# DOCUMENTATION ABOUT WORKING WITH THE MACH SIMULATION CONTENT (NX8)


1 Highlights - new with NX8 .... 2
2 Overview .... 2
3 Load Options .... 2
4 CAM-Setup .... 3
5 MCS-Handling .... 4
6 About Post Processors .... 5
6.1 Creating file with tool and offset data .... 5
6.2 MOM variable for MCS handling .... 6
6.3 Differences between ‘pure’ template posts and OOTB posts .... 7
7 About CAM Programming .... 8
8 Using the library tools .... 10
9 About CSE Simulation Drivers .... 10
9.1 Handling of Offsets: .... 10
9.2 Handling of the tool change .... 11
9.3 Handling the reference point (Fanuc) .... 12
10 About Sinumerik Cycles in the content .... 12
11 About swiveling cycles .... 13
11.1 PLANE SPATIAL .... 13
11.2 CYCLE800 .... 13
12 Example specific considerations .... 15
12.1 SIM01: .... 15
12.2 SIM03: .... 15
12.3 SIM07 and SIM09 CSE: .... 15
12.4 SIM07 with MTD: .... 15
12.5 SIM10: .... 15
12.6 SIM13: .... 15
13 The Mill-Turn Example sim15 .... 16
13.1 Kinematics Configuration .... 16
13.2 ONT Coordinate systems setup .... 17
13.3 Tool Change .... 18
13.4 About Spindles .... 18
13.5 TRANSMIT Operation and other milling motion types .... 19
13.6 TRAORI simultaneously 5-Axis milling .... 20
13.7 Part transfer from Main to Sub Spindle .... 20
13.8 Definition of Tools and related Settings (not specific for sim15) .... 21
14 Creating your own CSE examples based on the OOTB mach examples .... 22
15 Appendix .... 22
A. Example NC code TRAORI and circles .... 22
B. Detailed information about ini files .... 23
C. Example of a tool change subprogram .... 23
D. Post Processor features of the OOTB examples .... 24
E. Example of an INI file .... 25
F. PLANE SPATIAL .... 26
G. CYCLE800 .... 29
H. History .... 34

# 1 Highlights - new with NX8

- Provide an example for complex dual channel dual spindle Mill-Turn machine tool (sim15)
- Support PLANE SPATIAL function of TNC Heidenhain controller OOTB for most 5 axis machine tools.
- Support CYCLE800 function of Sinumerik controller OOTB for many machine tools.
- Cycle support for Sinumerik (In the template Post and with the delivered machine tools) The MACH content includes an example to simulate external NC code for many different Sinumerik Cycles.

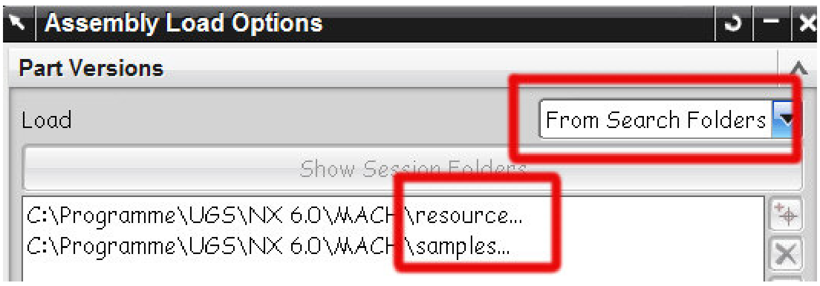
# 2 Overview

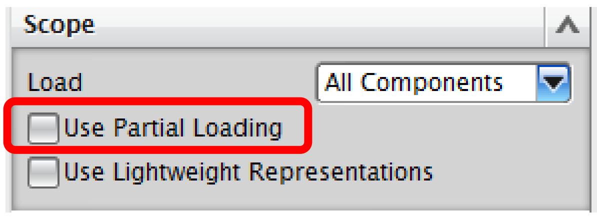
This document explains the handling and usage of the simulation examples provided out-of-the-box (OOTB) with NX Version 8.0. The example data is mainly contained in two locations: one is for the library machine tool models in the `$UGII_CAM_BASE_DIR/resource/library/machine/installed_machines` folder, the other for the CAM examples utilizing the library machine tools. These CAM examples can be found under `$UGII_CAM_BASE_DIR/samples/nc_simulation_samples`. All of the machine tools in the library have preconfigured geometry, assembly and kinematics models as well as post processors and CSE controller models for the major controller types Siemens SINUMERIK 840D, Fanuc family and TNC Heidenhain Conversational; posts are available for metric and inch units. For all machine tools in the library there is one CAM setup example available. The intention of these examples is to show best practice and to demonstrate the features of the NX CAM built-in machine simulation. Another intended use of the examples is as seed parts for customer specific simulation.

***NOTE:*** *The examples cannot contain or show every possible feature of NX CAM, NX Post and ISV. In certain cases such as complex or multi-function machine tools, and in order to achieve advanced capabilities, specific customization of the provided posts and/or controller models will be necessary.*

# 3 Load Options

To ensure all related assembly component parts are loaded correctly it is recommended to use the following load options and not using “USE Partial Loading”:

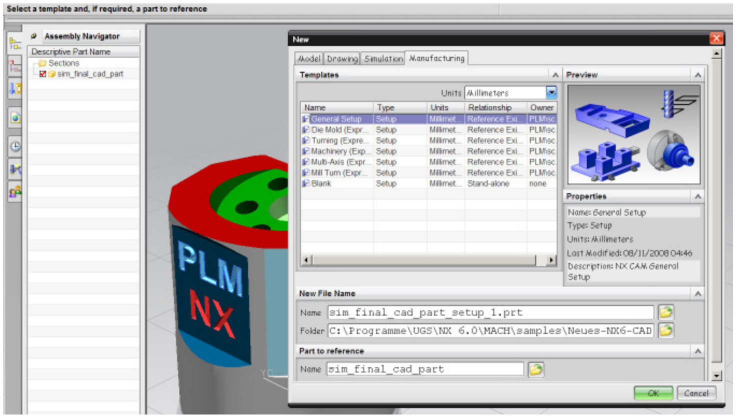


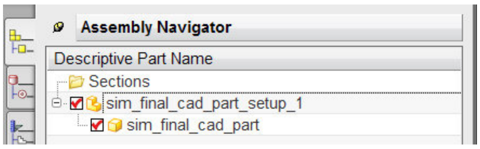
# 4 CAM-Setup

When creating a new CAM setup with the provided set of machine tools, the following best practice is suggested. It is assumed that the CAD geometry of the part to be machined already exists.

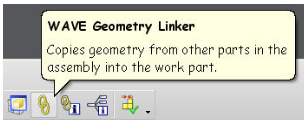
- Open the CAD part file in NX
- Select “New” and pick an appropriate entry in the Manufacturing tab.



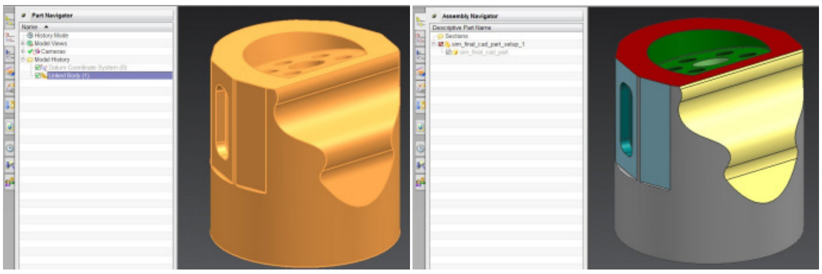
- The system will automatically create a master-model-concept-part-file referencing your CAD geometry.



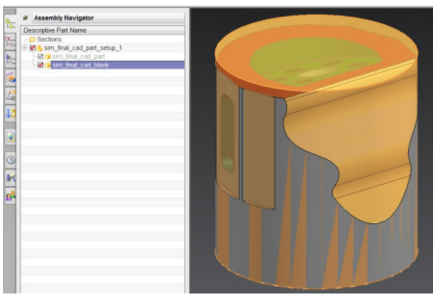

- Create a Wave link to the CAD geometry




- Select the body of the CAD model for the Wave object and hide the CAD component in the assembly navigator



- To prepare for material removal simulation it is recommended to have the blank geometry pre-defined as well. In the example shown it is a simple cylinder added to the assembly.

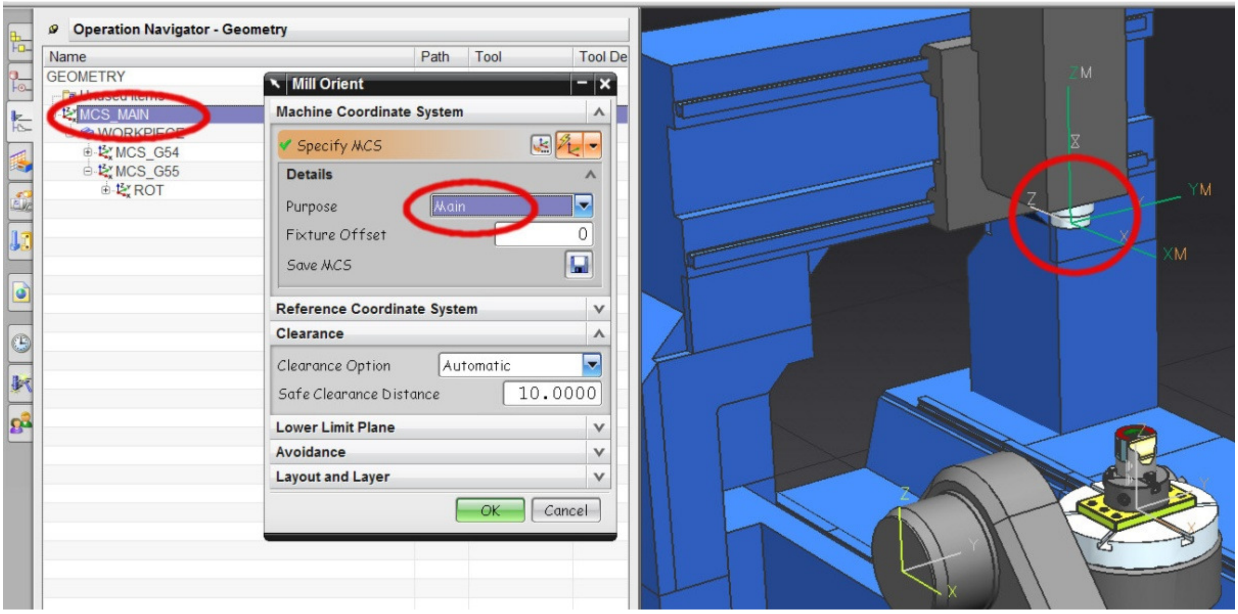


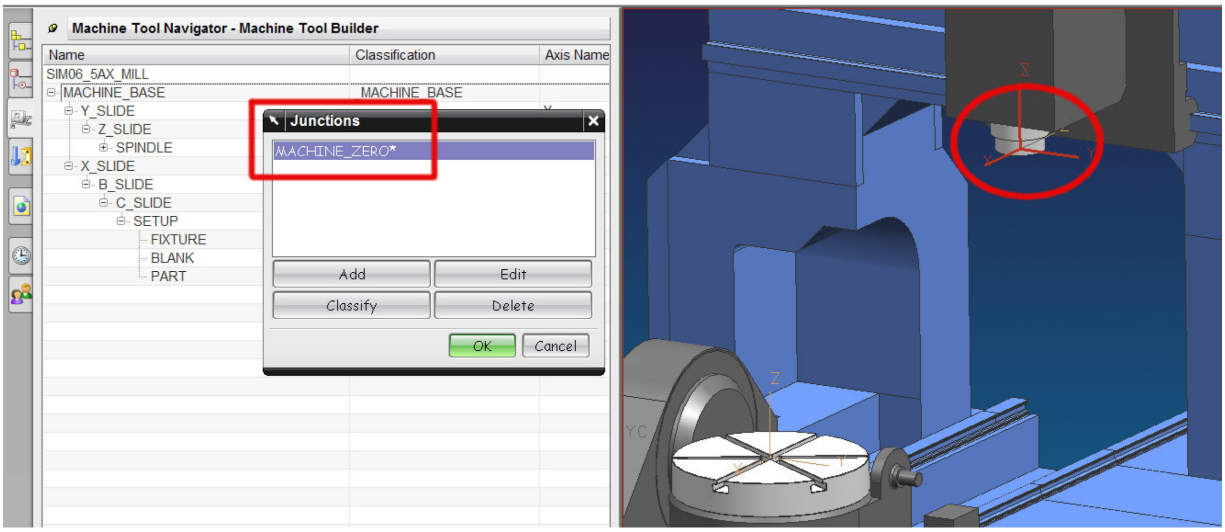
# 5 MCS-Handling

In order to achieve a complete and reliable machine simulation, different modules need to play together. These are: the CAM programming structure in the Operation Navigator (ONT), the internal Post Processor (MOM inside NX), the TCL based post processor and the simulation controller model.
Consider a few rules for the NX CAM setup to support the assigned post processor creating a valid NC code which can be used at the physical machine tool as well as for simulation. Our examples are using only one MCS, which is set to Purpose "Main" (see picture below). The post processor output depends on "MCS Purpose", "Special output" and "Fixture Offset" settings.

- Each CAM setup example using one of our OOTB MACH machine tools must not include more than one single MCS with the purpose "Main"
- The "Main" MCS needs to be placed at the same location and orientation as the MACHINE_ZERO coordinate junction of the machine tool.

***NOTE:*** *This is necessary to tell the post processor the position of the machine zero position. In future releases it is planned to enhance the Post-TCL interface to access the kinematics model. Once this is implemented, the rule to place the main MCS at the same location as machine zero junction will become obsolete.*

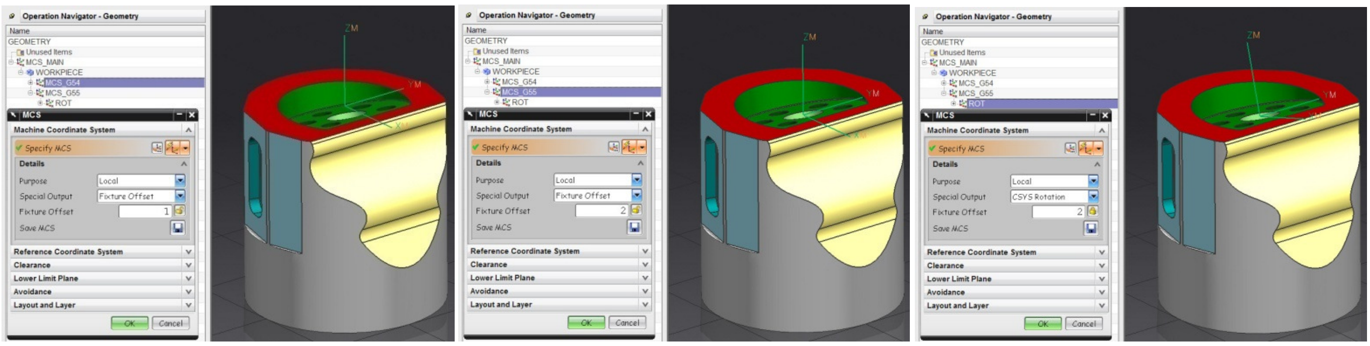


- All other used MCS's in the ONT need to be of purpose "Local"
- If the "Special Output" of the local MCS is "Fixture Offset", the post processor will output a fixture offset statement based on the number of the "Fixture Offset".
  E.g. if Fixture Offset is 2:
  G55 -> SINUMERIK
  G55 -> Fanuc
  CYCL DEF 7.0 -> TNC *(7.1/2/3 will include the offset values e.g. CYCLDEF 7.1 X 100)*
- If the "Special Output" of the local MCS is "CSYS Rotation" the postprocessor will output a special statement to indicate a translation and/or a rotation
  ROT -> SINUMERIK
  G68 -> Fanuc
  PLANE SPATIAL -> TNC (*supported for 5 axis milling sim05-sim09 & sim14)*

With these settings the example will look like:

| MCS Name | Purpose | Fixture Offset | Special Output | Post S840D | Post Fanuc | Post TNC |
|---|---|---|---|---|---|---|
| **MCS_MAIN** | Main | 0 | - | | | |
| \| **→MCS_G54** | Local | 1 | Fixture Offset | G54 | G54 | CYCL DEF 7.0 |
| \| **→MCS_G55** | Local | 2 | Fixture Offset | G55 | G55 | CYCL DEF 7.0 |
| \|**→ ROT** | Local | 2 (inherit) | CSYS Rotation | ROT | G68 | PLANE SPATIAL |



# 6 About Post Processors

All the OOTB delivered post processors are created with the Post Builder Version 8.0. based on the template post for Sinumerik S840D. Some differences between pure template post and OOTB post are listed in the 3rd subchapter.

## 6.1 Creating file with tool and offset data

The SINUMERIK 840D post processor creates a CAM setup specific initialization data file for offset values and the tool data. The post processor creates an additional ini file (to_ini-*Channel*.ini) in the SINUMERIK format. This file will be located in the *cse_files/subprog* subfolder of the CAM setup part:

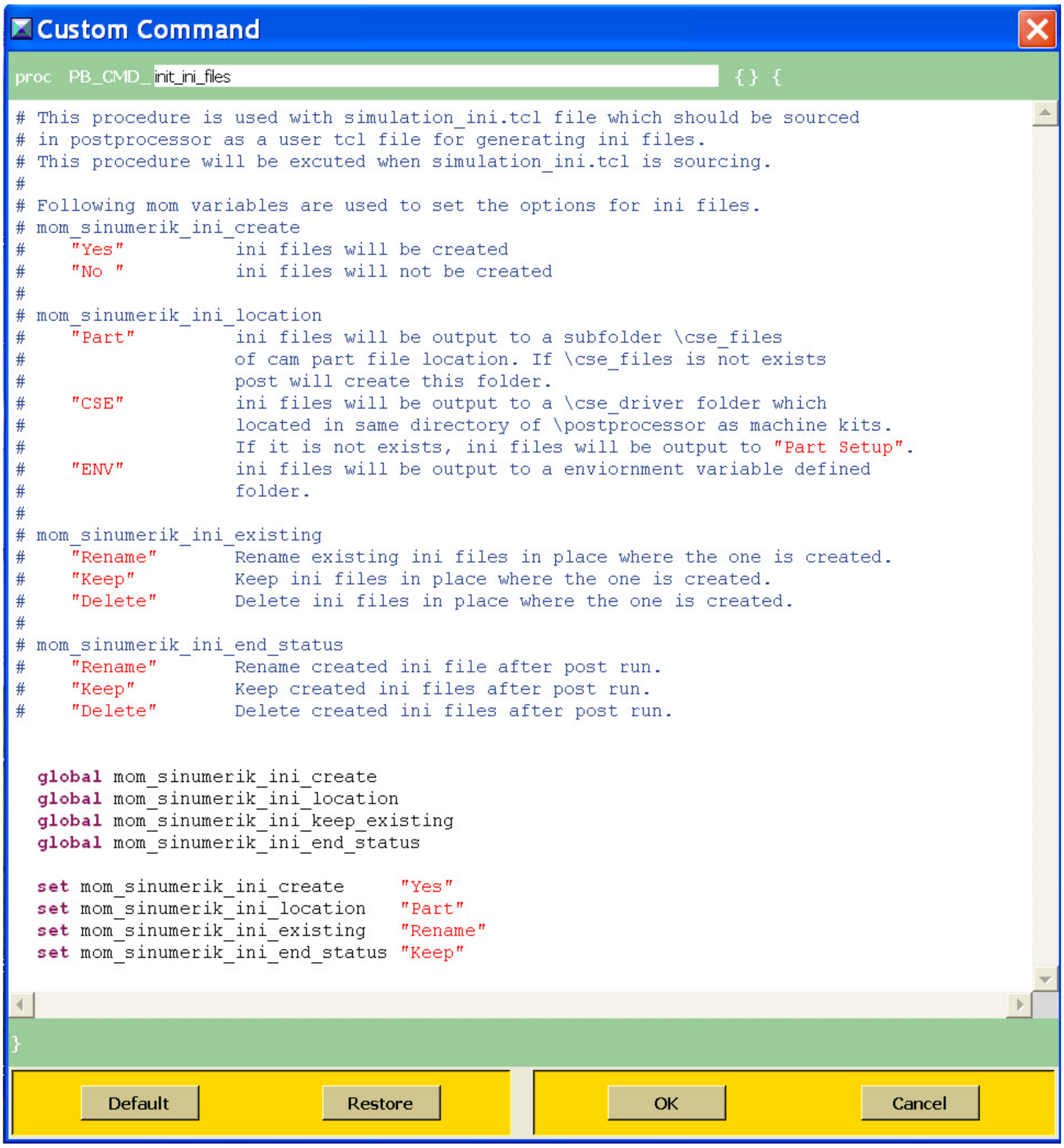
```
…MACH/samples/nc_simulation_samples/cse_files/subprog
```

This to_ini.ini file is handled as an subprogram and in the ini files of the machine tool in the library this file gets called (executed and loaded). When working with CSE controller models, the different ini files are loaded and executed before the simulation starts. The format and syntax of the files is controller specific and their purpose is to initialize certain settings upfront. For more details about ini files please see Appendix.

Before the post create a new file to_ini.ini it will back up the existing ini file by renaming it to "*.bck".

***Note:** With the Sinumerik template Post within PostBuilder Version 8 you are able to configure the behavior of creating the ini file.*

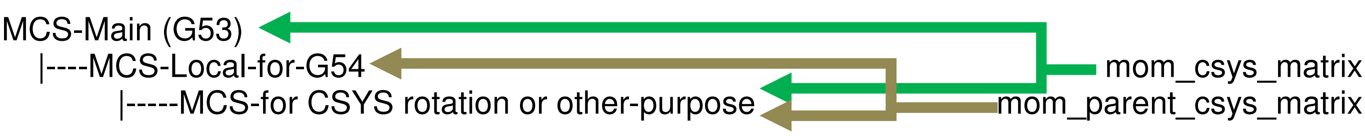
***NOTE:*** *The OOTB example machine tool ini files for Sinumerik (e.g. sim07_mill_5ax-Main.ini) have the entry TO_INI to call the post created file as a subprogram. If the creation of the file not work properly or the file gets deleted or the post does not have write access to create this file and the CSE Simulation starts it will give you a warning, that the related TO_INI file could not be found.*




```
# This procedure is used with simulation_ini.tcl file which should be sourced
# in postprocessor as a user tcl file for generating ini files.
# This procedure will be excuted when simulation_ini.tcl is sourcing.
#
# Following mom variables are used to set the options for ini files.
# mom_sinumerik_ini_create
#     "Yes"          ini files will be created
#     "No "          ini files will not be created
#
# mom_sinumerik_ini_location
#     "Part"         ini files will be output to a subfolder \cse_files
#                    of cam part file location. If \cse_files is not exists
#                    post will create this folder.
#     "CSE"          ini files will be output to a \cse_driver folder which
#                    located in same directory of \postprocessor as machine kits.
#                    If it is not exists, ini files will be output to "Part Setup".
#     "ENV"          ini files will be output to a enviornment variable defined
#                    folder.
#
# mom_sinumerik_ini_existing
#     "Rename"       Rename existing ini files in place where the one is created.
#     "Keep"         Keep ini files in place where the one is created.
#     "Delete"       Delete ini files in place where the one is created.
#
# mom_sinumerik_ini_end_status
#     "Rename"       Rename created ini file after post run.
#     "Keep"         Keep created ini files after post run.
#     "Delete"       Delete created ini files after post run.


  global mom_sinumerik_ini_create
  global mom_sinumerik_ini_location
  global mom_sinumerik_ini_keep_existing
  global mom_sinumerik_ini_end_status

  set mom_sinumerik_ini_create     "Yes"
  set mom_sinumerik_ini_location   "Part"
  set mom_sinumerik_ini_existing   "Rename"
  set mom_sinumerik_ini_end_status "Keep"
```

## 6.2 MOM variable for MCS handling

For coordinate rotation NC codes output inside the post, the related coordinate matrix is changed in PB_CMD_set_csys. Reason for this is because the Main MCS is representing Machine zero and Local fixture offset representing machining coordinate G54, G55…, but mom_csys_origin and mom_csys_matrix still map current local MCS to Main MCS. For CSYS rotation coordinate output, the value of linear offset (G68, CYCLE800 …) should be the offset between current local MCS to parent local MCS(e.g. G54,G55). Therefore, mom_parent_csys_matrix replaced mom_csys_matrix in this command.

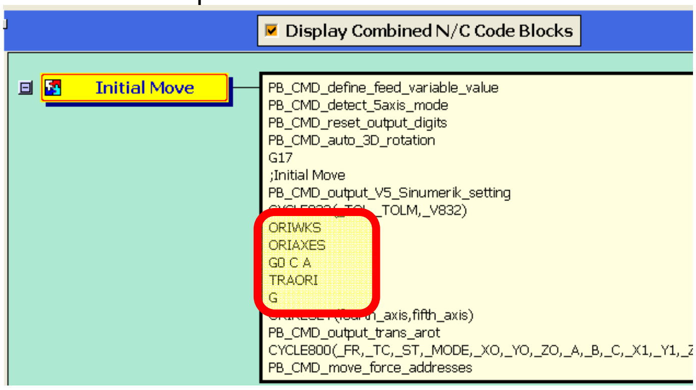
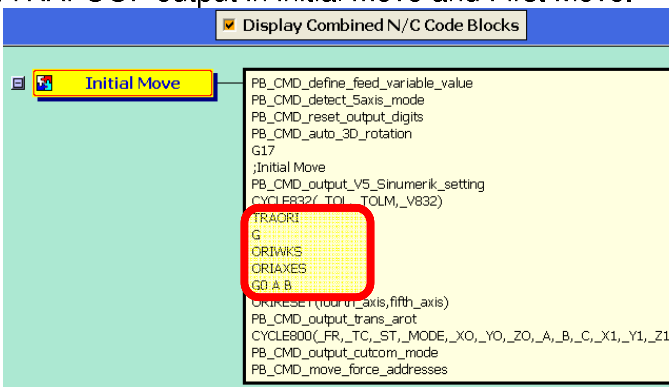


## 6.3 Differences between ‘pure’ template posts and OOTB posts

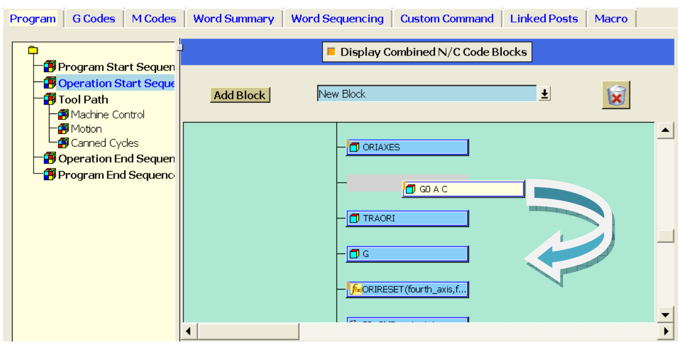
The OOTB posts are created based on the latest template posts for Sinumerik S840D. Some settings and or features are slightly different, which are listed here.

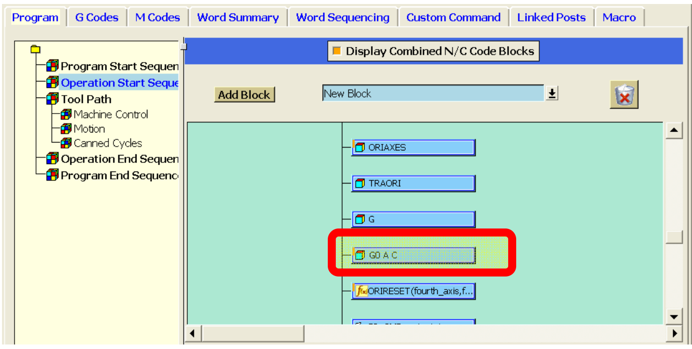
### 6.3.1 Order of Position rotary Axis and activate TRAORI mode

The order of the posted output between the initial positioning motion of the rotary axis and the activation of the TRAORI mode is different.

| Posts of OOTB Examples | Posts from template post |
|---|---|
| Preset the rotary axis angles **before** TRAORI /TRAFOOF output in Initial Move and First Move | Preset the rotary axis angles **after** TRAORI /TRAFOOF output in initial move and First Move. |
|   |   |

This can be easily changed by drag the box “G0 AC” and drop it under TRAORI.




### 6.3.2 Automatic detection on operation type

OOTB post will automatically detect 3 axis and 3+2 axis operation. This function is not implemented in template post, it is added in PB_CMD_before_motion. The command name is PB_CMD_detect_3axis_milling. With this function TRAFOOF will be output for 3 axis operations and TRAORI will be output for 3+2 operations.

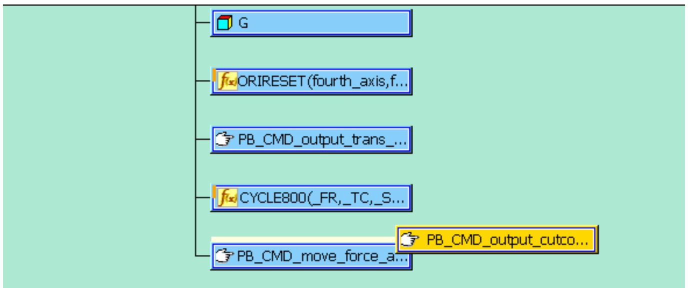
```
proc  PB_CMD_ before_motion  { } {
        set mom_siemens_5axis_output_mode 1
      }
   }

  # Output cutcom mode and caculate positions
   if { [llength [info commands PB_CMD_calculate_cutcom]] } {
      #PB_CMD_calculate_cutcom
   }

 # Auto detect 3 axis and 3+2 axis milling
   global mom_current_motion

   if {[string match "initial_move" $mom_current_motion] || \
       [string match "first_move" $mom_current_motion] } {
      PB_CMD_detect_3axis_milling
   }
```

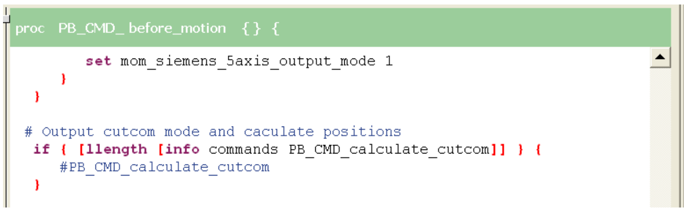
### 6.3.3 Output of 3DCUTCOM

This mode will not be outputted in the OOTB posts mainly because it is not supported by simulation at this point in time.

| Posts of OOTB Examples | Posts from template post |
|---|---|
| **No output** for 3D-CutCom Mode | **Output** for 3D-CutCom Mode |

To restore 3Dcutcom output in OOTB post, remove the "#" before PB_CMD_calculate_cutcom in PB_CMD_before_motion. And put PB_CMD_output_cutcom_mode into Initial move and First Move as below.



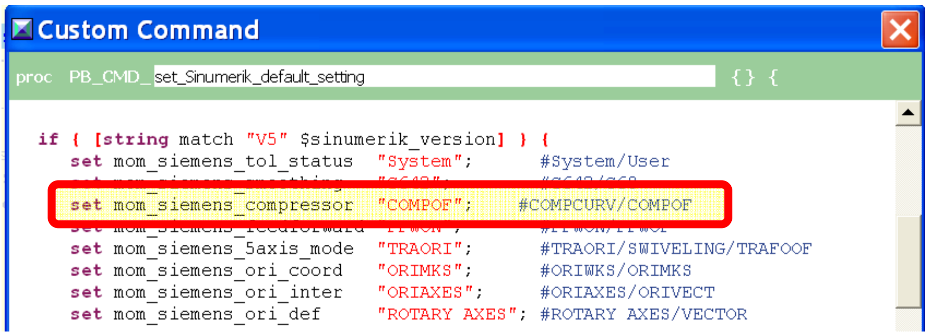




### 6.3.4 COMPRESS Function

The output of the compress functions is disabled in the OOTB Posts, mainly because it is not supported by simulation at this point in time.

| Posts of OOTB Examples | Posts from template post |
|---|---|
| **No output** for compress mode | **Output** for compress mode |

PB_CMD_set_sinumerik_default_setting to make circular motion is available. The Compressor mode also could be changed by Sinumerk_840D UDE on operations.

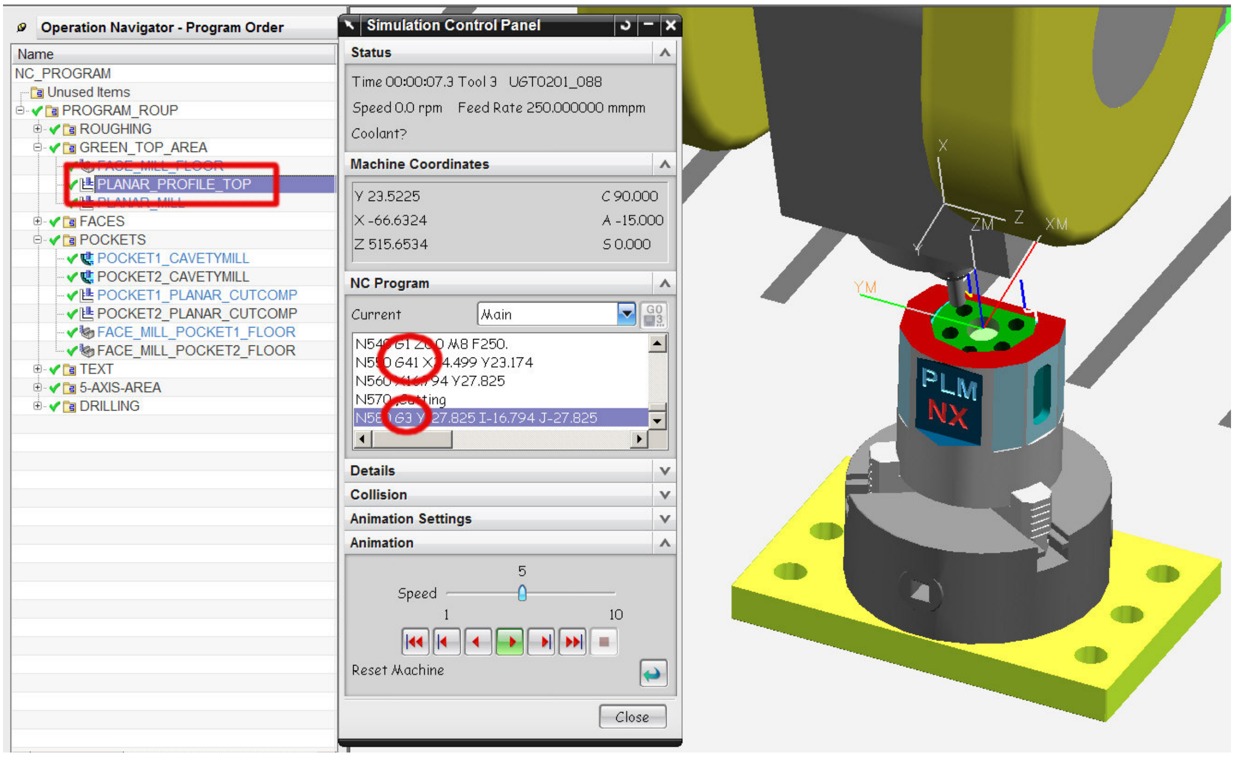



# 7 About CAM Programming

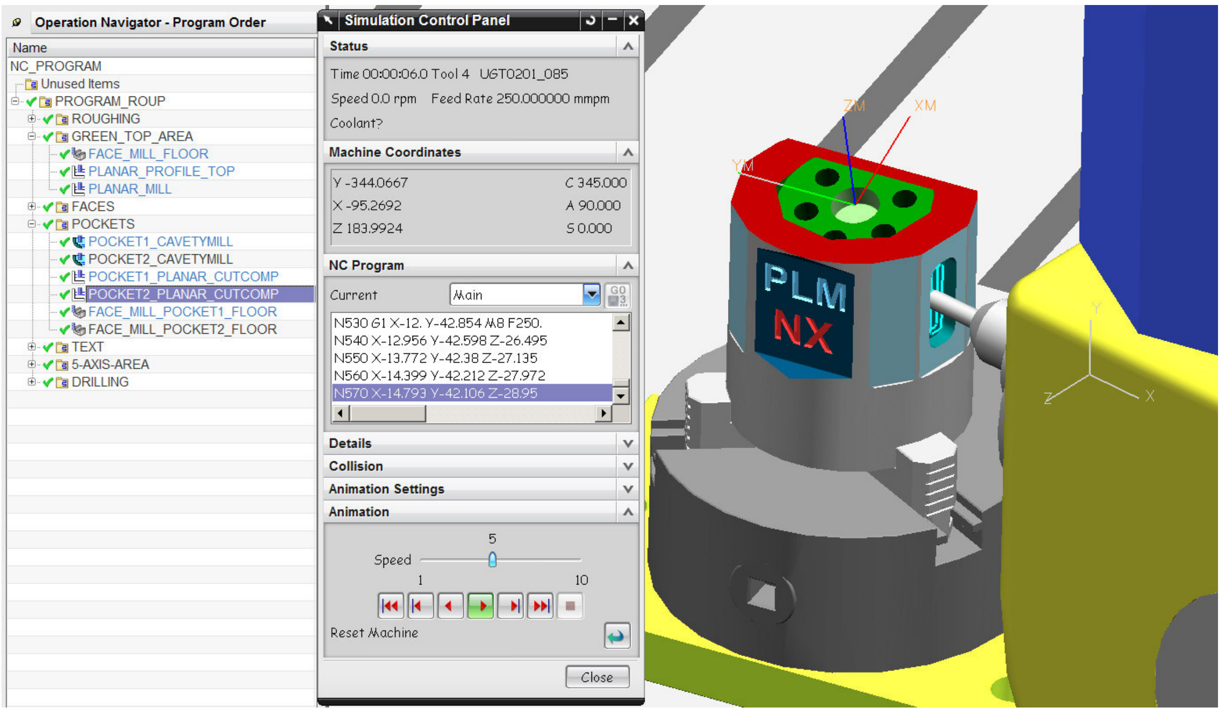
About Cutter Compensation for contact contour and circle statements in the NC code file: Whenever the NC code contains statements which refer to a plane such as G02/G03 or G41/G42, the controller needs to have a working plane defined. Therefore it is necessary to define the correct plane upfront. The initialization files provided with the machine tool examples define a default working plane. This is typically the XY-Plane (G17) for the milling machine tools and ZX-Plane (G18) for the turning machine tools.

Important to understand are cases where the surfaces to be machined are not parallel to one of the orthogonal machine tool planes and the post processor has a specific transformation like TRAORI activated to achieve this operation. A typical example is shown with the external NC code listed in Appendix simulated with the machine tool example sim06_mill_5ax_cam_sinumerik_mm.prt.

Activating cutter comp or circle output will only be correct if a related working plane is defined. This can be seen e.g. in the following picture for the green area and the operation PLANAR_PROFILE_TOP. It is taken from the example sim05 (head/head configuration). Here, the operation PLANAR_PROFILE_TOP includes circle statements for a plane which is not aligned with xy, yz, or zx. Circle records can correctly processed only because a “ROT” command was executed in a previous NC line and switched the plane in the TRAORI mode.




The example shown below (POCKET2_PLANAR_CUTCOMP, example sim05) is an alternative method when machining within a plane which is not aligned with xy, yz, or zx. Here, the working plane is not defined by a ROT statement; circular or cut comp output would not work and therefore are deactivated in the operation.



# 8 Using the library tools

The CAM setup simulation examples use as many as possible tools available from the OOTB tool library (ASCII in NX native)

# 9 About CSE Simulation Drivers

## 9.1 Handling of Offsets:

Like any physical NC controller, CSE drivers can process offset information (activated e.g. through G55) only if the required data is provided, i.e. coordinate values between the actual chosen offset e.g. G55 and the machine tool zero position. There are two ways to achieve this with CSE and both are used in the OOTB examples.

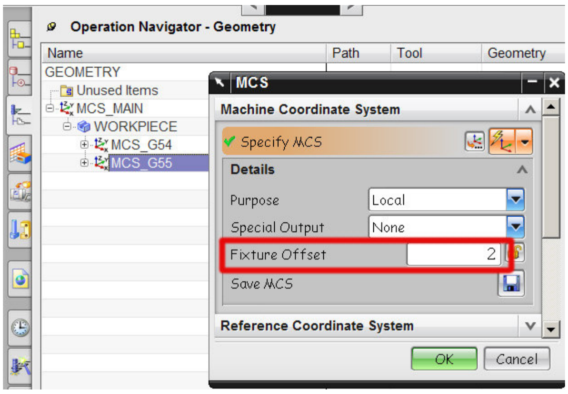
### 9.1.1 Alternative 1 (Fanuc)

In the case of Fanuc, the controller queries the information during simulation from the application (ISV NX). This is done internally by a command called *"LoadOffset"* during interpretation of an offset statement in the NC code like G55. When the simulation is executed from the operation in the ONT, the system used the related MCS information where the operation is located under.

***Note:*** *This mechanism will not work if more MCS levels are used and the related MCS, which defines the offset e.g. G54 is NOT one where the operation is located under. Example:*
*MCS-Main*
*|----MCS-Local-for-G54*
*|-----MCS-for-other-purpose*
*|----**Operation***

When the simulation is executed by selecting an external NC code file, the system looks for an MCS by searching the MCS objects in the GEOMETRY view of the ONT for the related fixture offset number. The first which is found will be used.

Example: In the NC code a G55 (2nd offset) is used. In this case the system cycles through all MCS objects and compares the value use in "Fixture Offset" with the given offset in the NC code.

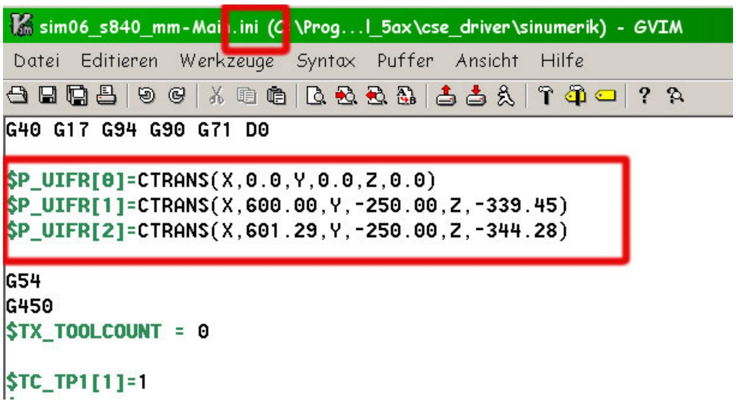



### 9.1.2 Alternative 2 (Sinumerik and TNC)

The SINUMERIK 840D CSE and the TNC controller doesn't use the "LoadOffset" mechanism as described above but follows an alternative implementation. On the physical controller in the shop floor, the offset values are typically measured by probing operations or are manually set in the controller. The controller stores these values in an offset table and in controller variables.

The SINUMERIK OOTB post processors create an initialization file for the actual CAM setup including information about the offsets and tool data. This initialization file is loaded before the simulation starts. So the offset definition is achieved by the definition of the variables in the initialization file.

Snapshot of the SINUMERIK ini file:




Later, when one of the offsets is activated in the NC code, the CSE controller uses these variables to calculate the offset transformation.
The TNC OOTB post processors outputs the offset data into the main program based on the actual CAM setup. The offset gets activated right away by parsing these NC code lines.

Example for TNC offset data:

```
…
8 CYCL DEF 7.0
9 CYCL DEF 7.1 X -0.0000
10 CYCL DEF 7.2 Y -225.0000
11 CYCL DEF 7.3 Z -327.4470
…
```

## 9.2 Handling of the tool change

Tool changes with CSE controller models in the existing examples are achieved by calling a tool change subprogram. This subprogram is located under the “subprog” folder for each CSE driver; it’s kept in the corresponding NC code syntax and basically positions the tool to the tool change location. Using one of these subprograms for a different machine tool will typically require an adjustment of the tool change position. Another section in the tool change subprogram is the Anycontroller (AC) part, which mimics the PLC portion of the tool change. This mainly takes care of the mounting and un-mounting of the tool itself. Refer to NX Help for more details about the AC language. The OOTB examples demonstrate different kinds of tool change methods.

- The “standard” way is that one spindle is defined on the kinematics model of the milling machine tool and the tools are mounted ‘on the fly’ during simulation. This can be seen in sim02 to sim09 as well as sim14.
  sim08 is similar, but shows how an advanced tool change mechanism can be animated using the AC language to open and close doors.
- In sim01, tools are already pre-mounted and visible in the CAM scenario on an eight pocket tool changer. During simulation, the system moves the spindle and the tool changer to mount and un-mount the tool.
- The turning examples sim10 to sim13 don’t mount tools during simulation. Here, all tools are already pre-mounted on the turret in the CAM scenario. The tool change subprogram takes care of the rotation of the turret and the activation of the selected tool.

An example of a tool change subprogram can be found in Appendix

***Note:*** *The machine tool view in the ONT reflects the definition of turrets and pockets in the kinematics model. Each time the machine tool is retrieved from library, the ONT is updated based on the kinematics model. It is strongly recommended not to add or remove turrets and pockets in the ONT machine tool view when working with machine tools.*

### 9.3 Handling the reference point (Fanuc)

In Fanuc NC code, the G command G28 is often used to move the reference point. The OOTB examples include this in the post and in the CSE controller models. This section describes how the position of the reference point is defined and stored to get good simulation results.
On physical machine tools, the reference point is stored inside of the controller. To mimic this in the virtual CSE simulation, the position of the reference point is defined in the *.ini file along with the machine tool. The NC code to define reference points in Fanuc syntax is shown in the sim01 example (sim01_mill_3ax_fanuc-Main.ini) as:

```
(This part sets the position of the reference point G28)
G54
G17
G90
G10L52
N1240P1R0000       -> X Position of the reference point related to machine zero
N1240P2R225.425    -> Y Position of the reference point related to machine zero
N1240P3R406.425    -> Z Position of the reference point related to machine zero
G11
```

If needed, the values of the reference point can easily be changed in the ini file. All coordinates are assumed to be metric.

## 10 About Sinumerik Cycles in the content

With NX8 the NX comes OOTB with a list of Sinumerik cycles for simulation. The cycles are covered by an encrypted archive file (*.cyc) and placed in the subprog folder of the Sinumerik CSE simulation. Two versions are added to the system one for the older Sinumerik controller “Sinumerik Powerline” and one for the actual new one “Sinumerik Solutionline”. The last one is used by default in the CSE simulation.

- The cycles of the Solutionline are in the file: SinumerikSL_Cycles.cyc
- The cycles of the Powerline are in the file SinumerikPL_Cycles.cyc_powerline

To use the cycle from Powerline instead of the one from Soluitonline the files need to be renamed like:

- SinumerikSL_Cycles.cyc_solutionline
- SinumerikPL_Cycles.cyc

Here the list of cycles in the archives files:

PowerLine

- CYCLE71
- CYCLE81 – CYCLE90
- CYCLE801
- HOLES1, HOLES2
- LONGHOLE
- SLOT1, SLOT2
- POCKET1, POCKET2, POCKET3, POCKET4
- CYCLE800 (kinematic specific, customized per machine tool)

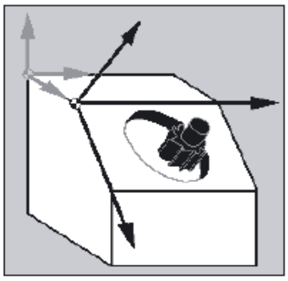
Solutionline

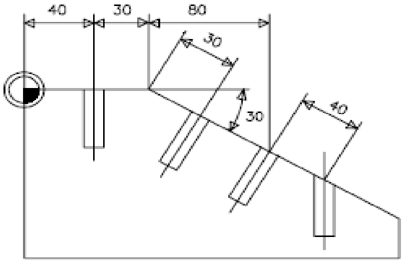
- CYCLE71
- CYCLE81 – CYCLE90
- CYCLE801
- HOLES1, HOLES2
- LONGHOLE
- SLOT1, SLOT2
- POCKET3, POCKET4

- CYCLE800 (kinematic specific, customized per machine tool)

# 11 About swiveling cycles

This will give an overview about what are swiveling cycles, how they are used. The man target is to machine on a plane not perpendicular to an existing linear axis. What it would take without swiveling:
You have to create a rotated coordinate system
Figure out how to move the rotary axis so that X, Y, Z can be used for the motions in the plane
Solve the problem that your work piece coordinate system does not rotate with the rotary axis






Working with swiveling cycles will make the life easier and let the controller do the work. In the following subchapters it is explained in detail how the swiveling Cycle PLANE Spatial on Heidenhain controllers and the CYCLE800 on Sinumerik S840D controllers work and how this is implemented n the CSE and the OOTB examples. In addition you will see a section how this is configured and can be reused for a different machine tool.

## 11.1 PLANE SPATIAL

With NX8 for the following machine tool the PLANE SPATIAL implementation will be part of the OOTB examples.

- Sim05
- Sim06
- Sim07
- Sim08
- Sim09
- Sim14

More details about the cycle and the ay how it is implemented in CSE can be found in the Appendix

### 11.1.1 <u>**How the ONT needs to be defined to achieve a proper Post output**</u>

To achieve an output of PLANE SPATIAL by the Post the following prerequisites needs to be fulfilled.

- milling operations and drilling operations needs to be 3+2 orientation
- The MCS need to be set to “CSYS Rotation”

## 11.2 CYCLE800

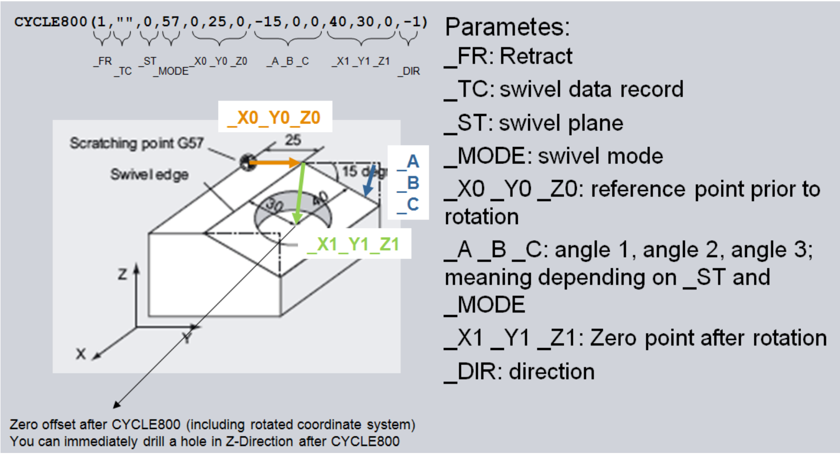
### 11.2.1 <u>**Includes in the OOTB examples:**</u>

With NX8 the following OOTB examples include a CYCLE800 implementation:

- Sim05
- Sim06
- Sim07
- Sim08
- Sim14

The Sinumerik Solutionline version is implemented in this approach. Much more details about the cycle itself and the ay how it is implemented in CSE can be found in the Appendix

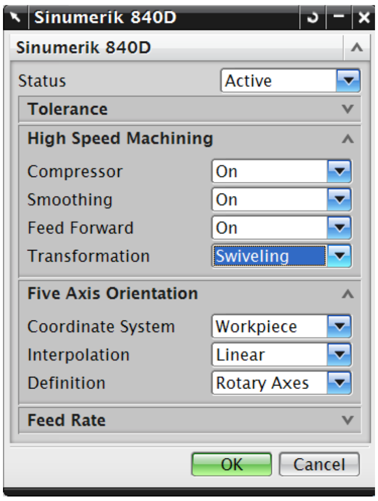
### 11.2.2 Example and parameter




### 11.2.3 How the ONT needs to be defined to achieve a proper Post output

To achieve an output of PLANE SPATIAL by the Post the following prerequisites needs to be fulfilled.

- milling operations and drilling operations needs to be 3+2 orientation
- Sinumerik 840 UDE should be added on the operations, Transformation option should choose “Swiveling” as




### 11.2.4 Machine Tool dependent data – customized ini files

This is the list of variables defined for each machine tool in the ini file and others files to define data for the CYCEL800. These are the data, which need to match the related machine tool.

Content of the OOTB INI file

```
SMAC; adopt the original DEF file from MC
PGUD; need remove the line with REDEF
TC_CARR; define tool holder data
G40 D0
$MN_MM_NUM_TOOL_CARRIER=1
M17
```

Definition data of the TC variables:

```
$TC_CARR7    x component of rotary axis v1
$TC_CARR8    y component of rotary axis v1
$TC_CARR9    z component of rotary axis v1
$TC_CARR10   x component of rotary axis v2
$TC_CARR11   y component of rotary axis v2
$TC_CARR12  z component of rotary axis v2
$TC_CARR23  kinematic type
```

```
$TC_CARR24  Offset of rotary axis v1
$TC_CARR25  Offset of rotary axis v2
$TC_CARR30[1]=-90; software minimum limit of rotary axes
$TC_CARR31[1]=-9999999
$TC_CARR32[1]=90; software maximum limit of rotary axes
$TC_CARR33[1]=9999999
$TC_CARR34  tool holder name
$TC_CARR35  Axis name 1
$TC_CARR36  Axis name 2
$TC_CARR37  Identifier how to set it?
$TC_CARR40  Z axis retract value
```

# 12 Example specific considerations

## 12.1 SIM01:

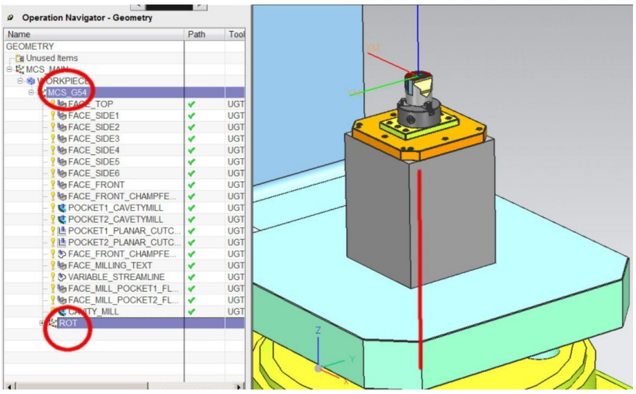
This example has a somewhat fancy tool change mechanism including a real tool changer. Re-using this example for other machines without a similar tool changer can cause unexpected behavior. In this case, use sim02 as reference for a simple 3 axis milling machine tool.

## 12.2 SIM03:

This metric Sinumerik example includes a cam setup, which handles operation using milling tool (T-Cutter and Step-Mill) dealing with multiple tracking points. (New enhancement of NX8)

## 12.3 SIM07 and SIM09 CSE:

These two machine tools are characterized by a rotary axis in the head and another rotary axis in the table. When using a “Local” MCS with the special settings “CSYS Rotation”, the following restriction exists for placing the MCS. In these cases, the parent MCS (e.g. MCS_G54) and the rotated MCS (e.g. ROT) need to be placed in the center of the rotation of the table. The following figure shows this case in sim07. Ignoring this rule will result in invalid G code. This restriction applies to NX CAM Post and Fanuc only.




## 12.4 SIM07 with MTD:

This case is only supported with CSE but not with MTD in the OOTB examples. As a workaround to allow MTD simulation with sim07, all related junctions and MCS need to be placed at the center of the table.

## 12.5 SIM10:

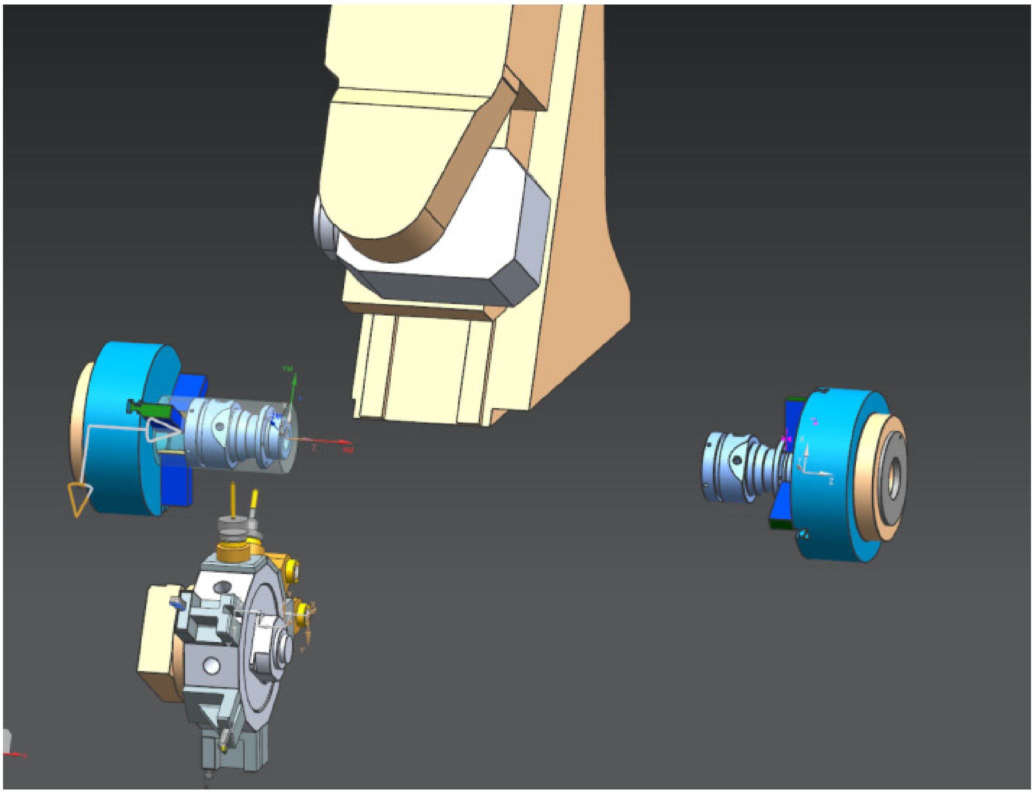
This machine tool is only available with a Fanuc controller model, no material removal or tool path based simulation.

## 12.6 SIM13:

This machine tool is only available with a Fanuc controller model.

# 13 The Mill-Turn Example sim15

With NX8 we provide an OOTB Mill-Turn example with a Post Processor and Simulation Driver for Sinumerik S840D. The machine tool is a two channel B-Axis head and lower turret configuration with a main and sub spindle additionally the kinematics includes a non orthogonal X,Y,Z configuration and allows simultaneously 5-Axis milling on the main spindle (TRAORI). The CAM example includes different operations for milling and turning including TRANSMIT and Part Transfer. The goal of this example is to show the power of the CSE simulation capability.

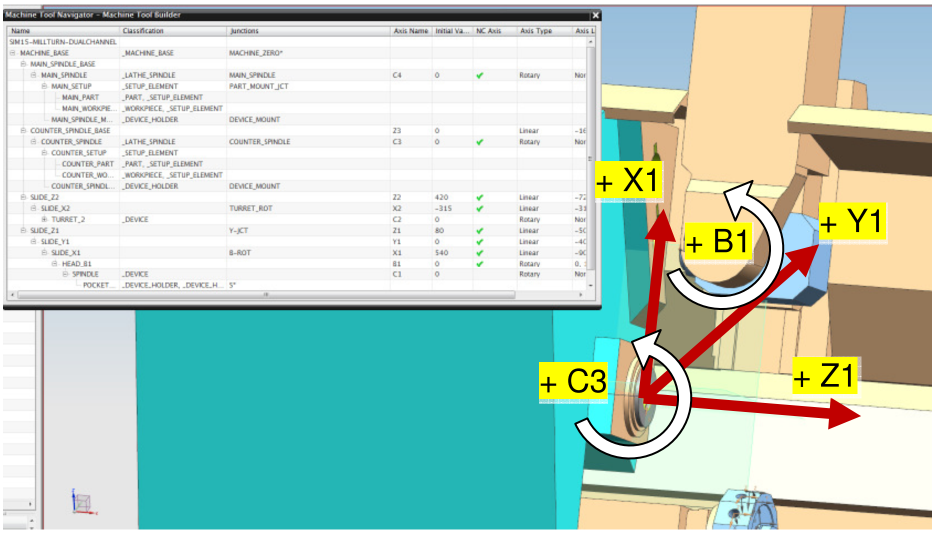


***Note:*** *Due to the general complexity and variations of mill turn machine tools it could not expected that these example can be reused for a different mill turn machine tool type. For such complex manufacturing centers a customization for Post and Driver is always mandatory.*

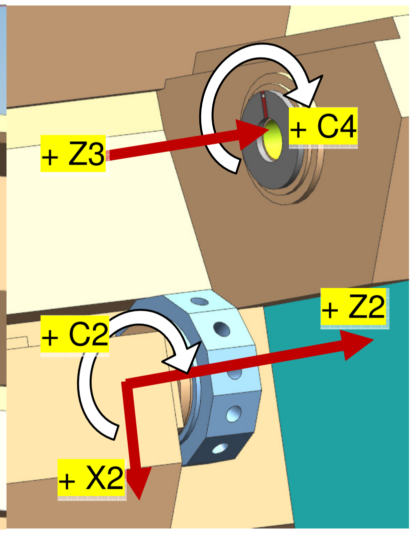
***Note:*** *This example is available in Sinumerik CSE only. No TPB and no MTD*

## 13.1 Kinematics Configuration

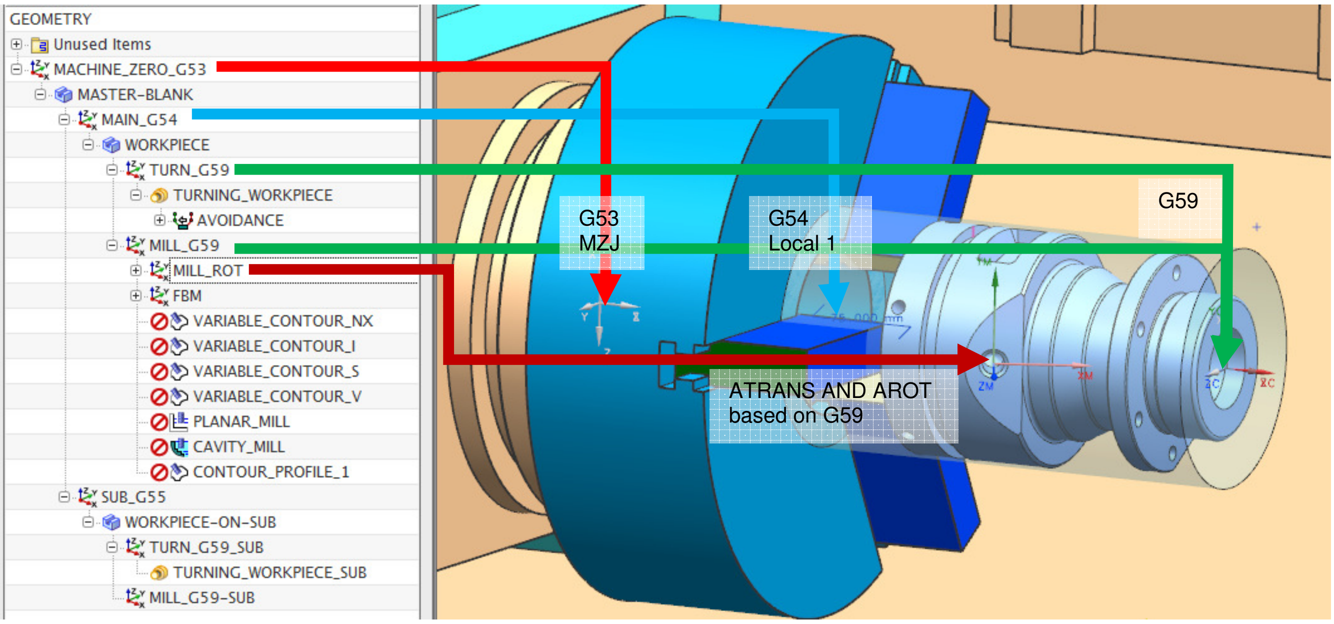
Main Spindle and B-Axis Head

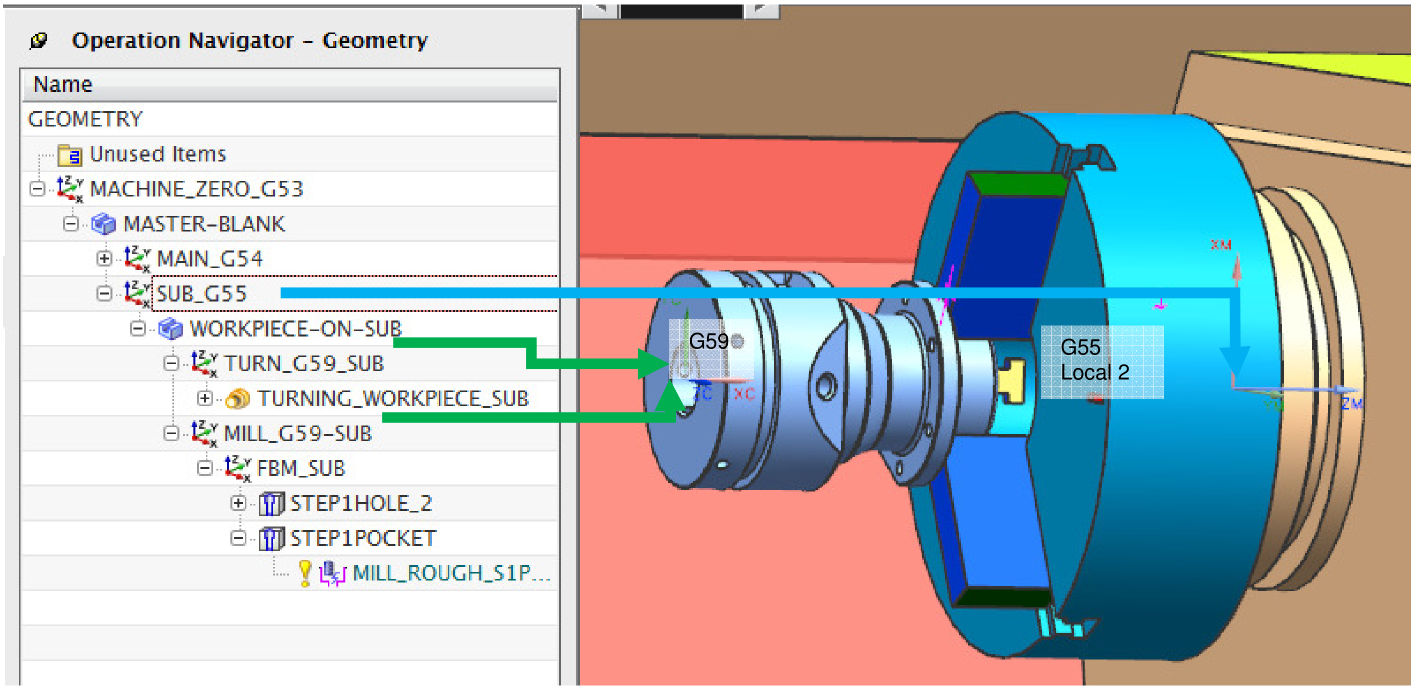


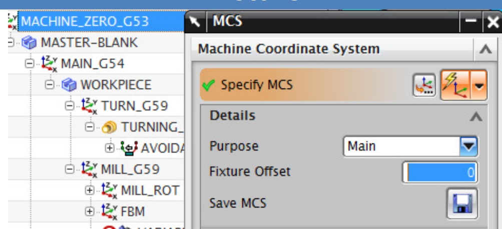
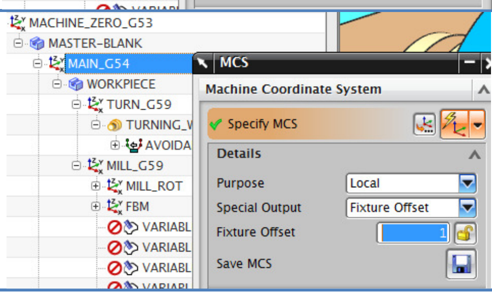

Sub Spindle and Lower Turret

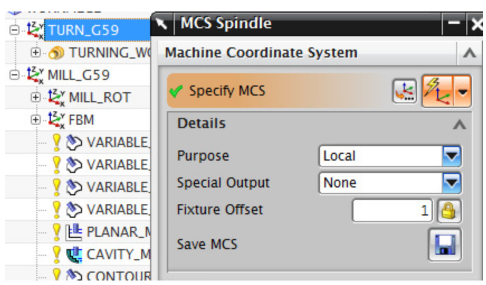
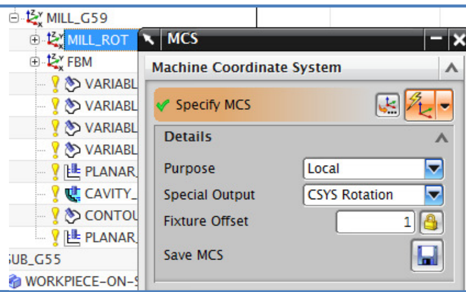
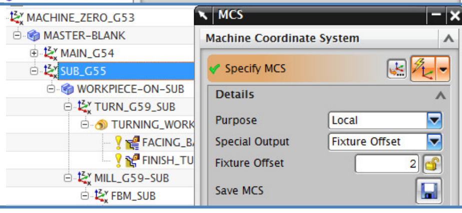


## 13.2 ONT Coordinate systems setup




| MCS | Description | Picture |
|---|---|---|
| **G53** | The machine zero point is placed at the center of the main turning spindle<br>MSC is “main” and “fixture offset” number is set to”0” |  |
| **G54** | The Main Offset G54 is placed at the center of the front face at the chuck<br>It is represented as a “local” MCS with “Fixture Offset” number set to“1”<br>(Main refers to the main spindle not to the MCS purpose) |  |

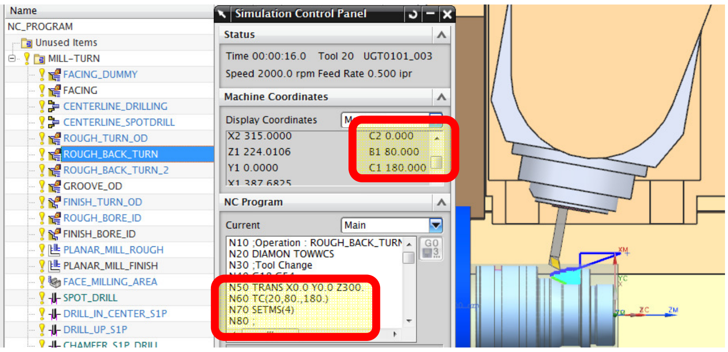
| | | |
|---|---|---|
| **G59** | At the front face of the work piece a MCS is placed. Due to the fact, that for turning and milling different MCS Objects required two of them are added The G59 is an additional offset based on G54. The G59 is similar as a TRANS but will not be reset or overwritten by a new TRANS. TURN_G59 and MILL_G59 are "Local", "´None" and inherit the fixture offset number from the parent. |   |
| **ATRANS AROT** | For arbitrary/ rotated work planes This will be indicated as an additional transformation ATRANS and AROT based on the G59-MCS. |   |
| **G55** | The Main Offset for the sub spindle G55 is placed at center of chuck face on sub spindle. It is represented as a "local" MCS with "Fixture Offset" number set to"2" |   |
| **G59** | Same as above, but for the part on the sub spindle | |

## 13.3 Tool Change

The Toll change on the B-Axis is done by calling a sub program TC.SPF. The parameter definition is defined as:

***TC(tool number, B-Axis position, Spindle position )***

Example:




## 13.4 About Spindles

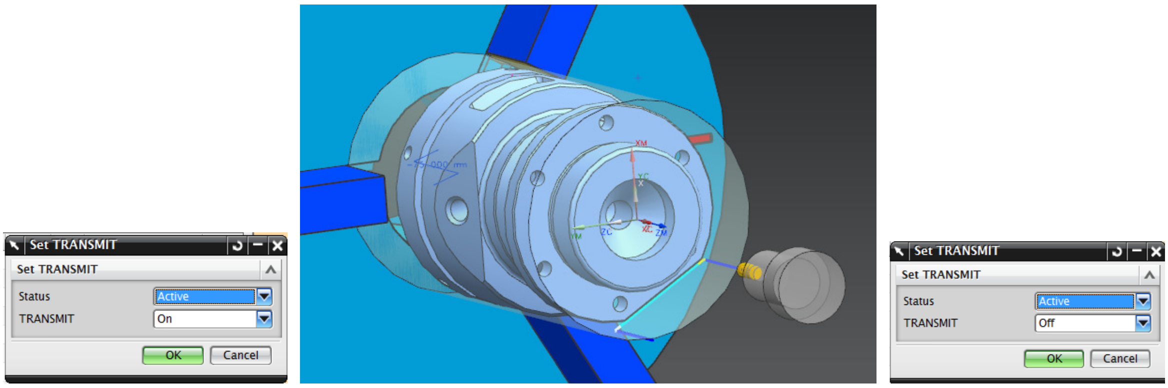
The machine tool has 4 spindles

- C1 – Tool spindle in the B-Axis Head
- C2 – The spindle for life tools on the lower turret
- C3 – The work piece spindle of the sub spindle - right side
- C4 – The work piece spindle of the main spindle - left side

The post outputs related code to activate (M3/M4) and stop (M5) them. At the beginning of an operation the related spindle will be set as main spindle with the SETMS command. At the beginning the main spindle C4 is assigned to the first channel (B-Axis Head) and the sub spindle C3 is assigned to the second channel (lower turret). If necessary to change this the post will output the command GETD(channel) to assigned the related spindle/axis to the channel which drive this spindle/axis

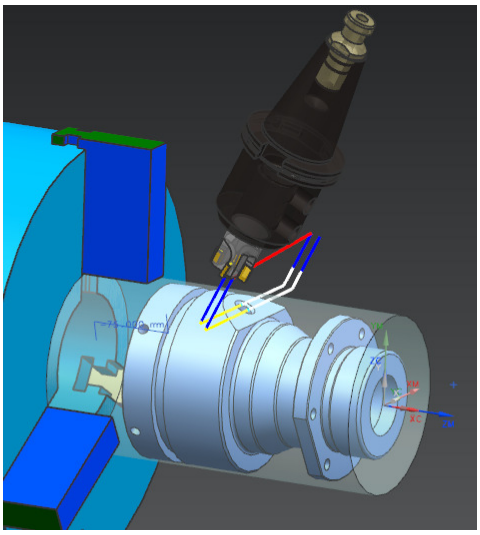
## 13.5 TRANSMIT Operation and other milling motion types

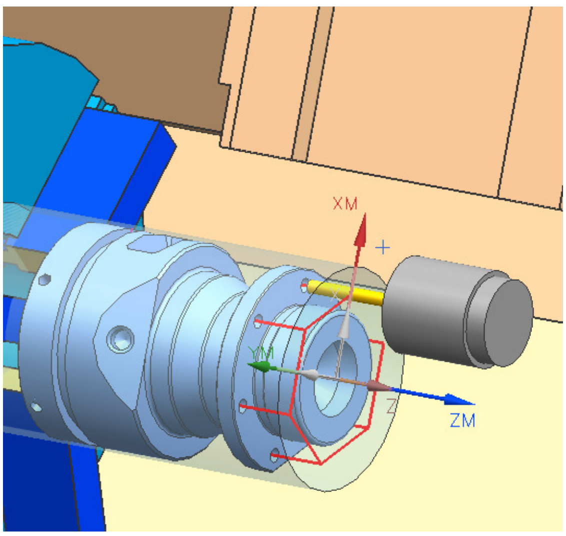
The example includes the TRANSMIT option (CX Polar Mode) for one operation. To activate and deactivate a start and end UDE needs to be used.




```
N5520 ;Operation : PLANAR_MILL_ROUGH
N5530 GETD(C4)
N5540 ;Tool Change
N5550 G17
N5560 DIAMOF TOWSTD
N5570 TC(5,0.,0.)
N5580 MSG("MILL_TRANSMIT")
N5590 SETMS(1)
N5600 ;
N5610 ;Initial Move
N5620 G0 B0.0 C4=0.0
N5630 TRANSMIT(1)
N5640 G54
N5650 TRANS X0.0 Y0.0 Z300.
N5660 G0 G90 X-42.544 Y-70.87 Z10. S1=2000 D1 M1=3
N5670 ;Approach Move
N5680 Z-52.
N5690 ;Engage Move
N5700 G94 G1 Z-55. M8 F200.
N5710 X-45.041 Y-66.538
N5720 ;Cutting
N5730 X-80.152 Y-5.636
N5740 ;Retract Move
N5750 X-82.649 Y-1.304
N5760 Z-52.
N5770 ;Departure Move
N5780 G0 Z10.
N5790 ;End of Path
N5800 TRANS X0 Y0 Z0
N5810 TRAFOOF
N5820 M1=5
```

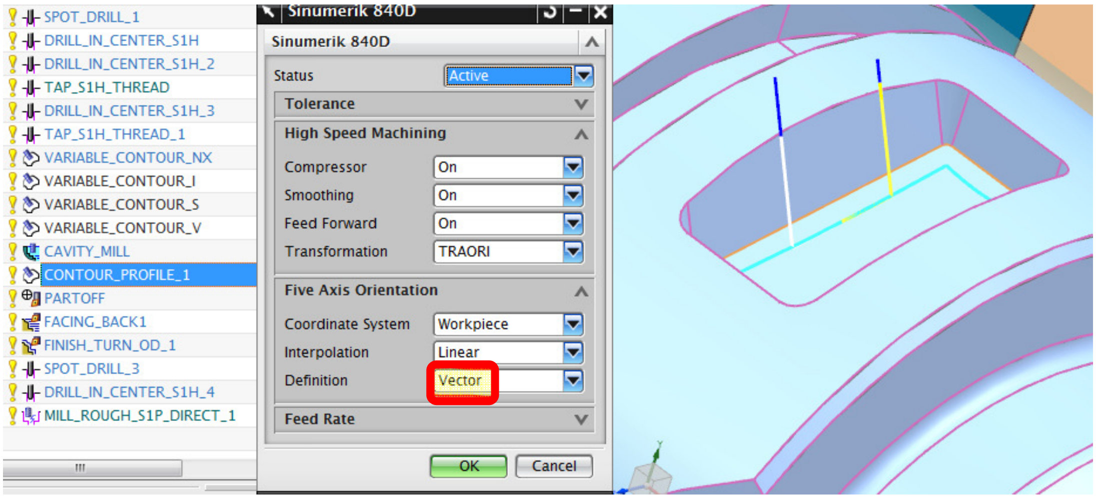
For other milling modes like XYZ motions (example operation is FACE_MILLING_AREA) and for axial drilling XYC motion (example operation DRILL_IN_CENTER_S1H) the post figure out this automatically by the operation.





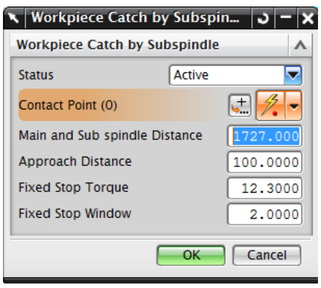
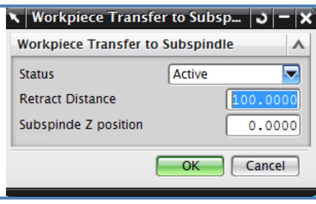
## 13.6 TRAORI simultaneously 5-Axis milling

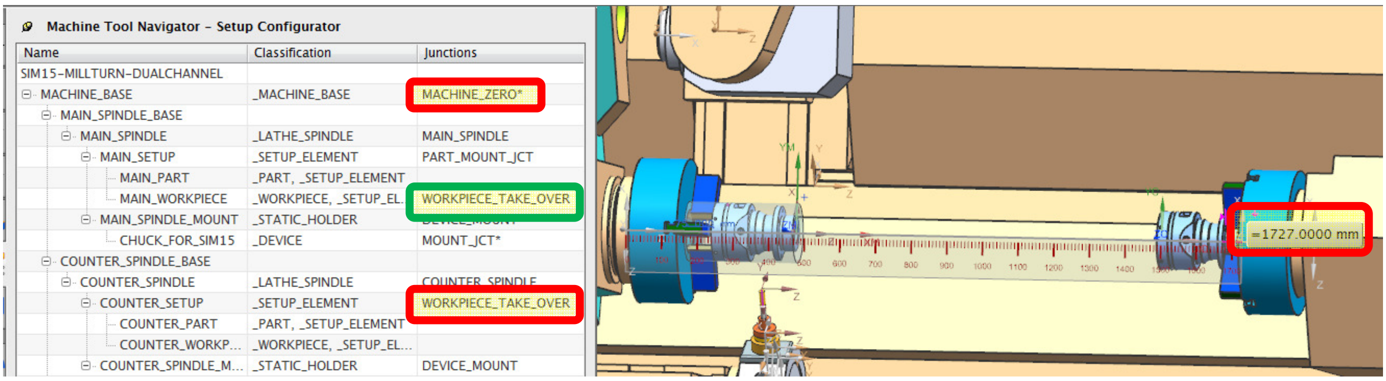
The example includes some multi axis operation, which simulated using the TRAORI function. As a show case one operation (CONTOUR_PROFILE_1) are created Vector output instead of Axis positioning. To achieve the vector output the following UDE is used.




## 13.7 Part transfer from Main to Sub Spindle

The part transfer is defined with two UDE one for start and one for end.

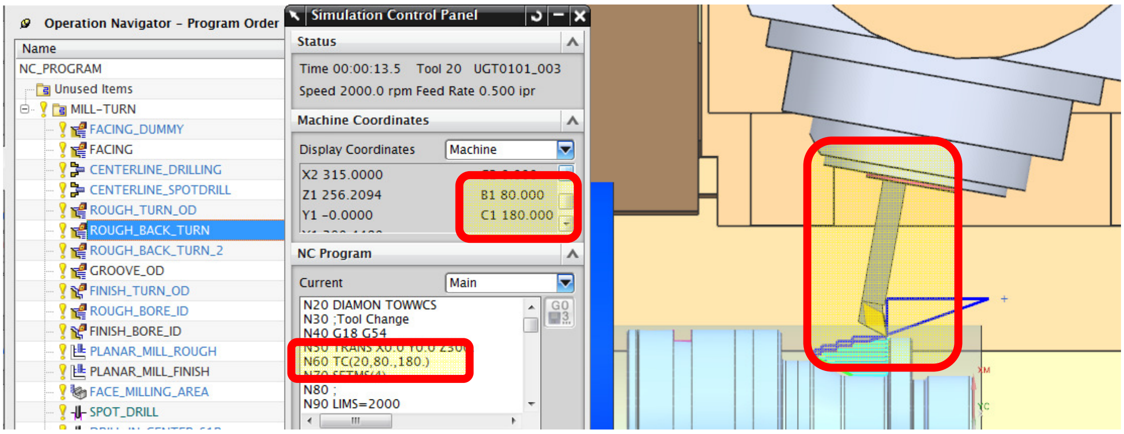
| UDE description | Figure | NC Code Example |
|---|---|---|
| **Contact point**: define the contact point on the work piece on the main spindle<br>**Main and Sub spindle Distance**: The distance between the part mount junction on the sub spindle and the machine zero (G53).<br>See Picture below<br>**Approach Distance**: Distance to where sub spindle move in rapid mode.<br>**Fixed Stop Torque**: used for FXST<br>**Fixed Stop Window**: used for FXSW |   | N50 M130; subspindle chuck open<br>N60 R200=-1250.<br>N70 GETD(C3)<br>N80 G0 C3=0 C4=0<br>N90 G0 G53 Z3=R200+100.<br>N100 G1 G53 Z3=R200 F100. FXS[Z3]=1 FXST[Z3]=12.3 FXSW[Z3]=2.<br>N110 ERR1: IF $AA_FXS[Z3] <> 2 GOTOF CONT1<br>N120 MSG("Travel to fixed stop not reached")<br>N130 M0<br>N140 STOPRE<br>N150 GOTOB ERR1<br>N160 CONT1:<br>N170 M131; subspindle chuck close |
| **Retract Distance**: This is the distance for the retract motion after part cut off.<br>**Subspindle Z position**: This defines the final position of sub spindle Z3 |   | N280 M140; mainspindle chuck open<br>N290 G1 G53 Z3=R200+100. F100. FXS[Z3]=1<br>N300 G0 G53 Z3=0.0 |



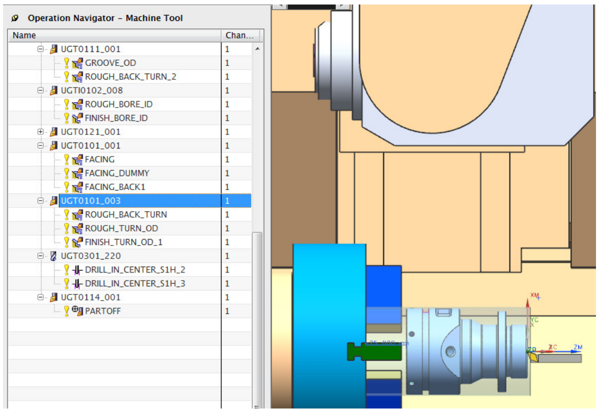

To achieve the mounting of the work piece from the main to the sub spindle the subprogram part_takeover.spf located in the subprog folder of sim15 takes the responsibility. The kernel command deal with junction names, so that it is important to have the correct names of the junctions. The two junctions with the name WORKPIECE_TAKE_OVER are used in this subprog and need to be placed and named correctly in the CAM setup.

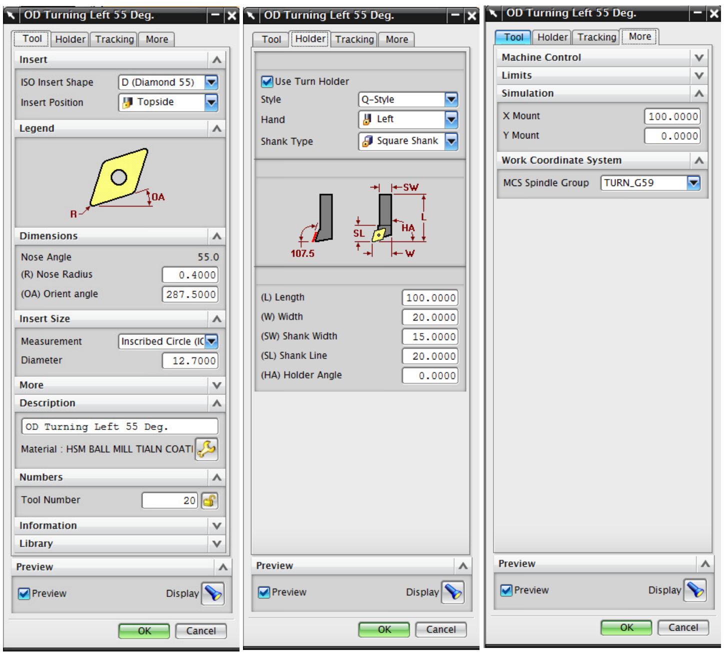
## 13.8 Definition of Tools and related Settings (not specific for sim15)

This chapter explains the dependencies between tool angle data, the related orientation and mounting in to a machine tool. This will be used to created specific operations and achieve the desired orientation. Even if this is done based on one operation of the sim15 example, this is not specific but can be seen as some more general CAM tool related information. As the example we are using the operation ROUGH_BACK_TURN. Desired is the following manufacturing.

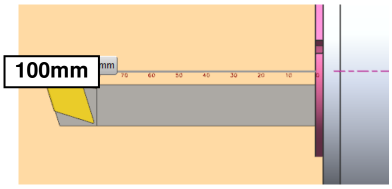



### 13.8.1 Settings in the tool



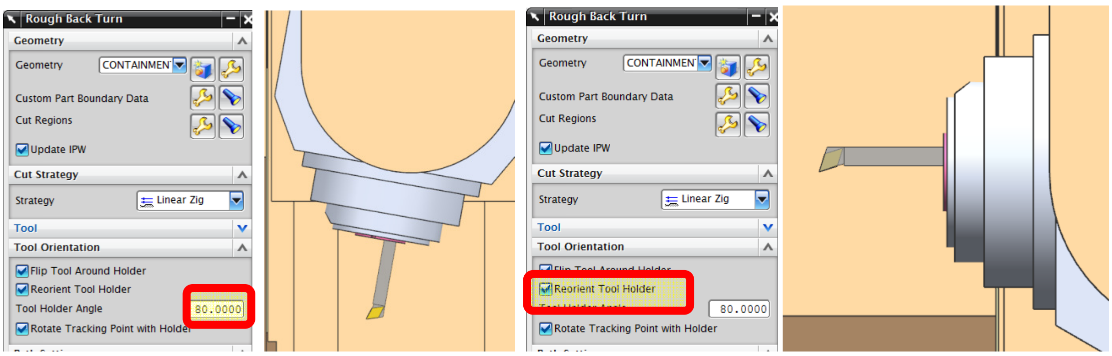



Selecting the tool in the ONT the tool will be displayed with the tracking point on the WCS in that orientation as it will be mounted into the pocket. In the case above this is the B-Axis head. To achieve the correct orientation the tool holder angle (OA), the insert position and the tool holder type can be changed accordantly. Usually the turning tool (if it is a parameter tool without tool assembly graphic) is mounted into the spindle with the tracking point. To change these values in the tab “More” section “Simulation” are used. In this example 100 / 0 will result in:




### 13.8.2 Settings in the Operation:

To achieve the desired position for the operation the tool needs to be more orientated vertical instead of horizontal (B-Axis motion) this will be set into the operation. Here to achieve 80 degree. To achieve the rotation about 180 degree in the spindle of the B axis, so that the cutting edge is more on the right as on the left, the “Flip Tool Holder is used.




# 14 Creating your own CSE examples based on the OOTB mach examples

It is possible to re-use the OOTB examples to build your own machine tool simulation by either selecting an alternative controller (SINUMERIK, Fanuc and TNC) or a comparable machine tool. If the target machine tool is too complex to be derived from one of our OOTB examples, please contact you Siemens PL service representative.

# 15 Appendix

## A. Example NC code TRAORI and circles

The used NC code example of TRAORI and circle handling

```
N90 G40 G17 G710 G94 G90 G60 G601 FNORM
N340 T="UGT0203_005"
N350 M6
G54
D6
G0 X0.0 Y0.0 Z0.0
; This NC code example should explain and demonstrate how circle handling
; and TRAORI plays together.
;
; TEST A:
; Position to a point on the geometry
G0 X26.750670246 Y42.242178464 Z0.0
; make a circle
F1000
G3 X26.750670246 Y-42.242178464 I-26.750670246 J-42.242178464
;
; TEST B:
```

```
; rotate the table including the part and do the motion again.
G0 B-60 C0.0
G0 X26.750670246 Y42.242178464 Z0.0
G3 X26.750670246 Y-42.242178464 I-26.750670246 J-42.242178464
; For sure the motion is the same - no one tell the controller to do different
;
; TEST C:
; Activate TRAORI and try again
ORIWKS
TRAORI
G0 X0.0 Y0.0 Z0.0
G0 X26.750670246 Y42.242178464 Z0.0
G3 X26.750670246 Y-42.242178464 I-26.750670246 J-42.242178464
; TRAORI activates the compensation for the moved coordinate system.
; so the circle is done in the rotated plane.
;
; TEST D:
; There are cases where the machine should do the circle still in the XY-Plane
; of the machine tool,
; (not in the rotated plane defined by the rotation and TRAORI

; Deactivate output TRAORI, to have circle in the original (XY) plane
TRAFOOF
G0 X0.0 Y0.0 Z0.0
;
; TRAORI is off
; so the NC code needs to have the offset of the rotation calculated (!).
;
; The values are based on the rotation -60° around B and 0° around C
; on the position of the original coordinate system in respect to the rotary
; NC axis (kinematics model)
;
G0 X141.506583681 Y0.0 Z-254.902851038
G0 X=141.506583681+13.375335123 Y=0.0+42.242178464 Z=-254.902851038-23.166760002
G3 X=141.506583681+13.375335123 Y=0.0-42.242178464 I-26.750670246 J-42.242178464

; Summary:
; The NC code examples shows, that the case 'D is general possible,
; but this not supported by the OOTB post processors.
; So the rule here is:
; Activate TRAORI and avoid outputting circle statements G2/G3
; after activating TRAORI and have NC axes rotated

M30
```

# B. Detailed information about ini files

A setup-specific INI file could contain, for example

- a list of setup-specific tools and tool count
- initial machine positions
- default fixture offset register
- inch or metric unit selection

Putting such INI files directly into the mach kit affects all setups that share the same machine. When creating your own copies of the OOTB machine tool examples, consider the following execution order for INI files:

- All INI files in the MACH kit directory
- All INI files in the part working directory
- All INI files in the directory specified by the UGII_CAM_CSE_USER_DIR environment variable

Within a directory, the execution order is specified by the optional last hyphen. If the order is omitted then execution order 0 is implied, and the ordering is further determined alphabetically by programName. Should programName be the same then the entire programName-Channel-executionOrder is compared.

# C. Example of a tool change subprogram

This example is taken from the sim05 Sinumerik. (ToolChange.SPF)

```
G0 G90
```

```
; Set the tool change position values in metric X,Y,Z
R501=800.000
R502=-1000.000
R503=700.000

; check the active unit and change values if inch is in use
IF ($P_GG[13] == 2)
   R501 = R501 / 25.4;
   R502 = R502 / 25.4;
   R503 = R503 / 25.4;
ENDIF

; move to the tool change position
G0 G53 X=R501 Y=R502
G0 G53 Z=R503
;Activate AC and do the tool mount based on preselected tool data
##LANGUAGE AC
  INT nToolID;
  STRING sToolName;
  nToolID   = getVariable("$P_TOOLP");
  sToolName = getArrayElement("$TC_TP2",nToolID);

  IF (sToolName != "");
     generateTool (sToolName, "S");
  ELSE;
     IF (nToolID > 0);
        generateTool (getToolNameByNumber(nToolID), "S");
     ELSE;
        // ERROR no Tool preselected - ?? Send error message ??
         ENDIF;
  ENDIF;

  IF (exist(getCurrentTool("S")));
    collision  (OFF, getCurrentTool("S"));
    visibility (     getCurrentTool("S"), OFF, TRUE);
    release    (     getCurrentTool("S"));
  ENDIF;

  IF (exist(getNextTool("S")));
    grasp      (     getNextTool("S"), getJunction("SPINDLE", "S"));
    position   (     getNextTool("S"), 0.0, 0.0, 0.0, 0.0, 0.0, 0.0);
    visibility (     getNextTool("S"), ON,  TRUE);
    collision  (ON,  getNextTool("S"), 2, -0.01);
    activateNextTool ("S");
  ENDIF;
; switch back to Sinumerik syntax
##LANGUAGE NATIVE

; return t o main program
RET
```

# D. Post Processor features of the OOTB examples

This chapter describes the supported functions of the example posts of the MACH library machine tools. All post processors are available as inch and metric versions.

**Milling Machine Posts**

Standard functions for 3-5 milling machines:

- Standard linear, circular and rapid motion (G0,G1, G2 and G3)
- Standard drilling cycles supported by NX. (G80-G89/CYCLE81-CYCLE89)
- Cutter compensation (G40, G41 and G42)
- Tool length compensation (G43/G43.1 for Fanuc)
- Multiple plane circular interpolation (G17, G18 and G19)
- Work coordinate offsets (G53-G59 /G505…/G54.1 P1… etc )
- Spindle control (M03, M04, M05, S)
- Coolant (M08, M09)
- The five- axis posts support coordinate system output on machines that have two rotary axes. This is commonly a G68/ROT/CYCLE SPATIAL. The posts will create a local coordinate system on the fly using the tool axis.
- Five axis tool tip control G43.4/TRAORI/M128

**Turning Machine Posts**

Standard functions for two axis lathes.

- Standard linear, circular and rapid motion (G0,G1, G2 and G3)
- Standard centerline drilling cycles supported by NX (G80-G85)
- Cutcom (G40, G41 and G42)
- Work coordinate offsets (G53-G59)
- Spindle control (M03, M04, M05)
- Coolant (M08, M09)

**Specific Functions in Sample Posts**

Specific functions supported:

- ini file will be generated by post for SINUMERIK controller including fixture offset values and tool information.
- Remove the tool at end of program for milling machine: example: T0 M06
- For 4 or 5 axis machine, rotary axis limit setting in postprocessor should be same as the machine model.
- For SINUMERIK machine, tool offset value D is decided by adjust register number in CAM setup.
- Fixture offset registers range
  SINUMERIK G54-G57 G505-G599
  Fanuc G54-G59 G54.1 P1 G54.1 P2….
- Fixture offset number in CAM setup will decide fixture offset output.
- For SINUMERIK machine, if number is between 1 and 4, corresponding output is G54-G57, if number is 5 output will be G505 and so on.
- For Fanuc machines, if number is between 1 and 6, corresponding output is G54-G59, if number is greater than 6, G54.1 Px will be output. X = number -6.
- Tool tip control
- TRAORI and M128 are similar function in Sinumerik and Heidenhain T530. They are both kinematics independent, means mom_mcs_goto should be output instead of mom_pos for X Y Z position.
  But for OOTB Fanuc examples, G43.4 only has capability to compensate tool axis length in variable axes milling operations. G43.1 should be used in fixed axes milling with head rotation. G43 is used in all operations with Z orientation spindle.

## E. Example of an INI file

```
CHANDATA(1)

; define the offsets G54 is for $P_UIFR[1]
; define the offsets G55 is for $P_UIFR[2]
; define the offsets G...is for $P_UIFR[...]

$P_UIFR[1]=CTRANS(X,100.0,Y,80.0,Z,110.0)
;
; What is needed to place tool data into this file without having alarms.
; This is related to the tool handling mechanism in the archive.

; 1. The used tool ID is needed to be assigned to a magazine
; Therfore use:
; TC_MPP6[1,2]=100
; This places the tool with the number 100 to the second pocket
; on the first magazine
;
; Set the tool name or number used inside the NC code to the tool 100
; $TC_TP1[100]=1
; $TC_TP2[100]="1401631125" ; Tool identifier in the NC code
; $TC_TP7[1]=1              ; Magazin number for the tool
; $TC_TP8[1]=10             ; Release status of the tool
;                             set Bit 1 - release and bit 3 tool is measured.
;                             this is the default on the machine tool
;
; Set tool offset values for the related D number of the tool 100
```

```
; $TC_DP1[100,1]=120    ; tool type 120->milling
; $TC_DP2[100,1]=133.00 ; flute length
; $TC_DP3[100,1]=133.00 ; length
; $TC_DP6[100,1]=68.500 ; radius
; $TC_DP7[100,1]=0      ; corner radius

$TC_TP1[1]=1
$TC_TP2[1]="UGT0203_065"
$TC_TP7[1]=1
$TC_TP8[1]=10
$TC_DP1[1,1]=120
$TC_DP2[1,1]=29.00
$TC_DP3[1,1]=109.00
$TC_DP6[1,1]=4.0
$TC_DP7[1,1]=4.0
$TC_MPP6[1,1]=1

;$TX_TOOLCOUNT = 9 ; tool numbers
M17
```

# F. PLANE SPATIAL

Inside the used MCF one global variable is set related to the type of the machine tool.
Name of the variable is:*GV_strMachineType (“HeadHead”, “HeadTable”,”TableTable”)*

## About SEQ+/- parameter in PLANE SPATIAL

Choose the shortest way as the preferred solutions. Implement this in the CCF file. First check if both solutions are achievable – due to limits of the axis. If neither solution is within traverse range, error message will be generated. This mechanism uses the axis limits set in the MCF file.

## About TABLE ROT/COORD ROT in PLANE SPATIAL

Transformation mode is an optional parameter for Plane function and TABLE ROT will used as default if omitted.

## Support PLANE SPATIAL for sim05 TNC

For dual-head machine, except rotation of coordinate, tool tip offset after rotation should be compensated.

- Create coordinate matrix by SPA SPB SPC
- Calculate IKS angles from new coordinate Z vector(two results)
- Call method GMe_GetPlaneSolution to choose rotation angles by SEQ option.
- Get linear offset by calculateIKSlinears
- Reset transformation “PLANE” origin, compensate offset values of setp4 and subtract tool length compensation.
- Set transformation rotation, the order is SPA->SPB->SPC
- Add tool length compensation in modal transformation “PLANE”.

## Support PLANE SPATIAL for sim07 and sim09 TNC

For head-table machine, both of rotation angle and origin point offset and tool tip offset should be compensated in TABLE ROT mode.

- Calculate IKS angles from new coordinate Z vector
- Get linear offset by calculateIKSlinears, if it’s COORD ROT, only consider head ration in calculateIKSlinears, and set Q122 to 0.
- Reset transformation “PLANE” origin, compensate offset values of setp2 and subtract tool length compensation.
- Set transformation rotation, it’s about table C axis rotation.
- Set transformation rotation modal, the order is SPA->SPB->SPC
- Add tool length compensation in modal transformation “PLANE”.

**Support PLANE SPATIAL for sim08 sim06, and sim14 TNC**

For dual-table machine, both of rotation angle and origin point offset should be compensated in TABLE ROT mode.

- Calculate IKS angles from new coordinate Z vector
- Get linear offset by calculateIKSlinears
- Reset transformation "PLANE" origin, compensate offset values of setp2
- If it's TABLE ROT, set transformation rotation, it is rotated about fourth-axis then fifth-axis vector.
- Set transformation rotation modal, the order is SPA->SPB->SPC

**Plane spatial Process in CCF file – figure needs to be updated**

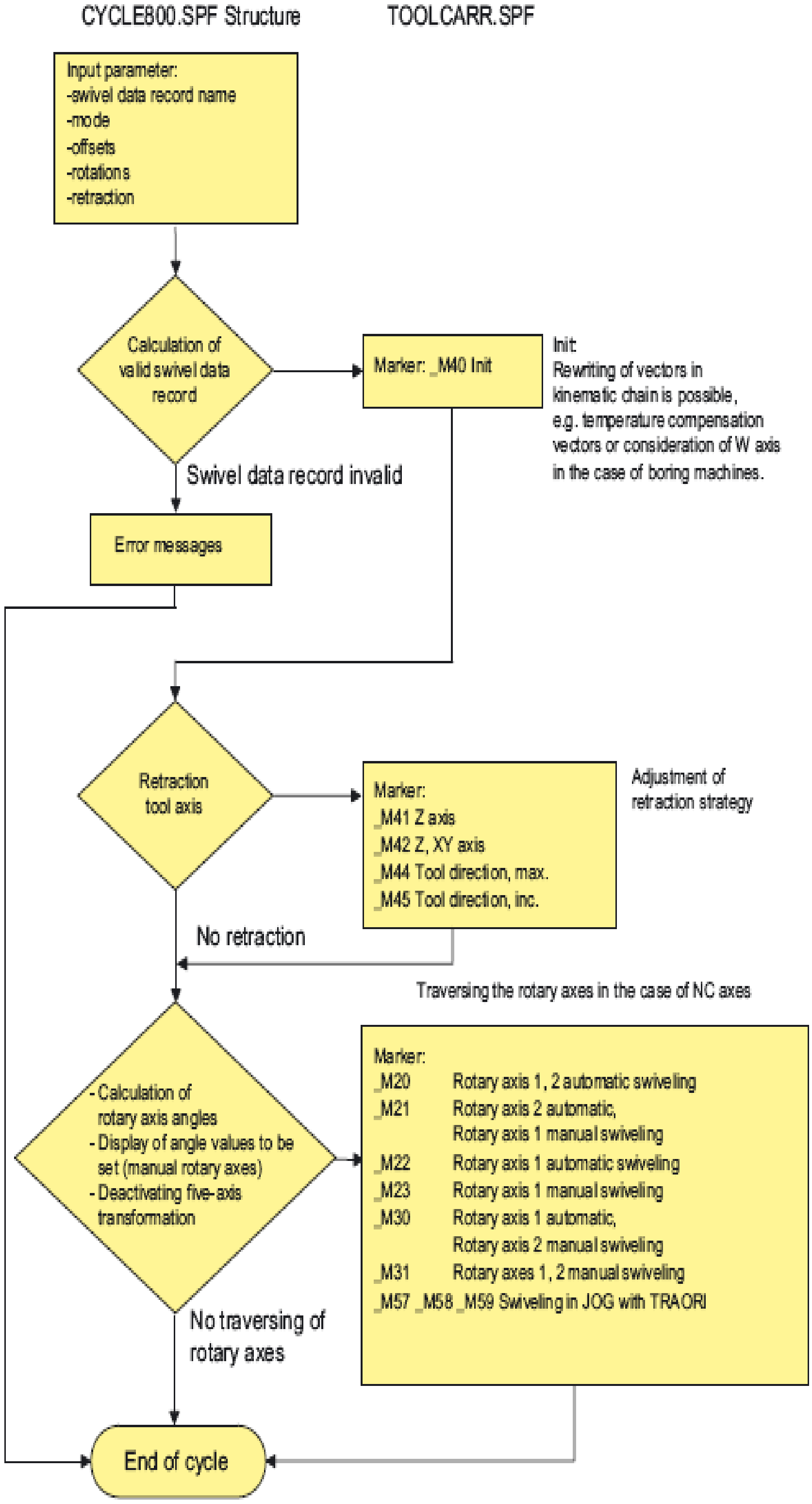
# G. CYCLE800

**Basic Workflow**

The CYCLE800.SPF consists of two subprograms, which are:

- On Powerline: TOOLCARR.SPF
- On Solutionline: CUST_800.SPF

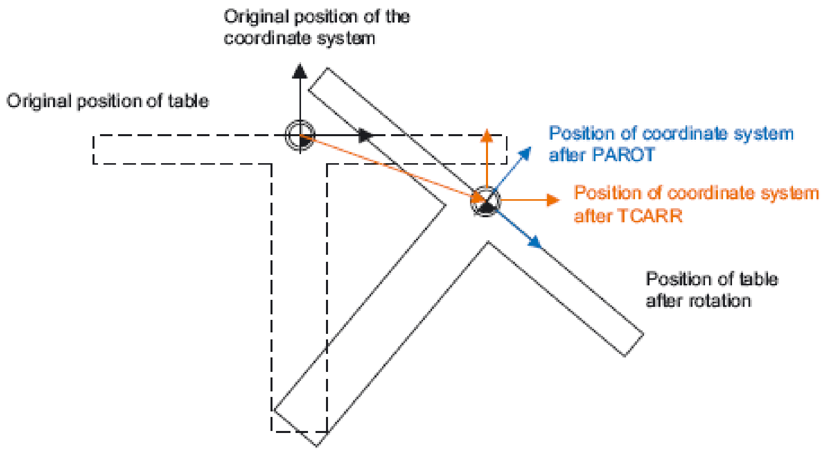
In addition CYCLE800 makes use of lots of system-variables and machine data, e.g. TC_CARRxx[x] that holds the kinematic configuration and also needs some definition files like PGUD.DEF and SMAC.DEF so on Powerline even more.



## Basic about TCARR and PAROT

The subprogram TCARR calculates rotational axis angle, and calculates the translation compensation based on current kinematics, and apply it to the according transformation.teh subprogram PAROT rotates the transformation “PARTFRAME” for the table.
Depending on the mode of $P_GG[42] (TCOABS, TCOFR) the compensation is calculated in TCARR.

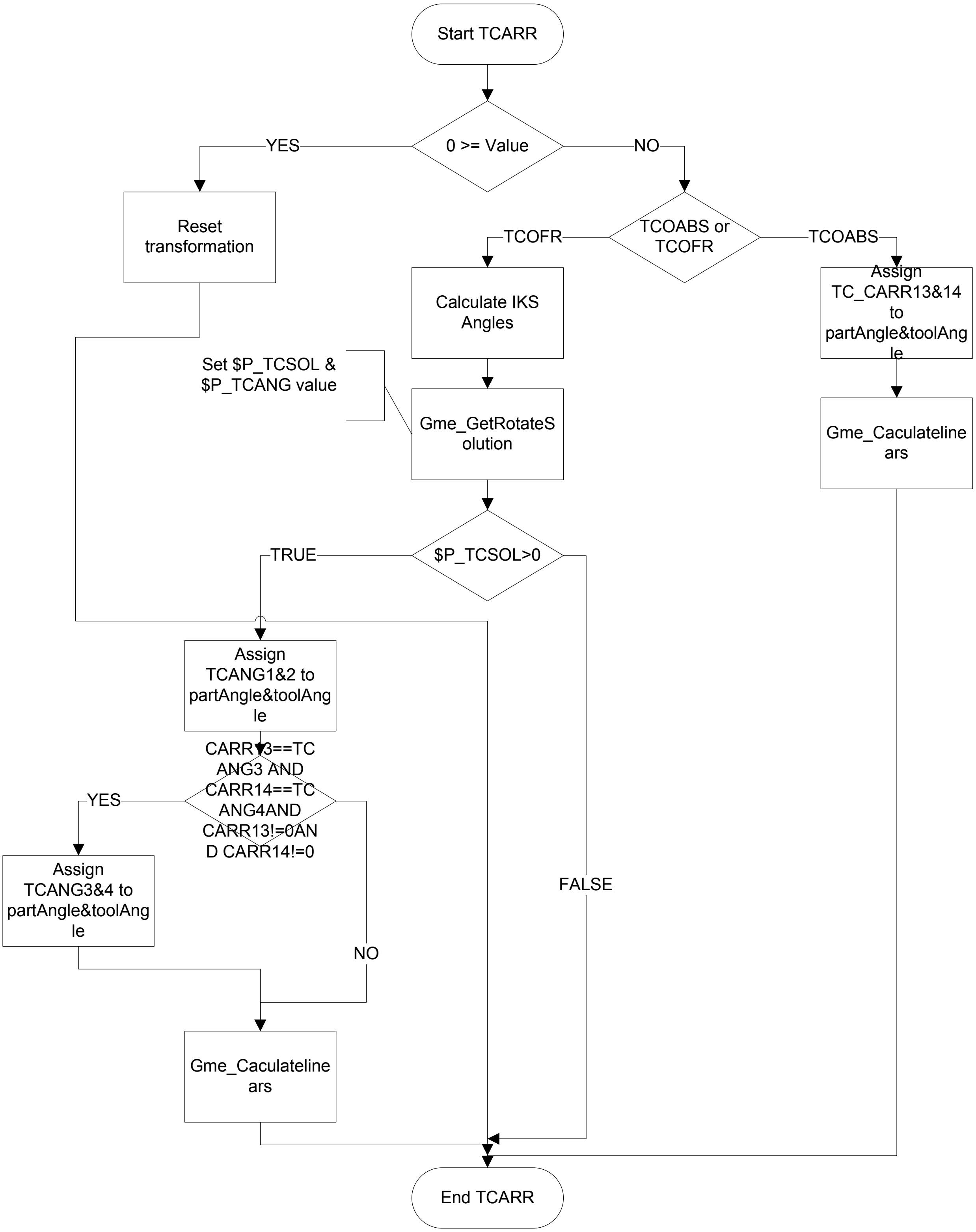
- With TCOABS
  Based on the angles defined in TC_CARR13 and TC_CARR14
- With TCOFR
  Based on the current tool orientation the angles are calculated first → IKS and picker required simCYCLE800 consists of two subprograms: CYCLE800.SPF(Siemens),



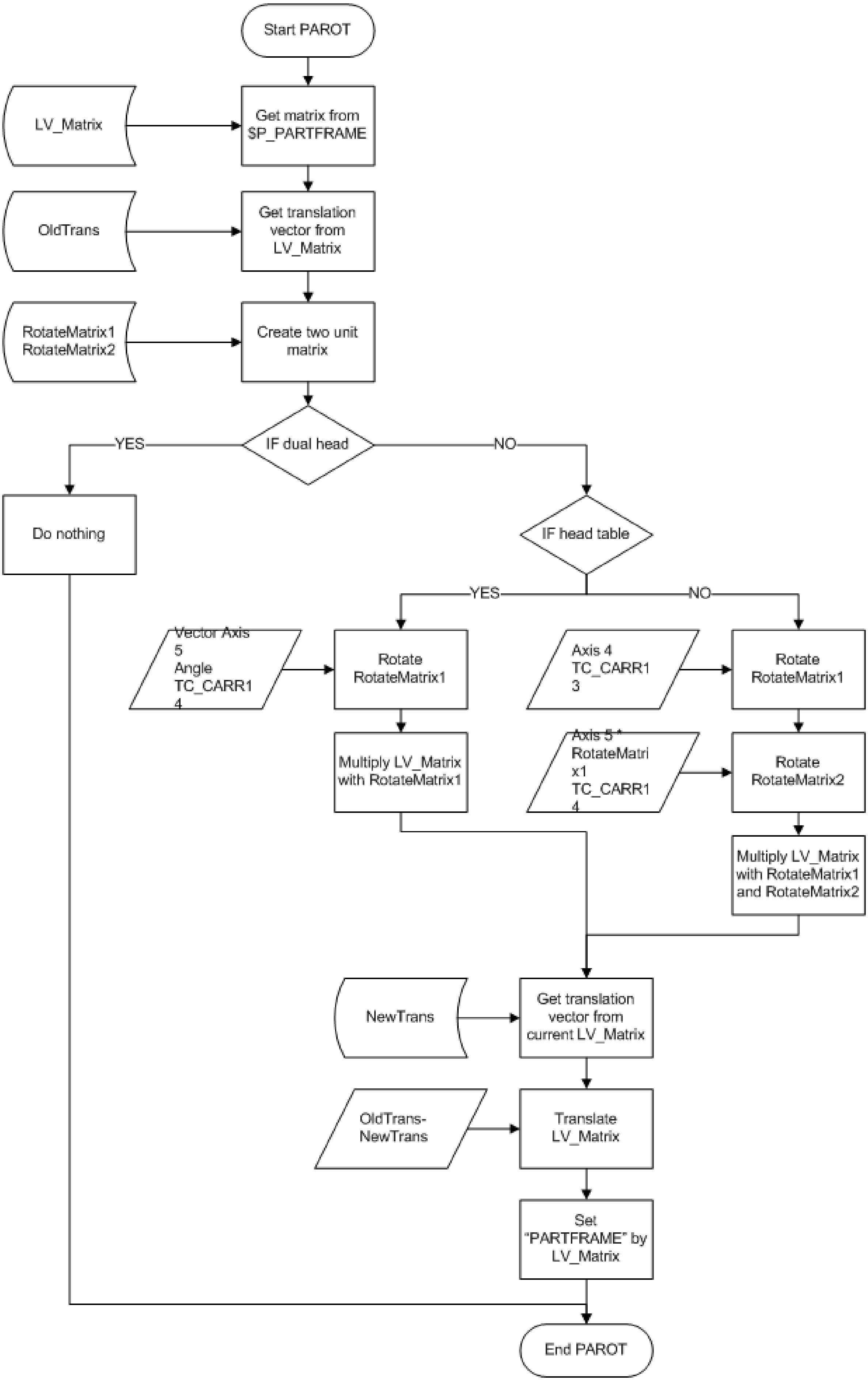

TOOLCARR.SPF(CUST_800 is the name used in current CYCLE800.SPF) It moves and rotates the tool or table according to the input mode. Currently it is implemented for M20 M40 M41 cases.

- M20 Rotate the tool or table according to $TC_CARR13 and $TC_CARR14
- M40 The initiation behavior defined by the user
- M41 Tool retract Axis Z

## Workflow of TCARR



## Workflow of PAROT



## How to set $TC_CARR37

The OOTB examples uses the Solutionline example:$TC_CARR37[1]=201003003

| Decimal position | Meaning | |
|---|---|---|
| ONES | Selects the swivel mode | |
| | 0 = | Axis by axis |
| | 1 = | Axis-by-axis + projection angle |
| | 2 = | Axis-by-axis + projection angle + solid angle |
| | 3 = | Axis-by-axis + direct |
| | 4 = | Axis-by-axis + projection angle + direct |
| | 5 = | Axis-by-axis + projection angle + solid angle + direct |
| TENS | Rotary axis 1 | |
| | 0 = | Automatic |
| | 1 = | Manual |
| | 2 = | Semi-automatic |
| HUNDREDS | Rotary axis 2 | |
| | 0 = | Automatic |
| | 1 = | Manual |
| | 2 = | Semi-automatic |
| THOUSANDS | Selection field, direction: Direction selection of the rotary axes | |
| | 0 = | No, no display of the direction reference for kinematics that only have one solution. Direction selection (_DIR) Minus is generated in the cycle call CYCLE800. |
| | 3 = | Direction reference, rotary axis 1 optimized, direction selection Minus in the basic setting of the kinematics. |
| | 4 = | Direction reference, rotary axis 2 optimized, direction selection Minus in the basic setting of the kinematics. |
| | 5 = | No, no display of the direction reference for kinematics that only have one solution. Direction selection (_DIR) Plus is generated in the cycle call CYCLE800. |
| | 8 = | Direction reference, rotary axis 1 optimized, direction selection Plus in the basic setting of the kinematics. |
| | 9 = | Direction reference, rotary axis 2 optimized, direction selection Plus in the basic setting of the kinematics. |
| | | The values 1, 2, 6 and 7 are not permitted. |

| Decimal position | Meaning | |
|---|---|---|
| TEN THOUSANDS | Selection field, correction of the tool tip or B axis kinematics | |
| | 0 = | No, no display of the Correction of tool tip input field |
| | 1 = | Yes, correction of tool tip by means of TRAORI. |
| | 2 = | No tracking of tool tip + B axis kinematics turning technology. |
| | 3 = | Correction of tool tip + B axis kinematics turning technology.<br>The Correction of tool tip function requires the "5-axis transformation (TRAORI)" option. |
| HUNDRED THOUSANDS | Reserved | |
| ONE MILLION TEN MILLION | Selection field, retraction | |
| | 00 = | No retraction |
| | 01 = | Retraction Z |
| | 02 = | Retraction Z, XY |
| | 03 = | Retraction Z or Z, XY |
| | 04 = | Maximum retraction in tool direction |
| | ... | |
| | 08 = | Incremental retraction in tool direction |
| | ... | |
| | 15 = | Retraction Z or Z, XY or<br>in maximum tool direction or<br>in incremental tool direction |
| | $TC_CARR38[n] | Retraction position X |
| | $TC_CARR39[n] | Retraction position Y |
| | $TC_CARR40[n] | Retraction position Z |
| HUNDRED MILLION | Swivel data set enabled<br>Swivel data set change<br>Tool change | |
| | 0 = | Swivel data set not enabled |
| | 4 = | Swivel data set enabled<br>automatic swivel data set and tool change |
| | 5 = | Swivel data set enabled<br>automatic swivel data set change and manual tool change |
| | 6 = | Swivel data set enabled<br>manual swivel data set change and automatic tool change |
| | 7 = | Swivel data set enabled<br>manual swivel data set and tool change |