SIEMENS

ADDITIONAL MACHINE SIMULATION HELP

WORKING WITH MACH SIMULATION EXAMPLES NX11.0.1

December 15, 2016

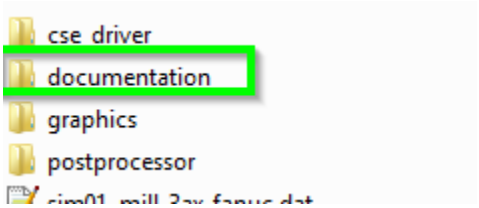
Version #1

1 What´s new with NX11.0.1

CSE-Driver
All OOTB Sinumerik machine tools CSE driver uses the NCK Cycles version 4.7. SP2 HF1.

Siemens840D.CCF – changes in TCARR/PAROT – details in chapter 11.2.3 Machine Tool dependent data – customized ini files -Definition data of the TC variables. → new global variable GV_bSetToolCarrierDataFromKim to define location of toolcarrier data.

Sample Machines
Machine specific information will be found in 'Documentation' folder located in the machine folder



Updated Chapter: 6.2. - Note 5

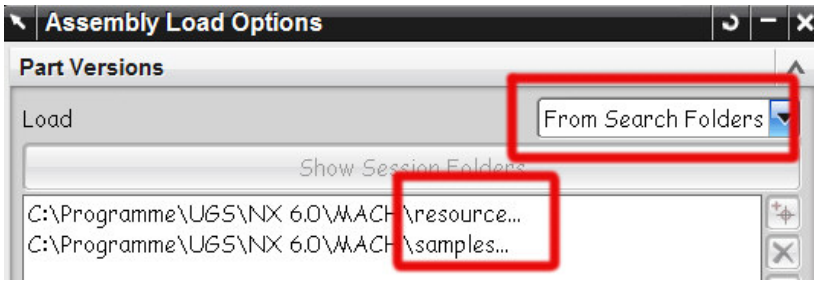
2 Overview

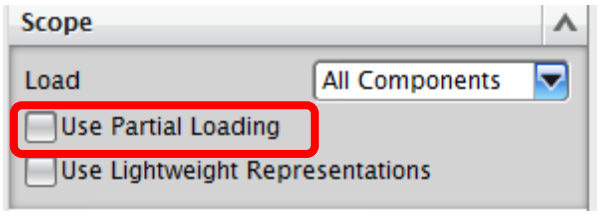
This document explains the handling and usage of the simulation examples provided out-of-the-box (OOTB). The example data is mainly contained in two locations: one is for the library machine tool models in the `$UGII_CAM_BASE_DIR/resource/library/machine/installed_machines` folder, the other for the CAM examples utilizing the library machine tools. These CAM examples can be found under `$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 at least 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. Especially in the case of complex Mill-Turn machine tool like sim15 or the head change machine tool sim16 it cannot be simply re-used for other complex machine tool. Here additional customization in most cases is needed.*

3 Load Options

To ensure all related assembly component parts are loaded correctly it is recommended to use the "From Search Folder" load option with the following search paths. Be certain to include the three dots at the end of each path. Leave the “Use Partial Loading” - option unchecked'

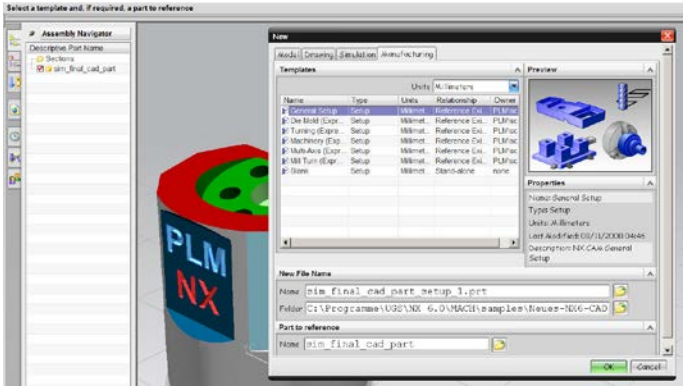




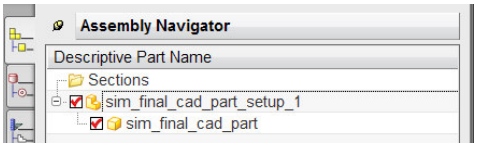
4 CAM-Setup

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.

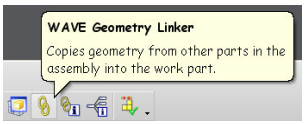
- Open the CAD part file in NX
- Select “New” and pick an appropriate entry in the Manufacturing tab.



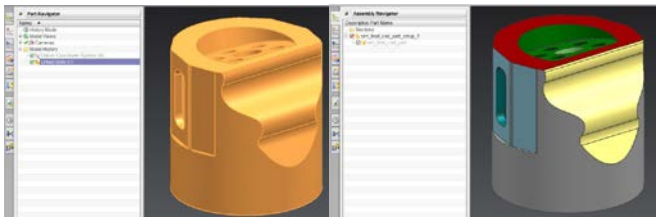
- The system will automatically create a master-model-concept-part-file referencing your CAD geometry.



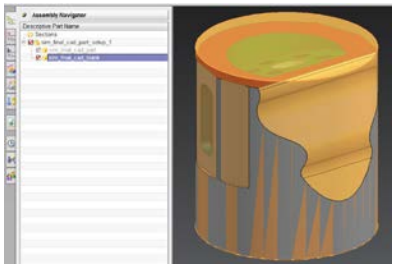
- Create a Wave link to the CAD geometry



- 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.



Starting with NX9 all CAM setups part files in the sample folder are pre-configured to be used as template or seed parts.

Since NX9.0 the CAM setup part files in the samples folder are prepared to be used as template files when entering/create a new CAM setup scenario.

5 MCS-Handling (Best practice for the OOTB examples)

In order to achieve a complete and reliable machine simulation, different modules need to play together. These are:

- CAM programming structure in the Operation Navigator (ONT),
- Internal Post Processor (MOM inside NX),
- TCL based post processor
- Simulation controller model.

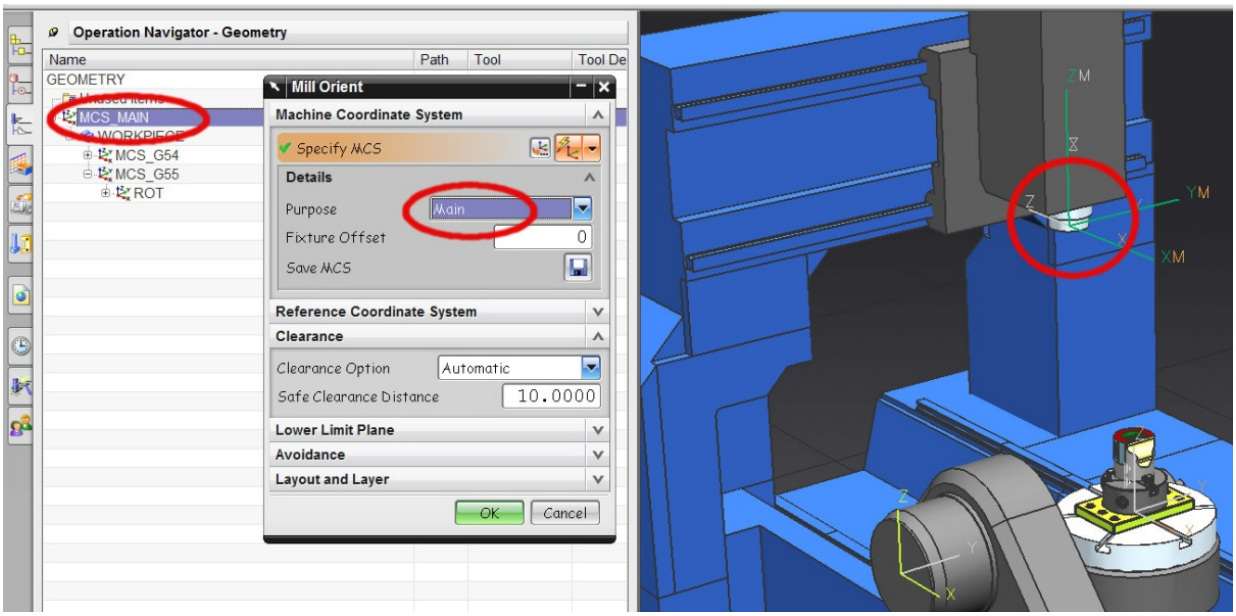
A few rules need to be considered for NX CAM setup to support the assigned post processor creating a valid NC code which can be used at the physical machine tool as well as for simulation.

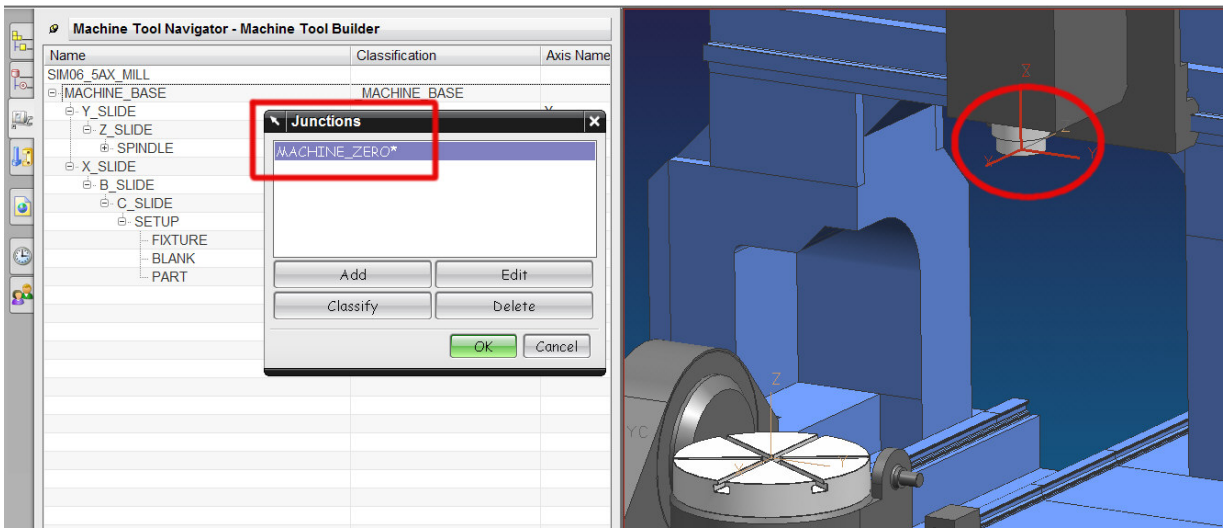
5.1 General settings and rules

- 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 was used to tell the post processor the position of the machine zero position prior NX8.5. Starting with NX8.5 the post has access to the junctions in the kinematics model. For more details refer to Creating file with tool and offset data. The examples still include the MAIN MCS to represent the Machine Zero Position; this is due to planned future improvements and enhancements for a better usability.*





- 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)*

  The Sinumerik Post will create an to_ini.ini file including the offset vector from the local MCS with fixture offset number to the Main MCS and output that like $P_UIFR[2]=CTRANS(…)
  It is not suggested to use fixture offset number 0, which will output a $P_UIFR[0] and that is mainly the machine zero system defined with the Main MCS object

- 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. The fixture offset number must always inherit its value from the parent MCS object.
  TRANS/ROT or CYCLE800 (based on UDE) -> SINUMERIK
  G68 -> Fanuc
  PLANE SPATIAL -> TNC

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	CYCLE800	G68.2	PLANE SPATIAL

More details and example are shown in Appendix: "OOTB MCS Handling output details"

- For the OOTB post, operations must not be placed under MCS objects with the special output "none" or "Use Main MCS". In such cases the post will not identify the correct offset for the operation and the offset creation in the to_ini.ini files will fail.

5.2 Special rules and settings for Mill-Turn machine tools.

Working with Mill-Turn machines the work piece spindle object needs to be represented in the ONT. Therefore a MCS Spindle object is mandatory. That MCS object needs to be aligned with the work piece spindle and the X/Y or Z/X plane represents the lathe work plane. Also the Lathe axis needs to be aligned with the spindle axis. Working with multiple spindles e.g. Main and sub, each spindle needs to be defined by an MCS Spindle object in the ONT.

Additional information about best practice to set up a multi-spindle Mill-Turn geometry view is available in the NX Help under "CAM"→"Manufacturing Turning"→"Geometry structure sample for a 2 spindle multifunction machine"

5.3 Library machine tool specific handling

- SIM15 post rule for local MCS object with special output “None”
  A local MCS with special purpose “None” located as child of a local MCS with special purpose “Fixture Offset number” will be used to output G59. See more in the chapter about sim15

6 About Post Processors

All the OOTB delivered post processors are created with the Post Builder Version 9 based on the Sinumerik S840D basic template post. Some differences between pure template post and OOTB post are listed in the 3rd subchapter.

6.1 General supported features of the OOTB Posts

All three major controllers support the basic functionality for 2 axis drilling, 3 axis, 3+2 axis and 5 axis milling and also some advanced functionality.

6.1.1 Basic functionality

- Addresses (X, Y, Z, A, B, C)
- Program control (M30, M0...)
- Basic motion type: rapid, linear, circular
- Metric/Inch units
- Radius/diameter programming
- Fixture offset
- Coordinate system rotation(support Local CSYS and Auto3D way)
- (G68, TRANS/AROT)
- Tool radius compensation
- Tool length compensation
- Feed rate, feedrate unit
- Spindle mode and spindle speed
- Constant surface speed
- Tool change
- TCP (TRAORI, G43.4, M128)
- Drilling cycle

6.1.2 Advanced functionality:

- Swiveling function (G68.2, CYCLE800, PLANE SPATIAL)
- Variable axis drilling (fully support cycle plane change)
- Turning cycle (CYCLE95, G71)

6.1.3 Sim15 Mill-turn (S840D and Fanuc)

- One mill-turn post
- Part transfer
- Polar mode
- Drilling XZC output

6.2 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.ini or 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 a 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".

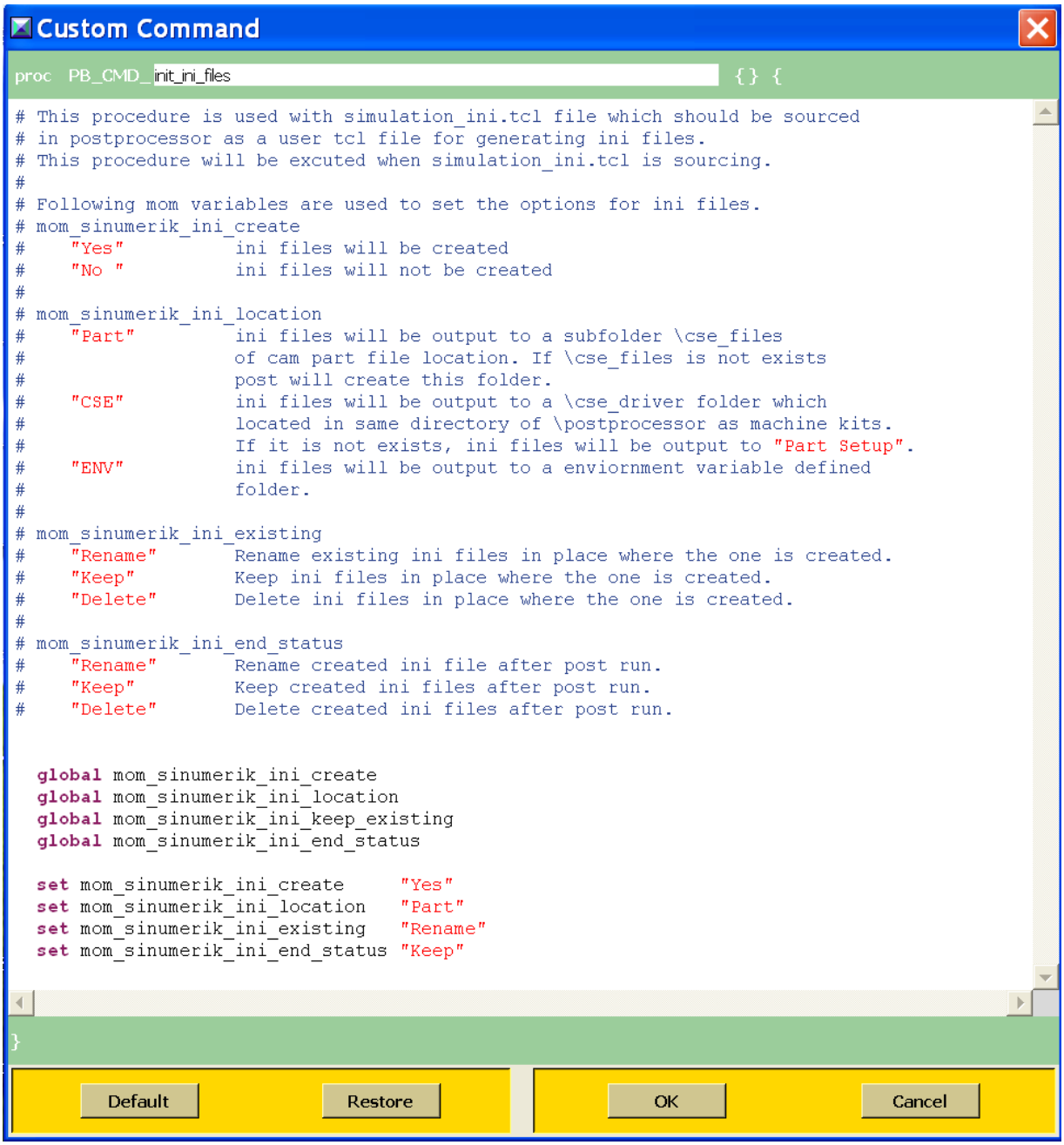
Note 1: *Since NX85 the post query the kinematics model using the TCL query function* **MOM_ask_init_junction_xform** *and store the results in the variable* **mom_sim_result[9], mom_sim_result1[3]** *to get the data of the machine zero junction needed to calculate offset information for e.g. G54.*

Note 2: *With the Sinumerik template Post within PostBuilder Version 8 or newer you are able to configure the behavior of creating the ini file.*

Note 3: *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 in the details window, that the related TO_INI file could not be found. The simulation will not stop in that case.*

Note 4: *New with NX11 is that the location of ini file will be reset to directory which defined by UGII_CAM_CSE_USER_DIR, if this environment variable is already set up in system.*

Note 5: *New with NX11 the values inside to_ini.ini are in the same unit as the ones inside the main program. Using older versions of the post in inch modus the output of to_ini.ini will be wrong running with the newest CSE driver. If they are not working, use one of the new standard posts provided with NX instead.*



6.3 MOM variable for MCS handling

For coordinate rotation NC codes output inside the post, the related coordinate matrix is changed. 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 postprocessor.

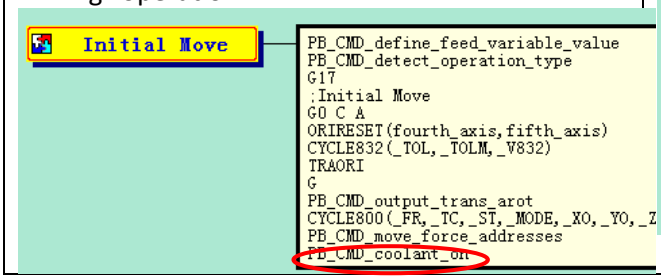
6.4 Differences between 'pure' template posts and OOTB posts (Sinumerik)

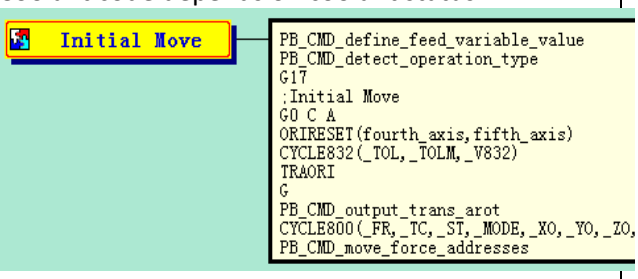
The OOTB posts are created based on the latest Sinumerik S840D basic template posts. Some settings and or features are slightly different, which are listed here.

6.4.1 Coolant

Coolant output is different.

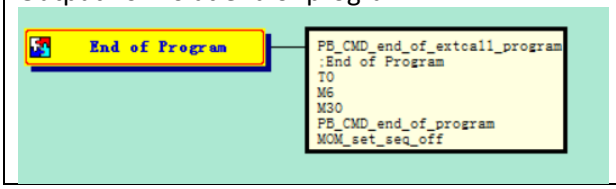
Posts of OOTB Examples	Post from template post
Coolant code always be M8 , except "Deep Hole Drilling" operation.	Coolant code depends on coolant status

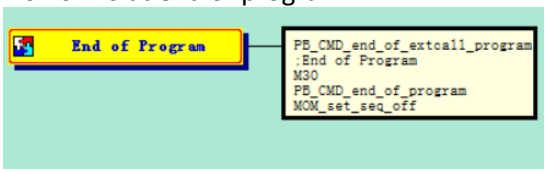




6.4.2 Output of "T0 M6" at end of program

Posts of OOTB Examples	Post from template post
Output T0 M6 at end of program	No T0 M6 at end of program





6.5 About UDEs and OOTB

The post shipped with the OOTB machine tools in the library are coming with their own UDEs (CDL file), which provides a series of UDE which all are supported.

6.6 Post output for Coolant (Sinumerik, TNC, Fanuc)

OOTB post processors only support M8 (coolant on) and M9 (coolant off), to output other coolant codes, post processors need to be customized.

6.7 About Special handling for 5 axis motions with Fanuc for table machine tool configuration

The OOTB setting for Fanuc controller will not move the programming system together with the table when doing 5 axis motions like G43.X. The post takes care about that when outputting the X/Y/Z data.

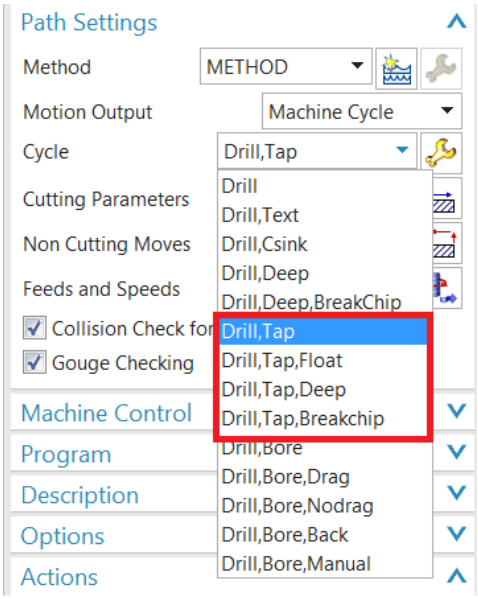
The CSE controller takes care about that by using a specific mode when activating the 5 axis transformation. This mode can be changed:
In Post do the following:

Change variable dpp_ge(sys_output_coord_mode) value to “TCP_FIX_MACHINE” in command PB_CMD_customize_output_mode

In Simulation do the following:
Change the setting of the variable #19696 in the INI file to =0 (default is 32 for non-rotating workpiece coordinate system)

6.8 Parameter Setting for new tap cycles

In the NX version, cycle type “Drill, Tap, Float”, “Drill, Tap, Deep”, and “Drill, Tap, Breakchip” are introduced. Please see the picture below. So accordingly, some enhancements are implemented in OOTB post processor. The detail will be showed in the content below.



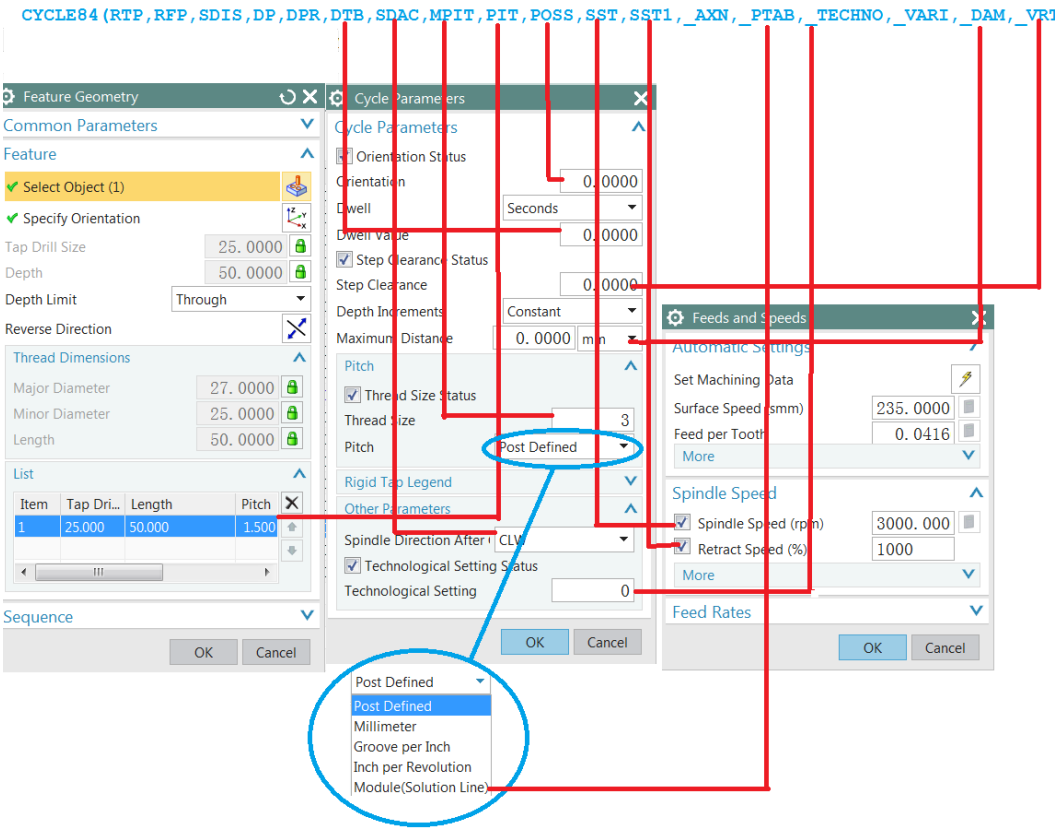
The table below shows the tap cycle of three kinds of controller.

Tap cycle / Controller	Tap	Tap Float	Tap Deep	Tap Breakchip
S840D	CYCLE84(PTP,...VARI,...) **NOTE:VARI=0**	CYCLE840(xx,...)	CYCLE84(PTP,...VARI,...) **NOTE:VARI=2**	CYCLE84(PTP,...VARI,...) **NOTE:VARI=1**
Fanuc	M29 S G84.2/G84.3X Y Z R P F	G84/G74X Y Z R P F	M29 S G84.2/G84.3 X Y Z R P Q F **NOTE: PCP(No.5200)=0**	M29 S G84.2/G84.3 X Y Z R P Q F **NOTE: PCP(No.5200)=1**
ITNC	CYCL DEF 207 Q200 Q201 Q239 Q203 Q204	CYCL DEF 206 Q200 Q201 Q206 Q211 Q203 Q204	CYCL DEF 209 Q200 Q201 Q239 Q203 Q204 Q257 Q256 Q336 **NOTE:Q256=0**	CYCL DEF 209 Q200 Q201 Q239 Q203 Q204 Q257 Q256 Q336 **NOTE:Q256 !=0**

NOTE: *In OOTB Fanuc post, it uses tap cycle of series15 format.*

6.8.1 Set CYCLE84 parameter for S840D in UI

Take tap break chip UI as an example for Powerline CYCLE84. Please see the picture below.

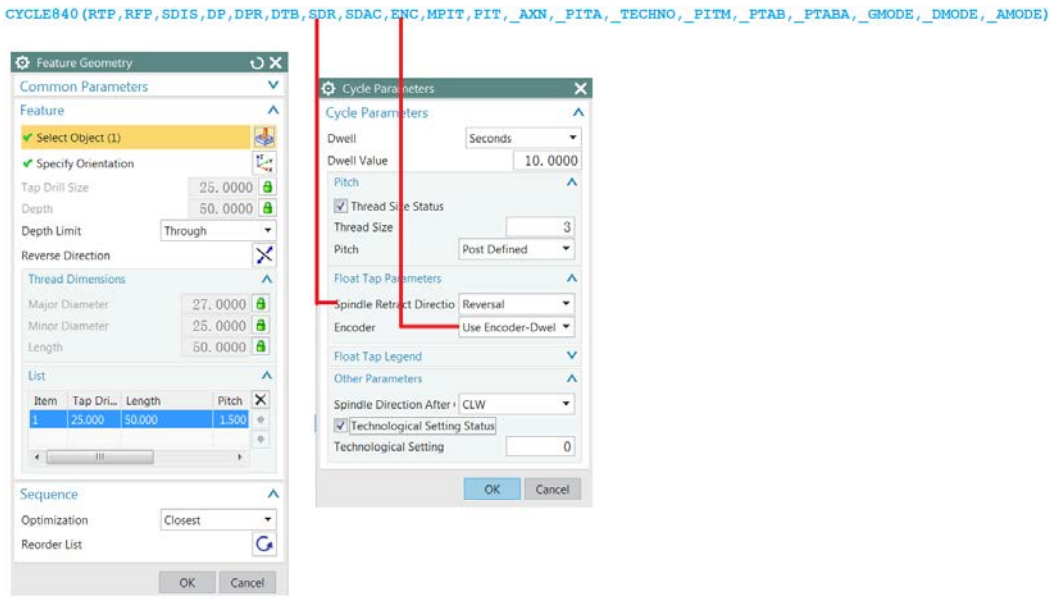


NOTE: *Except parameters mentioned above, other parameters information is showed in the table below.*

CYCLE84 Parameter	Comment
RTP	Retract plane
RFP	Reference plane
SDIS	Safety clearance
DP	Final drilling depth
DPR	Final drilling depth relative to the reference plane. It is always set to none in the OOTB.
_AXN	Tool axis. It is always set to none in the OOTB.
MPIT	If it is set, then PIT will not output.
_PITA(solutionline)	Its setting is the same as _PTAB.
PITM(solutionline)	It is set to 0 in the OOTB.
_PTABA(solutionline)	It is set to 0 in the OOTB.
_GMODE(solutionline)	It is set to none in the OOTB.
_DMODE(solutionline)	It is relative with spindle axis, the value may be 1, 2 or 3.
_AMODE(solutionline)	It is set to 0 in the OOTB.

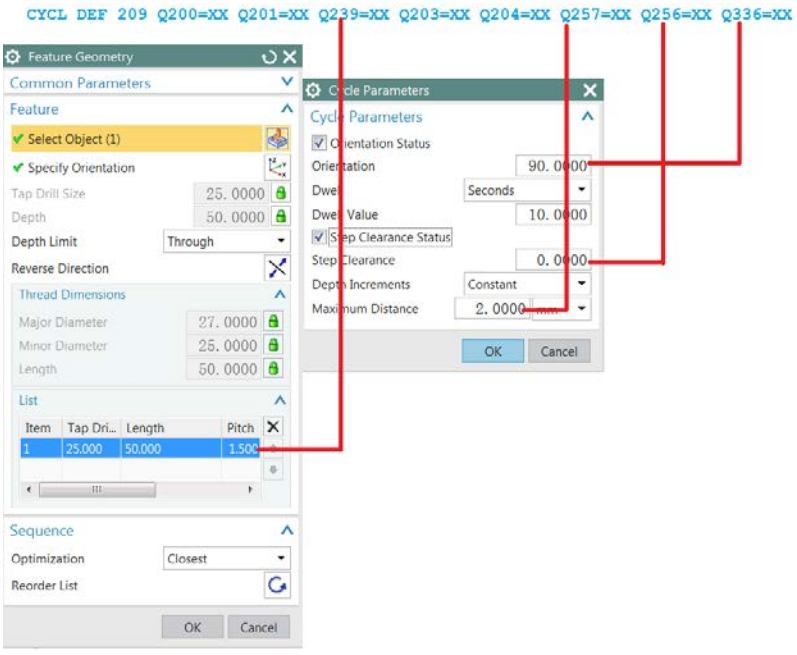
6.8.2 Set CYCLE840 parameter for S840D in UI

Except parameter SDR and ENC, other parameter settings are the same as CYCLE84. Please see the SDR and ENC setting in the picture below.



6.8.3 Set CYCLE209 parameter for ITNC in UI

Take tap breakchip UI as an example for ITNC. Please see the picture below.



NOTE: *Except parameters mentioned above, other parameters information is showed in the table below.*

CYCLE 209 Parameter	Comment
Q200	Distance between tool tip (at starting position) and workpiece surface.
Q201	Distance between workpiece surface and end of thread.
Q203	Coordinate of the workpiece surface.
Q204	Coordinate in the tool axis at which no collision between tool and workpiece (clamping devices) can occur.

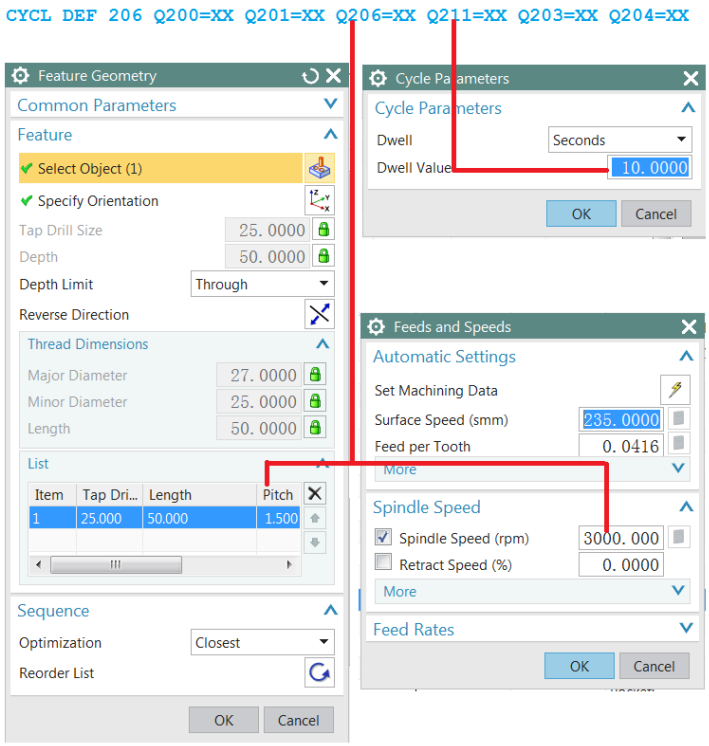
6.8.4 Set CYCLE207 parameter for ITNC in UI

CYCL DEF207 Q200=XX Q201=XX Q239=XX Q203=XX Q204=XX

CYCLE207 parameters are similar to CYCLE209, so the parameters setting are the same as CYCLE209.

6.8.5 Set CYCLE206 parameter for ITNC in UI

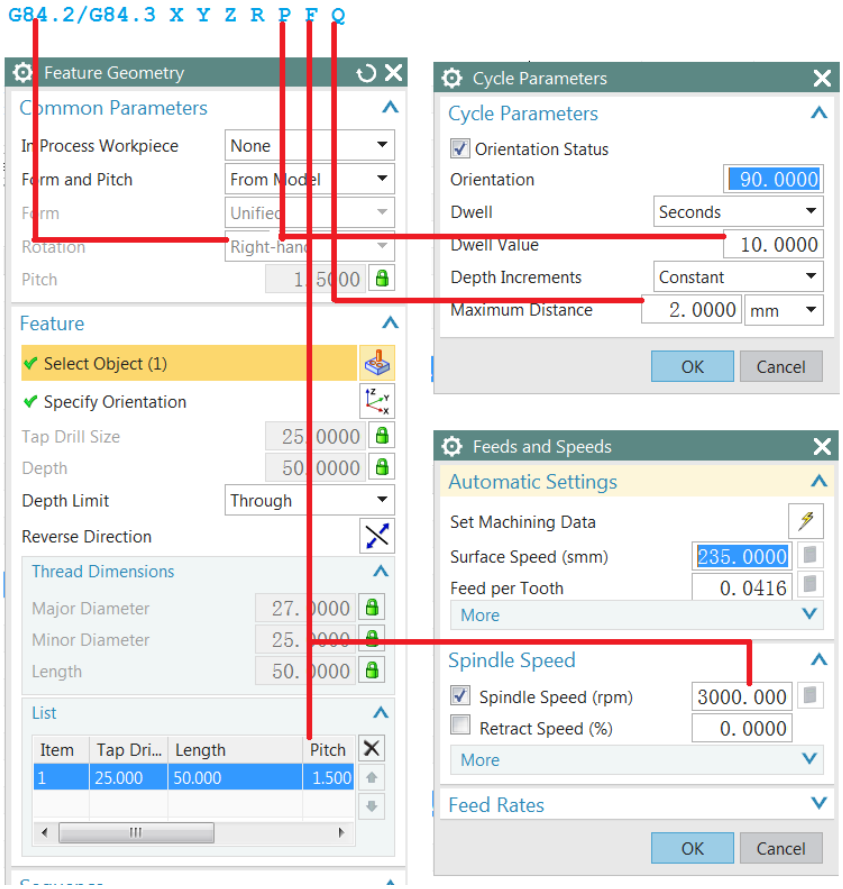
Except parameter Q206 and Q211 of CYCLE206, other parameters setting are the same as the CYCLE209.Please see the picture below.



NOTE: *Q206 is decided by pitch and spindle speed.*

6.8.6 Set G84.2/84.3 parameter for Fanuc in UI

Take tap breakchip UI as an example for Fanuc. Please see the picture below.



NOTE: *G84.2 and G84.3 are decided by "Rotation" value, if it is "Right-hand", then it outputs G84.2, or it outputs G84.3. And parameter F is decided by pitch and spindle speed. Except parameters mentioned above, other parameters information is showed in the table below.*

G84.2/G84.3 Parameter	Comment
X Y	Hole position data.
Z	The distance from point R to the bottom of the hole and the position of the bottom of the hole.
R	The distance from the initial level to point R level.

NOTE: *G84/G74 parameters are similar to G84.2/G84.3, so G84/74 parameters setting are the same as G84.2/84.3.*

7 About CAM Programming

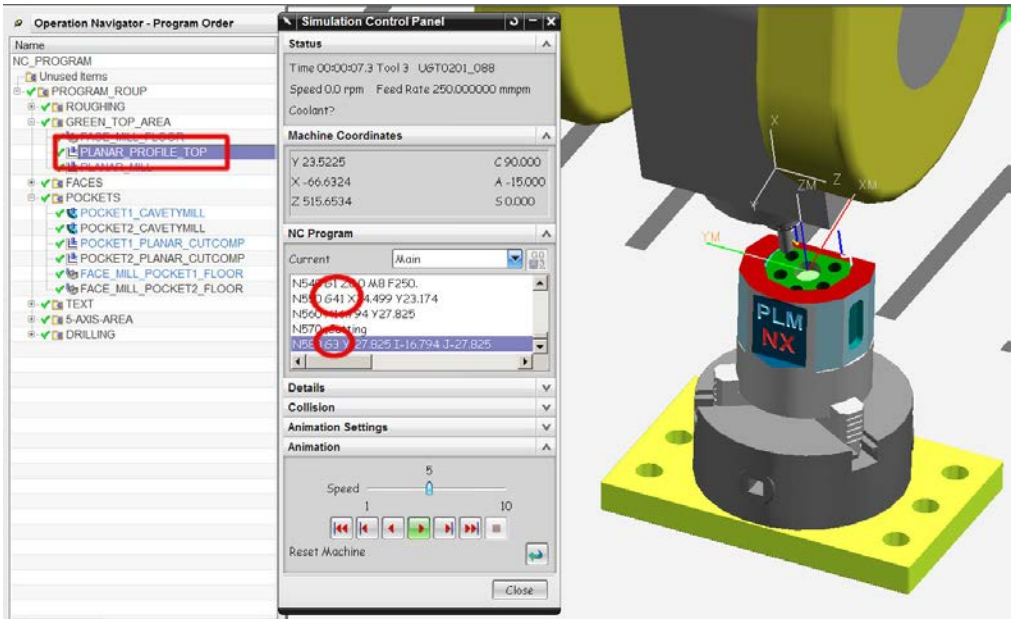
About Cutter Compensation for contact contour and circle statements in the NC code file hereby important information. Whenever the NC code contains statements which refer to a plane such as

G02/G03 or G41/G42, the controller needs to have the related 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.

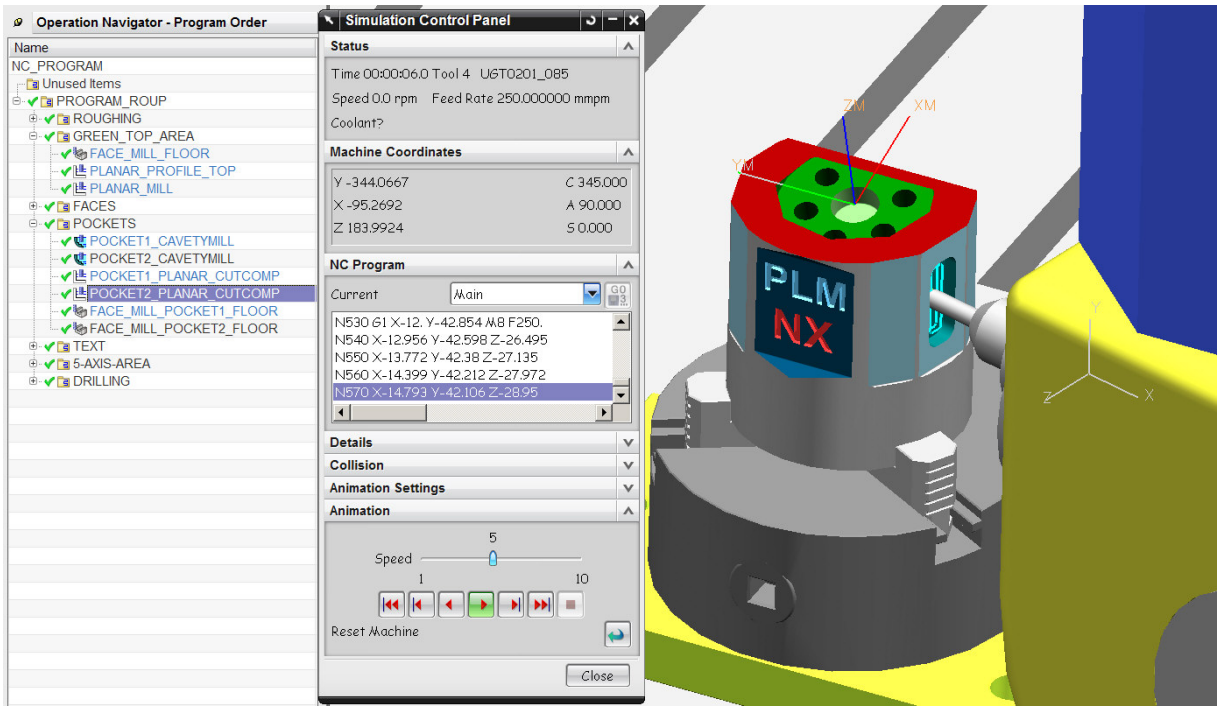
Important to understand are cases where the surfaces to be machined are not parallel to one of the orthogonal machine tool planes and the post processor has a specific transformation like TRAORI or using swivel cycles activated to achieve this operation. A typical example is shown with the external NC 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 swivel the plane in the TRAORI mode.

The latest used post in the example don’t output TRAORI anymore, but use CYCLE800 to swivel the plane.



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) as delivered with the actual NX version. It is not suggested to adding part files with the existing tool library entry names. (e.g. ugt010101_003.prt). It may happen that the added 3D tool does not match the geometry created based on the parameters.

9 About CSE Simulation Drivers

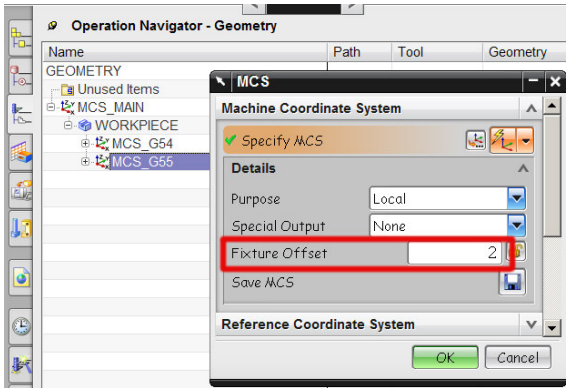
9.1 Handling of Offsets:

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. The system searched in the ONT for an MCS object with the related fixture offset number.

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.

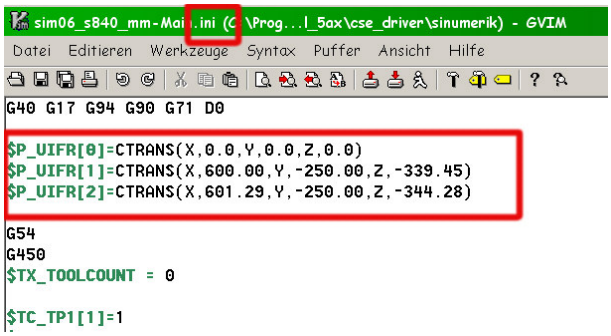


9.1.2 Alternative 2 (Sinumerik and TNC)

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 define 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
…
```

Note: *Working with CYCLE247 and/or is possible with CSE, but not part of the OOTB example.*

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 correct simulation results.

On physical machine tools, the reference point is stored inside of the controller. To mimic this in the 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.

9.4 Handling of tool correction data

Handling tool data CSE is able to get the data on different ways. One way is a direct connection to the application ISV and query the data through an API function during simulation e.g. when activating the tool correction e.g. Dxx or G43 Hxx).

An alternative way is to rely on data predefined in controller variables e.g. in an ini file. The behavior can be triggered in the OOTB CCF by setting the global variable GV_bUseSetToolCorrection in the CSEInitializeChannel. Most controller use the first described way and get the tool correction data via the API during simulation form ISV. **New with NX9** for the Sinumerik CCF (main purpose is for showcase) is, that these CCF relies completely on the provided data in the to_ini.ini file created by the OOTB post and doesn't query any tool correction data during simulation.

10 About Sinumerik Cycles in the content

The OOTB library for Sinumerik supports list of cycles for simulation. The cycles are covered by an encrypted archive file (*.cyc) and placed in the subprog folder of the Sinumerik CSE simulation. The cycles of Operate Version 4.7 SP2 HF1 are in the file: *SinumerikSL_Cycles_47_SP2_HF1.cyc*

- The cycles of the Solutionline Version 4.4 are in the file: *SinumerikSL_Cycles.cyc*
  shipped since Version NX85
- The cycles of the Solutionline Version 2.6 are in the file: *SinumerikSL_Cycles.cycV26*
  shipped until Version NX85
- The cycles of the Powerline are in the file *SinumerikPL_Cycles.cyc_powerline*
  shipped until Version NX85

To use the cycle other than form the default rename the files so that the desired one has the extension *.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/Operate

- CYCLE71
- CYCLE81 – CYCLE90
- CYCLE801
- HOLES1, HOLES2
- LONGHOLE
- SLOT1, SLOT2
- POCKET3, POCKET4
- CYCLE95
- CYCLE97
- CYCLE800

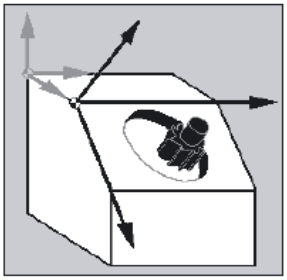
11 About swiveling cycles

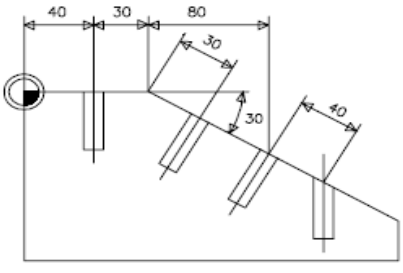
This will give an overview about what are swiveling cycles and how they are used. The main target is to machine on a plane not perpendicular to an existing linear axis. What it would take without swiveling:

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/AXIAL and

CYCLE19 on Heidenhain, CYCLE800 on S840D and G68.2 on Fanuc controllers work and how this is implemented in 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. With NX85 all the five axis machine tools in OOTB support swiveling for all three controller type.

11.1 Swiveling on Heidenhain

11.1.1 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. (PLANE SPATIAL is supported by OOTB post; PLANE AXIAL and CYCLE19 are supported in the CCF for simulation only)

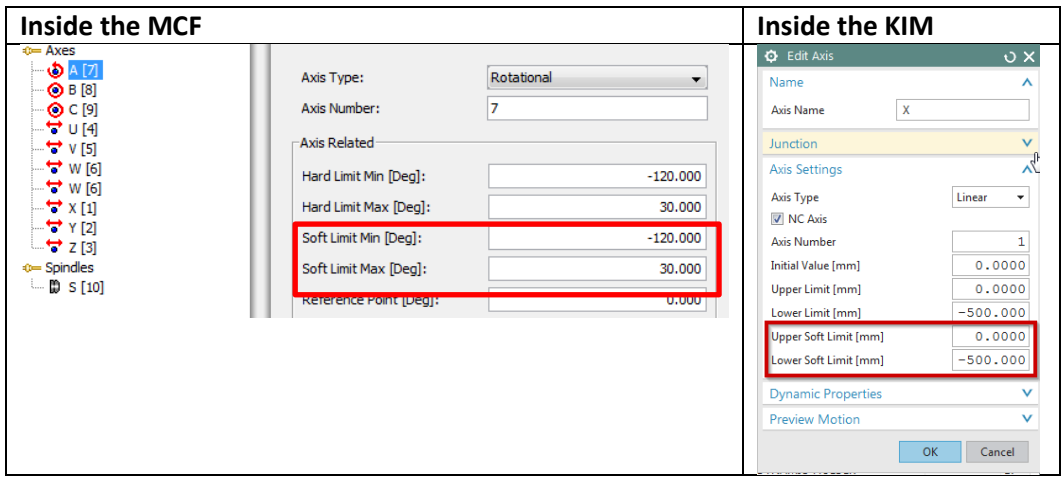
- milling operations and drilling operations needs to be 3+2 axis operation
- The MCS need to be set to “CSYS Rotation”

11.1.2 How to customize the machine data to achieve correct simulation

The settings are valid for all supported modes which are PLANE SPATIAL, PLANE AXIAL and CYCLE19. Inside the MCF global variable in Internal variables tab must be set related to the type of the machine tool and the axis vector directions. Name of the variables are:

- GV_strMachineType (“T”, “M”,”P”)
  „T“ for Tool → Head/Head
  “P” for Part → Table/Table
  “M” for Mixed → Head/Table
- GV_strSwivelingChainName
  The default name for the chain will be “default”
- GV_strFourthAxisName
- GV_strFifthAxisName
- GV_dFourthAxisX (x component of fourth axis)
- GV_dFourthAxisY (y component of fourth axis)
- GV_dFourthAxisZ (z component of fourth axis)
- GV_dFifthAxisX (x component of fifth axis)
- GV_dFifthAxisY (y component of fifth axis)
- GV_dFifthAxisZ (z component of fifth axis)

And the soft min/max limit of the rotary axes should be set if you wish rotary axis solution can be selected based on axes limit.

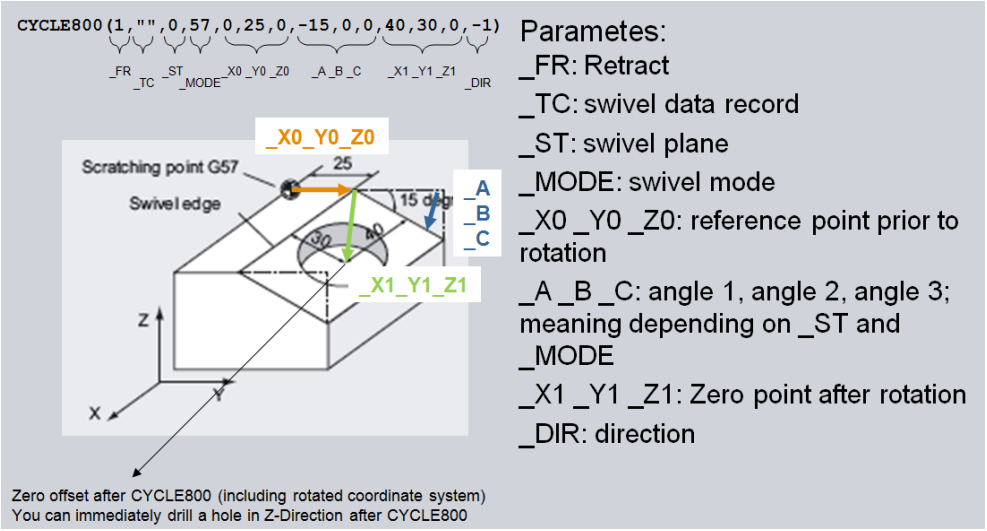


More details about the cycle and the way how it is implemented in CSE can be found in the Appendix

11.2 Swiveling in Sinumerik - CYCLE800

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

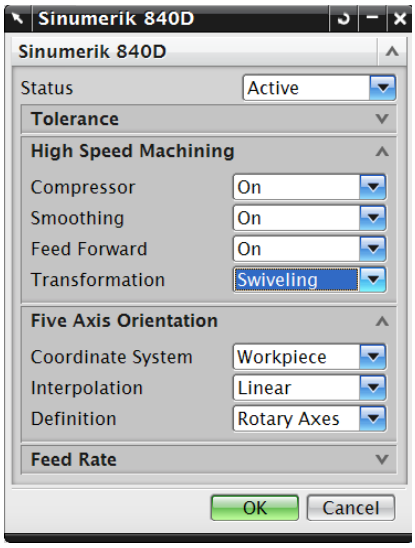
11.2.1 Example and parameter



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

To achieve an output of CYCLE800 by the Post the following prerequisites needs to be fulfilled.

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



11.2.3 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 for NCK4.4. cycles

```
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
```

- Content of the OOTB INI file for NCK4.7. cycles

```
PMAC; adopt the original DEF file from MC
PGUD; need remove the line with REDEF
TC_CARR; define tool holder data
CHAN_DATA
G40 D0
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*        ;software minimum limit of rotary axes v1
$TC_CARR31*        ;software minimum limit of rotary axes v2
$TC_CARR32*        ;software maximum limit of rotary axes v1
$TC_CARR33*        ;software minimum limit of rotary axes v2
$TC_CARR34         ;tool holder name
$TC_CARR35*        ;Axis name 1
$TC_CARR36*        ;Axis name 2
$TC_CARR37         ;Identifier
$TC_CARR40         ;Z axis retract value
```

```
*With global variable “GV_bSetToolCarrierDataFromKim” the user can decide if the local
toolcarrier dataset should be used or the information should be retrieved from the current active
kinematic chain and axis information from kinematic model. Default value in CCF file is TRUE.
```

11.3 Swiveling in Fanuc - G68.2

11.3.1 How the ONT need to be defined to achieve a proper Post output

To achieve an output of G68.2 by the Post the following prerequisites needs to be fulfilled.

- Milling operations and drilling operations needs to be 3+2 axis operation
- The MCS need to be set to “CSYS Rotation”

11.3.2 How to customize the machine data to achieve correct simulation

The requirement for G68.2 is the same to the requirement for PLANE SPATIAL. Please refer to previous chapter. More details about the cycle and the ay how it is implemented in CSE can be found in the Appendix

12 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 library directory – parallel to the place where the MCF file is located
- All INI files in the CAM-Setup-part working directory (subfolder cse_files)
- 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)
- 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. Swiveling Cycles

All three CCFs for TNC, Sinumerik and Fanuc include one swivel cycle implementation for the related NC code PLANE SPATIAL, TCARR/PAROT and G68.2. The way how that is implemented is consolidate over the different controllers and follow one general workflow. Here the details of that workflow:

- Initialize the system/global variables to define chain kinematics.
- Apply work piece coordinate rotation (and translation) to transformation (WPFRAME for Sinumerik, ROTATIONAL for Fanuc, PLANE for TNC).
- Calculate the tool vector after work piece coordinate rotation.
- Calculate tool angle and part rotating angle with rotated tool vector.
- Select the tool and part rotating angle based on the restriction (SEQ, limitation).
- Compensate the linear offset caused by tool and part rotating.
- Compensate the rotating offset caused by table rotating if applicable (for HeadTable and TableTable machine).
- Move and rotate head and/or table.

G. PLANE SPATIAL

Inside the used MCF global variable are set related to the type of the machine tool and the axis vector directions. Name of the variables are:

- GV_strMachineType (“T”, “M”,”P”)
- GV_strFourthAxisName (fourth axis name)
- GV_strFifthAxisName
  (fifth axis name – is dependent on forth axis for T and P; for M it is the one near to the part)
- GV_dFourthAxisX (x component of forth axis)
- GV_dFourthAxisY (y component of forth axis)
- GV_dFourthAxisZ (z component of forth axis)
- GV_dFifthAxisX (x component of fifth axis)
- GV_dFifthAxisY (y component of fifth axis)
- GV_dFifthAxisZ (z component of fifth axis)
- GV_dFourthAxisLimitMin (fourth axis limit min)
- GV_dFourthAxisLimitMax (fourth axis limit max)
- GV_dFifthAxisLimitMin (fifth axis limit min)
- GV_dFifthAxisLimitMax (fifth axis limit max)

Here an example how that will look like in the case of the complex DMU EVO machine tool

- GV_strMachineType: "P"
- GV_strFourthAxisName: "B"
- GV_strFifthAxisName: "C"
- GV_dFourthAxisX 0.584825
- GV_dFourthAxisY -0.573576
- GV_dFourthAxisZ 0.573576
- GV_dFifthAxisX 0.0
- GV_dFifthAxisY 0.0
- GV_dFifthAxisZ -1.0
- GV_dFourthAxisLimitMin -9999
- GV_dFourthAxisLimitMax 9999
- GV_dFifthAxisLimitMin -9999
- GV_dFifthAxisLimitMax 9999

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. This takes effect only if the configuration is TABLE ROT, machine type head-head and SPC>0 (Q122=0)

Workflow of internal PLANE SPATIAL implementation

- Apply SPA SPB SPC rotation to PLANE
- Calculate the tool vector with SPA SPB SPC rotation
- Generate possible tool and part angle solution using calculateIKSAngles
- Get valid tool and part angle solution via GMe_SwivelingGetRotateSolution
- Set Q120 Q121 Q122 accordingly
- Set linear compensation due to tool holder rotation via GMe_SwivelingCalculateLinears
- Align WCS on table rotation via GMe_SwivelingCompTableRotation
- Handle TABLE ROT exception
- Move rotary axis with meta code MOVE/TURN/STAY and ABST.

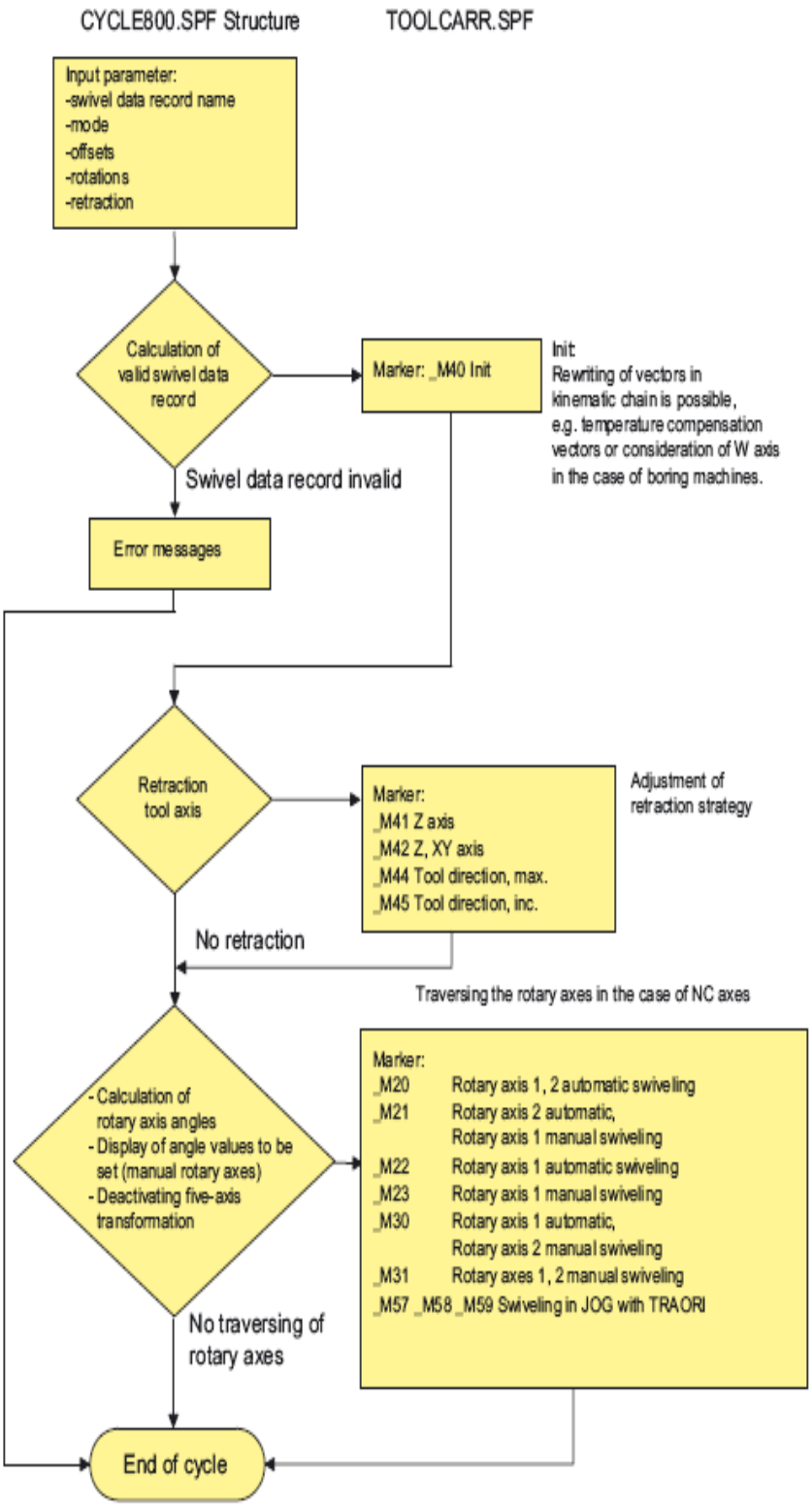
H. CYCLE800

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.

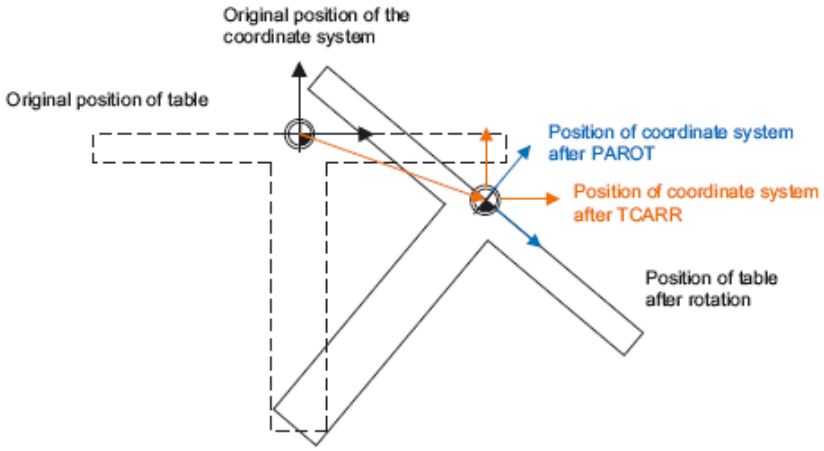

CYCLE800.SPF Structure
TOOLCARR.SPF
Input parameter:
-swivel data record name
-mode
-offsets
-rotations
-retraction
Calculation of valid swivel data record
Marker: _M40 Init
Init:
Rewriting of vectors in kinematic chain is possible, e.g. temperature compensation vectors or consideration of W axis in the case of boring machines.
Swivel data record invalid
Error messages
Retraction tool axis
Marker:
_M41 Z axis
_M42 Z, XY axis
_M44 Tool direction, max.
_M45 Tool direction, inc.
Adjustment of retraction strategy
No retraction
Traversing the rotary axes in the case of NC axes
- Calculation of rotary axis angles
- Display of angle values to be set (manual rotary axes)
- Deactivating five-axis transformation
Marker:
_M20 Rotary axis 1, 2 automatic swiveling
_M21 Rotary axis 2 automatic, Rotary axis 1 manual swiveling
_M22 Rotary axis 1 automatic swiveling
_M23 Rotary axis 1 manual swiveling
_M30 Rotary axis 1 automatic, Rotary axis 2 manual swiveling
_M31 Rotary axes 1, 2 manual swiveling
_M57 _M58 _M59 Swiveling in JOG with TRAORI
No traversing of rotary axes
End of cycle

Basic about TCARR and PAROT

The command TCARR calculates rotational axis angle, and calculates the translation compensation based on current kinematics, and apply it to the according transformation. The command PAROT rotates the transformation “PARTFRAME” for the table.

Depending on the mode of $P_GG[42] (TCOABS, TCOFR) the compensation is calculated in TCARR.

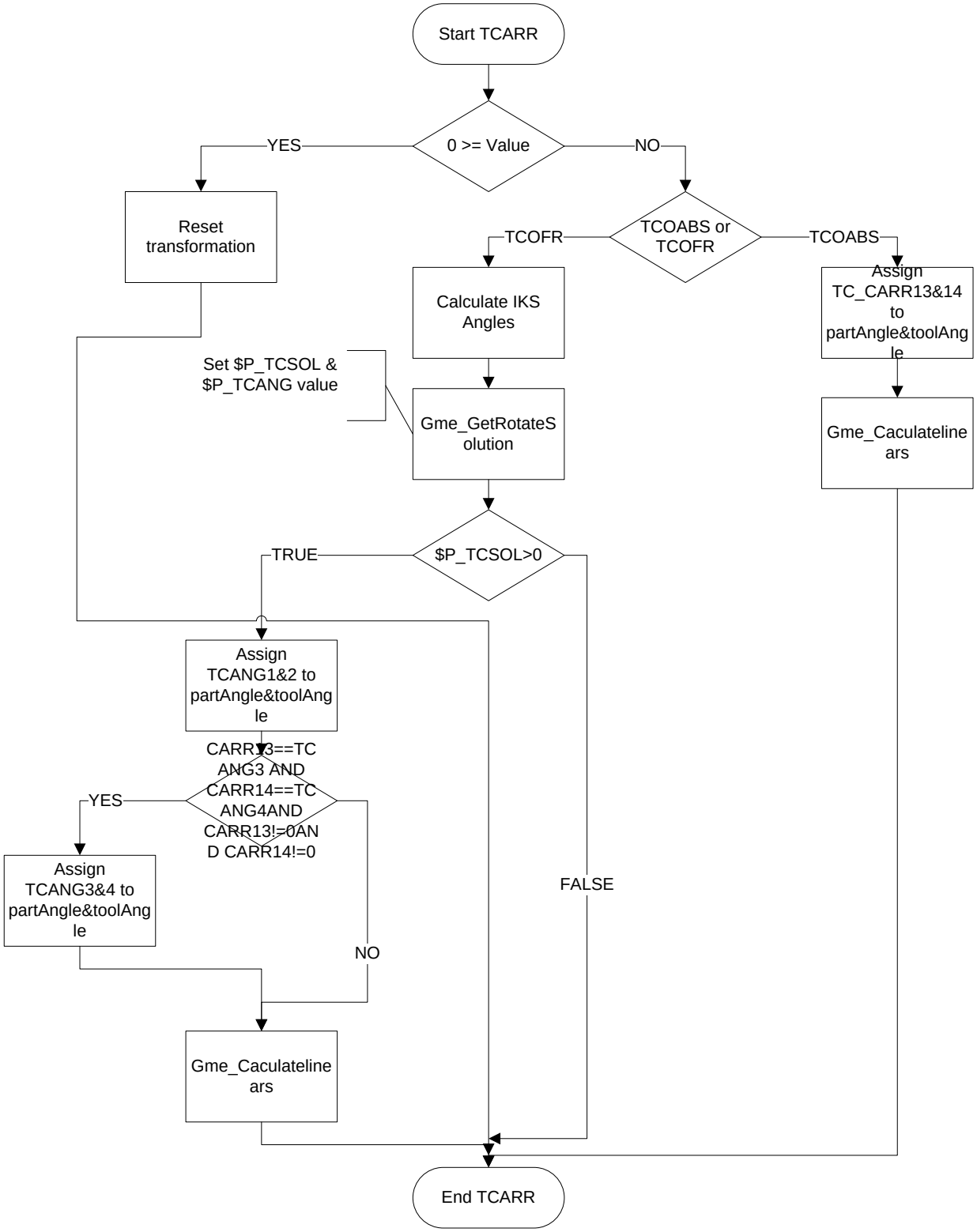
- With TCOABS
  Based on the angles defined in TC_CARR13 and TC_CARR14
- With TCOFR
  Based on the current tool orientation the angles are calculated first → IKS and picker required
  simCYCLE800 consists of two subprograms: CYCLE800.SPF(Siemens),



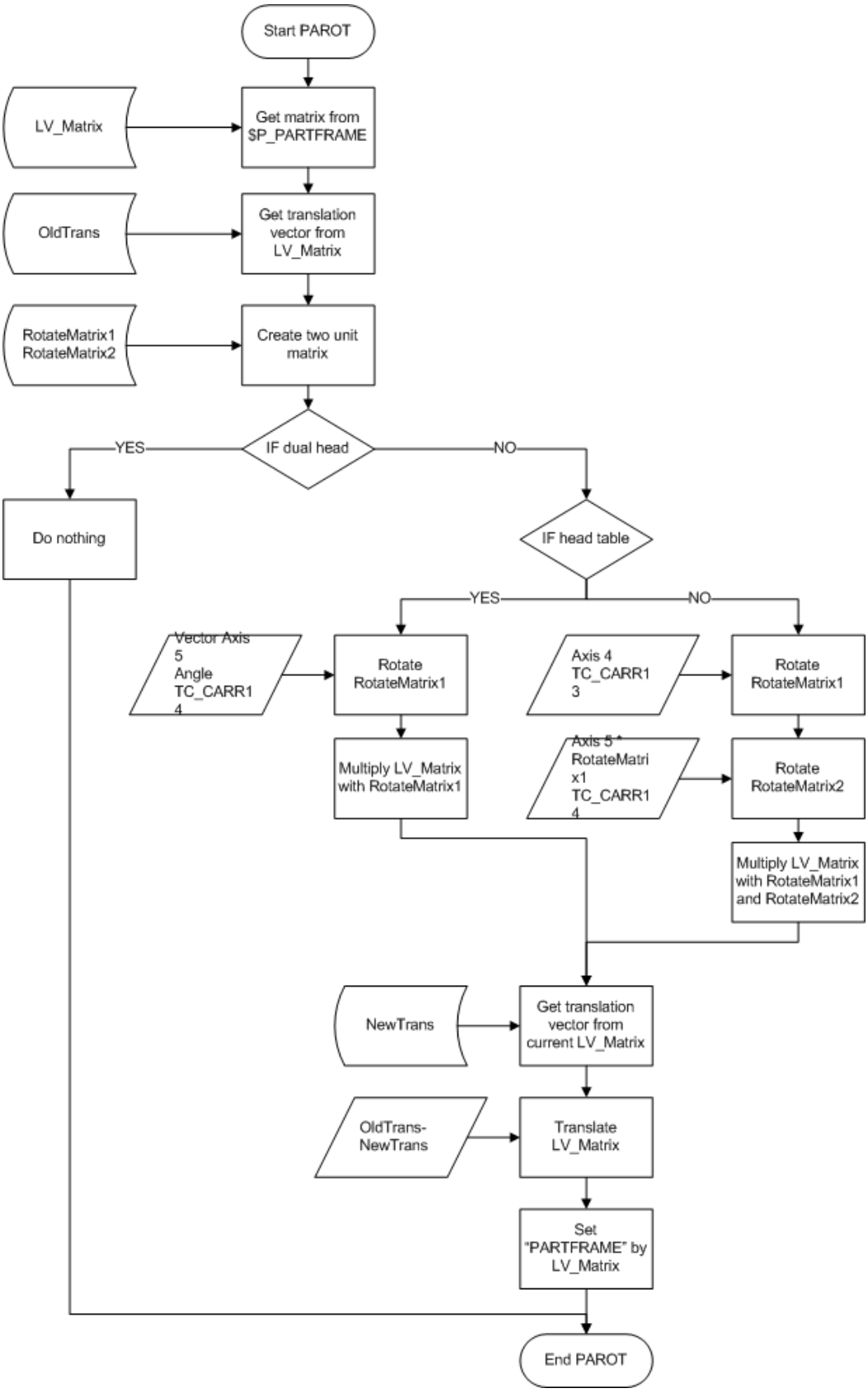
TOOLCARR.SPF(CUST_800 is the name used in current CYCLE800.SPF) It moves and rotates the tool or table according to the input mode. Currently it is implemented for M20 M40 M41 cases.

- 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



Working with OOTB MACH Simulation Examples

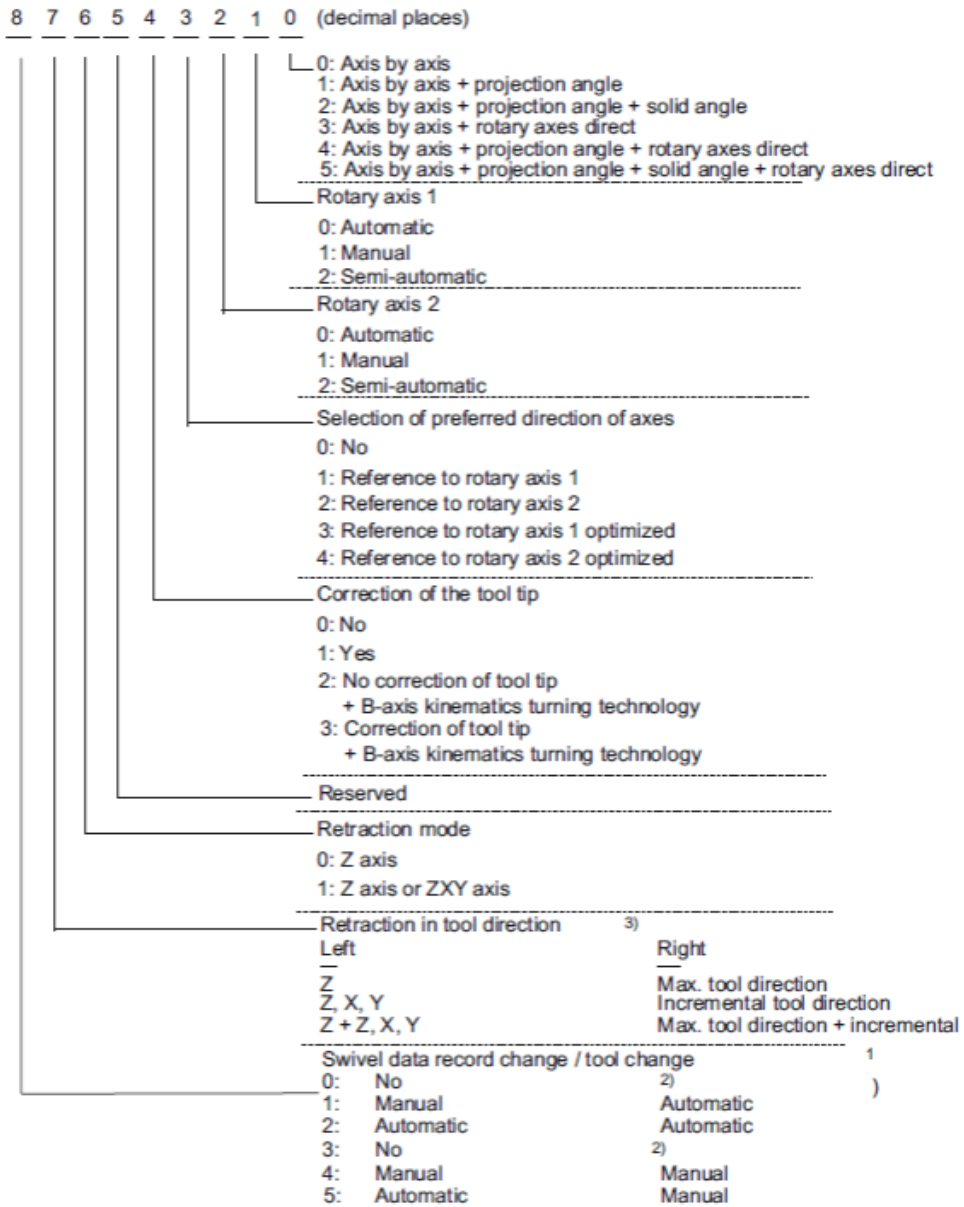
How to set $TC_CARR37

The OOTB examples uses the Solutionline version 4.4 example: $TC_CARR37[1]=201003003

Display variants of input screen forms CYCLE800 $TC_CARR37[n]

(n ⇒ swivel data record)

If the following display variants are not set, the value will not be displayed in the input screen form (see Section "Programming via Screen Form").



1) Only relevant for ShopMill/ShopTurn
2) If no swivel data record change is declared, the setting automatic/manual tool change setting is not relevant
3) Coding of retraction modes, see following table

I. G68.2

Workflow of internal G68.2 implementation

- Apply XYZ translation and IJK rotation to ROTATIONAL
- Calculate the tool vector with IJK rotation
- Generate possible tool and part angle solution using calculateIKSAngles
- Select the tool and part rotating angle based on the axis limitation by GMe_SwivelingGetRotateSolution.
- Set GV_dFourthAxisAngle and GV_dFifthAxisAngle accordingly
- Set linear compensation due to tool holder rotation using GMe_SwivelingCalculateLinears
- Align WCS on table rotation via GMe_SwivelingCompTableRotation

About G53.1

If G68.2 specifies the relationship between the feature coordinate system and the work piece coordinate system, G53.1 will automatically specifies the +Z direction of the feature coordinate system as the tool axis direction even if no angle is specified for the rotary axis.

G53.1 must be specified in a block after the block that contains G68.2

G53.1 must be specified in a block in which there is no other command.

J. Default controller settings inside the CCFs

This section list the settings, which are take place when a CSE simulation is initialized based on the OOTB CCF. Thus the method CSEInitializeChannel is called, which sets these defaults. Afterwards possible ini files get registered, which add addition defaults or can overwrite the ones already set.

Sinumerik:

Controller default set in the CCF:

- G0 G17 G54 G60 G601 G710 G90 G94 CFC NORM G450 BRISK ORIMKS DIAMOF TCOABS
- GV_bUseLoadOffset=FALSE to deactivate the LoadOffset functionality.
- GV_bUseSetToolCorrection=FALSE to not use SetToolCorrection in D-Metacode and use data from ini file instead.
- $MC_FRAME_ADD_COMPONENTS=1. This is used by G58 and G59. Also means that TRANS stores in $P_PFRAME[.., TR] and ATRANS stores in $P_PFRAME[.., FI]
- $MN_SCALING_SYSTEM_IS_METRIC=25.4
- $MC_DIAMETER_AX_DEF=”X”
- $MN_INT_INCR_PER_MM=1000
- $AN_NCK_VERSION=75000
- $P_PROG[0]=”_N_MAIN_MPF”
- $SCS_DRILL_TAPPING_SET_GG12[0]=1
- $SCS_DRILL_TAPPING_SET_GG12]1]=1
- $SCS_DRILL_TAPPING_SET_GG21[0]=1
- $SCS_DRILL_TAPPING_SET_GG12[1]=1
- $SCS_DRILL_TAPPING_SET_GG24[0]=1
- $SCS_DRILL_TAPPING_SET_GG24[1]=1
- $SCS_DRILL_TAPPING_SET_MC[0]=1
- $SCS_DRILL_TAPPING_SET_MC[1]=1
- $MC_MM_SYSTEM_FRAME_MASK=62

Internal used variables starting with SIM_...

Controller defaults from the ini File:

- TO_INI ; load a to_ini file including tool and offset data produced by the OOTB post
  (the file TO_INI is handled like a subprogram. If this is not found the system will NOT raise an error message as usual)
- PMAC ; definition subprogram from Siemens SL
- PGUD ;definition subprogram from Siemens SL
- CHAN_DATA; Controller Variable Initialization
- TC_CARR ; Sets data for the CYCLE800 see TC_CARR.DEF

Fanuc:

Defaults set in the CCF:

- G21 G0 G17 G90 G94
- Default-Configuration for the controller:#983=1 : Serie M - machining centers.
- (Change in INI program to 0 for Serie T
- #3401=1: calculator notation, G-Code System
- #983=1
- ABit 0: 1 = calculator notation; 0 = smallest increment notationBit 6+7: 0 = G-Code System A; 64 = G-Code
- System B; 128 = G-Code System CThe smallest increment multiplier can be specified in GMe_SetUnit (Default:
- 1000.0).#19696=0 : rotating table coordinate system for G43.4/5. Change in INI program to 32 for non-rotating
- workpiece coordinate system
- GV_bUseLoadOffset=TRUE to activate the LoadOffset functionality

Defaults from the ini File:

- (#19696=0 : rotating table coordinate system for G43.4/5. Change in INI program to 32 for non-rotating workpiece coordinate system)
- #19696=32
- G54
- (Implement the setting of the reference point G28 unit dependent)
- G10L52
- N1240P1R0
- N1240P2R0
- N1240P3R-200.000
- G11

TNC Heidenhain:

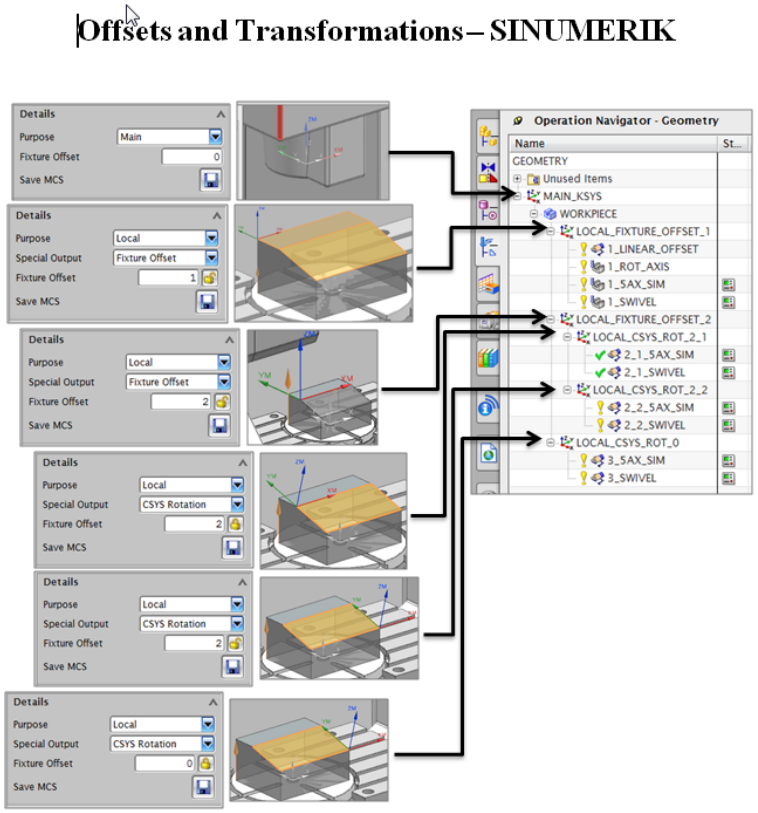
Defaults set in the CCF:

- GV_bUseLoadOffset=TRUE (used for CYCLE_DEF_7)
- GV_bUseLoadOffset247=FALSE (used for CYCLE_DEF_247)
- Systemdata: 7680[6]=0
- Systemdata: 7682[4]=1 ;1 when GV_strFourthAxisName is available, 0 if not
- Systemdata: 7682[5]=1 ;1 when GV_strFifthAxisName is available, 0 if not
-

Defaults from the ini File:

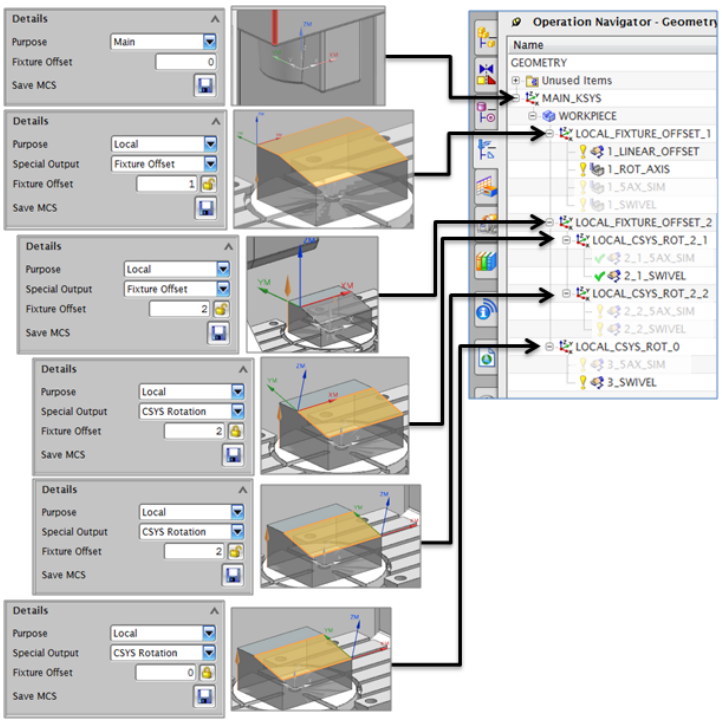
- No ini File

K. OOTB MCS Handling output details



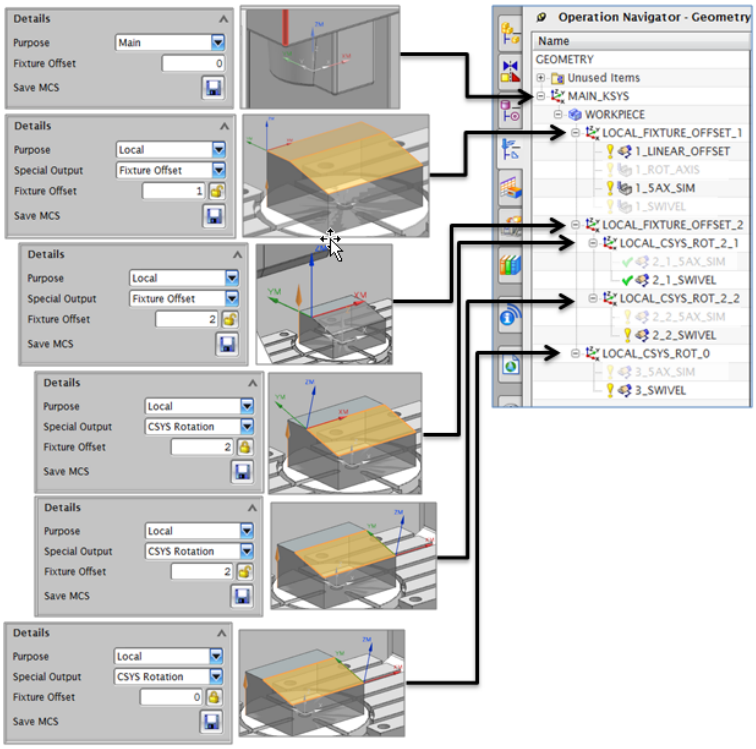

```
1_LINEAR_OFFSET: 3 Axis Mill Operation
    TRAFOOF
    G54
1_ROT_AXIS: 3+2 Axis Mill Operation
    G0 A15. C0.0
    TRAFOOF
    G54
1_5AX_SIM: 5 Axis Mill Operation
(840D UDE: TRAORI, ROTARY AXES)
    G0 A15. C0.0
    TRAORI
    G54
    TRANS X0.0 Y0.0 Z0.0
    AROT X15.
1_SWIVEL: 3+2 Operation
(840D UDE: SWIVEL, ROTARY AXES)
    G54
    CYCLE800(1,"R_DATA",0,57,0.,0.,0.,15.,0.,0.,0.,0.,0.,1,1.)
    2_1_5AX_SIM: 5 Axis Mill Operation
    (840D UDE: TRAORI, ROTARY AXES)
        G0 A15. C0.0
        TRAORI
        G55
        TRANS TRANS X0.0 Y0.0 Z0.0
        AROT X15.
    2_1_SWIVEL: 3+2 Operation
    (840D UDE: SWIVEL, ROTARY AXES)
        G55
        CYCLE800(1,"R_DATA",0,57,0.,-0.,0.,15.,-0.,-0.,0.,0.,0.,1,1.)
    2_2_5AX_SIM: 5 Axis Mill Operation
    (840D UDE: TRAORI, ROTARY AXES)
        G0 A15. C0.0
        TRAORI
        G55
        TRANS X200. Y-100. Z-26.79
        AROT X15.
    2_2_SWIVEL: 3+2 Operation :
    (840D UDE: SWIVEL, ROTARY AXES)
        G55
        CYCLE800(1,"R_DATA",0,57,200.,-100.,-26.79,15.,-0.,-0.,0.,0.,0.,1,1.)
3_5AX_SIM : 5 Axis Mill Operation :
(840D UDE: TRAORI, ROTARY AXES)
    G0 A15. C0.0
    TRAORI
    G500
    TRANS X100. Y-305. Z-456.79
    AROT X15.
3_SWIVEL :3+2 Operation :
(840D UDE: SWIVEL, ROTARY AXES)
    G500 ;(similar to G53)
    CYCLE800(1,"R_DATA",0,57,100.,-305.,-456.79,15.,-0.,-0.,0.,0.,0.,1,1.)
```

Offsets and Transformations– Fanuc




```
Note:
With that operation examples and the default settings in NX CAM it´s
not possible to obtain similar 5 axis simultaneous operation like in
Sinumerik.

Further information you´ll find in the BASIC VERIFICATION TESTS for
FANUC controller

1_LINEAR_OFFSET: 3 Axis Mill Operation
    G54
1_ROT_AXIS: 3+2 Axis Mill Operation
    G54
    G0 A15. C0.0
1_5AX_SIM: 5 Axis Mill Operation

1_SWIVEL: 3+2 Operation

    2_1_5AX_SIM: 5 Axis Mill Operation

    2_1_SWIVEL: 3+2 Operation
        G55
        A-15 C90
        G68.2 X0 Y0 Z0 I1 J0 K0 R15
    2_2_5AX_SIM: 5 Axis Mill Operation

    2_2_SWIVEL: 3+2 Operation:

3_5AX_SIM : 5 Axis Mill Operation

3_SWIVEL :3+2 Operation
    G00 A-15. C90.
    G68.2 X0.0 Y0.0 Z0.0 I1 J0 K0 R15.
```

Offsets and Transformations– Heidenhain




```
Note:
With that operation examples and the default settings in NX CAM it´s
not possible to obtain similar 5 axis simultaneous operation like in
Sinumerik.

Further information you´ll find in the BASIC VERIFICATION TESTS for
Heidenhain controller

1_LINEAR_OFFSET: 3 Axis Mill Operation
    CYCL DEF 7.0
    CYCL DEF 7.1 X-100
    CYCL DEF 7.2 Y-145
    CYCL DEF 7.3 Z-430
1_ROT_AXIS: 3+2 Axis Mill Operation

1_5AX_SIM: 5 Axis Mill Operation
    L A15. C0.0
    M128
1_SWIVEL: 3+2 Operation
    2_1_5AX_SIM: 5 Axis Mill Operation

    2_1_SWIVEL: 3+2 Operation
        CYCL DEF 7.0
        CYCL DEF 7.1 X-100
        CYCL DEF 7.2 Y-205
        CYCL DEF 7.3 Z-430
        PLANE SPATIAL SPA15. SPB0.0 SPC0.0 TURN F+500 TABLE ROT
    2_2_5AX_SIM: 5 Axis Mill Operation

    2_2_SWIVEL: 3+2 Operation
        CYCL DEF 7.0
        CYCL DEF 7.1 IX 200
        CYCL DEF 7.2 IY -100
        CYCL DEF 7.3 IZ -26.7949
        PLANE SPATIAL SPA15. SPB0.0 SPC0.0 TURN F+500 TABLE ROT
3_5AX_SIM: 5 Axis Mill Operation

3_SWIVEL: 3+2 Operation :
    CYCL DEF 7.0
    CYCL DEF 7.1 X 0
    CYCL DEF 7.2 Y 0
    CYCL DEF 7.3 Z 0

    CYCL DEF 7.1 IX 100
    CYCL DEF 7.2 IY -305
    CYCL DEF 7.3 IZ -99

    PLANE SPATIAL SPA15. SPB0.0 SPC0.0 TURN F+500 TABLE ROT
```

Global Technical Access Center

Installation assistance

For additional installation assistance, or to report any problems, contact the Global Technical Access Center (GTAC).

Website:
http://support.industrysoftware.automation.siemens.com/gtac.shtml

Phone:
United States and Canada: 800-955-0000 or 714-952-5444

Outside the United States and Canada: Contact your local support office.