search mode.



#### Programming Example

##### Block search type 4, start program

```
%t_sv_number.nc
N00 G00 G90 X0 Y0
P1 = 0
N80 G01 G91 X10 F500
N095 Y10

$FOR P1 = 1, 5, 1
N100 X-2
$ENDFOR

N120 Y-10
N130 M30
```

## 11 Appendix

### 11.1 Suggestions, corrections and the latest documentation

Did you find any errors? Do you have any suggestions or constructive criticism? Then please contact us at [documentation@isg-stuttgart.de](mailto:documentation@isg-stuttgart.de). The latest documentation is posted in our Online Help (DE/EN):



QR code link: <https://www.isg-stuttgart.de/documentation-kernel/>

The link above forwards you to:

<https://www.isg-stuttgart.de/fileadmin/kernel/kernel-html/index.html>



#### Notice

##### Change options for favourite links in your browser;

Technical changes to the website layout concerning folder paths or a change in the HTML framework and therefore the link structure cannot be excluded.

We recommend you to save the above "QR code link" as your primary favourite link.

##### PDFs for download:

DE:

<https://www.isg-stuttgart.de/produkte/softwareprodukte/isg-kernel/dokumente-und-downloads>

EN:

<https://www.isg-stuttgart.de/en/products/softwareproducts/isg-kernel/documents-and-downloads>

**E-Mail:** [documentation@isg-stuttgart.de](mailto:documentation@isg-stuttgart.de)

# Keyword index

<b>B</b>	
<hr/>	
Bahn	
Block search	
Stopp:Grund .....	55
Interpolator:active .....	47
Interpolator:distance:continuation position .....	50
<b>C</b>	
<hr/>	
Continuation position	
counter	
covered path	
Block search:distance .....	50
line:NC program .....	50
current:NC program .....	49
<b>F</b>	
<hr/>	
Fahrweg	
Fortsetzposition	
aktuell:NC-Programm .....	54
aktuell:NC-Satz .....	49, 54
Satzvorlauf:Abstand .....	55
<b>H</b>	
<hr/>	
Halt	
Bahn:Bedingung .....	55
<b>L</b>	
<hr/>	
line	
counter:NC program .....	50
<b>N</b>	
<hr/>	
NC program	
NC-Programm	
NC-Satz	
covered path:current .....	49
line:counter .....	50
Fahrweg:aktuell .....	54
Zeile:Zähler .....	55
Fahrweg:aktuell .....	49, 54
<b>P</b>	
<hr/>	
Path	
PCS	
Position	
Stop:reason .....	50
covered path:NC block:to go .....	49
Fahrweg:NC-Programm:Rest .....	54
Block search: continuation .....	50
Satzvorlauf:Fortsetzung .....	55
<b>S</b>	
<hr/>	
Satzvorlauf	
Satzvorlauf an/aus	
Stopp	
Stopp	
Interpolator aktiv .....	53
Interpolator:Abstand:Fortsetzposition .....	55
Interpolator:Zustand .....	48, 53
an PLC .....	58
Path condition .....	50
Path:reason .....	50
Bahn:Grund .....	55
<b>Z</b>	
<hr/>	
Zähler	
Zeile	
Zeile:NC-Programm .....	55
Zähler:NC-Programm .....	55



© Copyright  
ISG Industrielle Steuerungstechnik GmbH  
STEP, Gropiusplatz 10  
D-70563 Stuttgart  
All rights reserved  
[www.isg-stuttgart.de](http://www.isg-stuttgart.de)  
[support@isg-stuttgart.de](mailto:support@isg-stuttgart.de)

