



NX Post Configurator Tasks and Solutions

Unrestricted © Siemens AG 2020

Task 1



Create an additional UDE with Post Configurator for a horizontal milling machine tool to setup W and V axis for Sinumerik 840D:

Conditions:

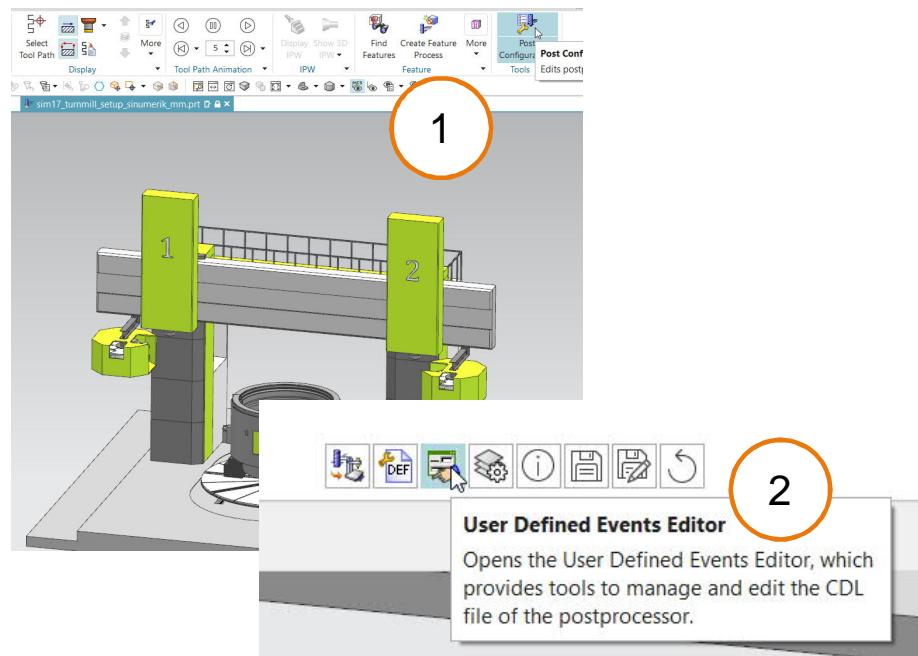
- Possibility to switch between ATRANS and specific OEM Cycle (e.g. L9958, Waldrich Coburg)
- The switch must be done by a property
- The output appears on the right location in NC code
- Contains logic that the values are entered correctly in the UDE

Sample Code Sinumerik standard
ATRANS Z=200 W=-150 V=-50
G0 V0 W0 Z0

Sample Code L9958
ZV=200 WV=-150 VV=-50
L9958()

Task 1

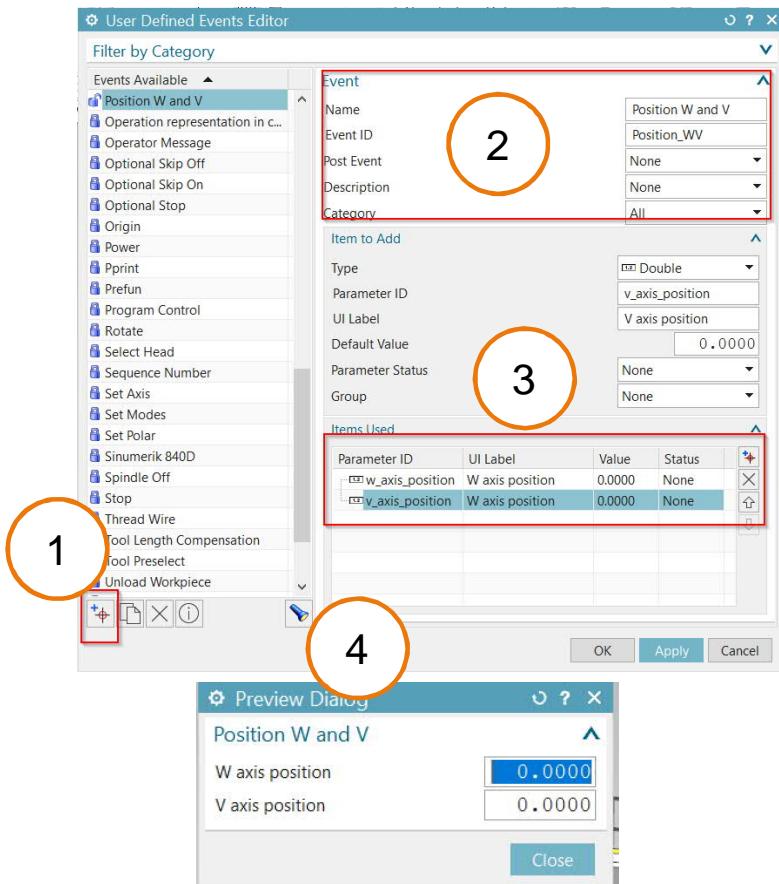
Solution 1/15



1. Open sim17 OOTB sample and Post Configurator application
2. Start the UDE editor to define a new UDE for W and V axis values

Task 1

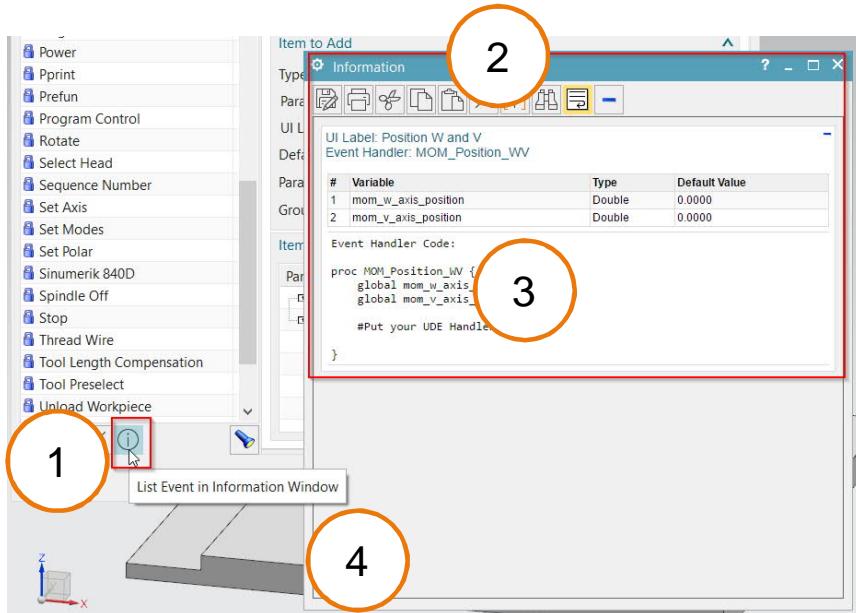
Solution 2/15 – Create UDE for W and V axis



1. Add a new event for positioning W and V axis
2. Add UI name and MOM event name to execute later Tcl code
3. Add two new items as double to set up the values for W and V axis
4. Check the UDE with the Preview Button and that this contains the created items

Task 1

Solution 3/15 – Get created Tcl information



1. Add a new event for positioning W and V axis
2. Add UI name and MOM event name to execute later Tcl code
3. Add two new items as double to set up the values for W and V axis
4. Check the UDE with the Preview Button and that this contains the created items

Task 1

Solution 4/15 – Add UDE event to service file

```

Editor
868 proc ootb_convert_home_pos {} {
869 # This proc is used to convert mom sys home_pos to inch and metric units.
870 if {[info exists ::mom_sys_home_pos(0)]} {
871     set ::mom_sys_home_pos_metric(0) $::mom_sys_home_pos(0)
872     set ::mom_sys_home_pos_metric(1) $::mom_sys_home_pos(1)
873     set ::mom_sys_home_pos_metric(2) $::mom_sys_home_pos(2)
874
875     set ::mom_sys_home_pos_inch(0) [expr $::mom_sys_home_pos_metric(0)/25.4]
876     set ::mom_sys_home_pos_inch(1) [expr $::mom_sys_home_pos_metric(1)/25.4]
877     set ::mom_sys_home_pos_inch(2) [expr $::mom_sys_home_pos_metric(2)/25.4]
878
879     unset ::mom_sys_home_pos(0)
880     unset ::mom_sys_home_pos(1)
881     unset ::mom_sys_home_pos(2)
882 }
883
884
885 #This command must be called at very end of ootb service layer.
886 ootb_convert_home_pos
887
888 proc MOM_Position_WV {} {
889     global mom_w_axis_position
890     global mom_v_axis_position
891
892     #Put your UDE Handler Tcl here
893 }
894
895

```

1. Close UDE editor and keep information listing open and open the service file/layer
2. Copy the eventhandler information from listing window
3. Add eventhandler to service file

Task 1

Solution 5/15 – Create new group and property

```
LIB_GE_CREATE_obj Custom_tree {UI_TREE} {
    LIB_GE_property_ui_name      "UIName"
    LIB_GE_property_ui_tooltip   "UITooltip"

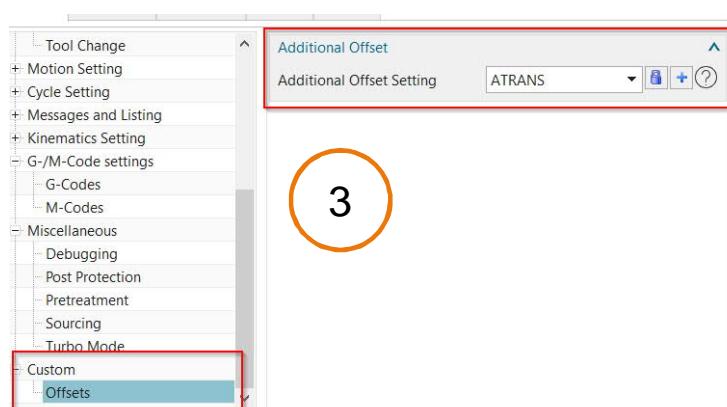
    set id "cst_tree_root"
    set $id "0"
    set datatype($id)      "NODE"
    set access($id)        222
    set dialog($id)         {{Custom}}
    set descr($id)          {{Custom Elements}}
    set group_status($id)   0
    set ui_parent($id)      "root"
    set ui_sequence($id)    -1

    set id "cst_tree_node_offset"
    set $id "0"
    set datatype($id)      "NODE"
    set access($id)        222
    set dialog($id)         {{Offsets}}
    set descr($id)          {{Node description}}
    set group_status($id)   0
    set ui_parent($id)      "cst_tree_root"
    set ui_sequence($id)    -1

    set id "cst_tree_group_additional_offset"
    set $id "0"
    set datatype($id)      "GROUP"
    set access($id)        222
    set dialog($id)         {{Additional Offset}}
    set descr($id)          {{Group description}}
    set group_status($id)   1
    set ui_parent($id)      "cst_tree_node_offset"
    set ui_sequence($id)    -1
}
```

```
LIB_GE_CREATE_obj CST_CTRL {} {
    LIB_GE_property_ui_name      "Name"
    LIB_GE_property_ui_tooltip   "Tooltip"

    set id "additional_offset"
    set $id 0
    set options($id)      {ATRANS|WaCo L9958}
    set options_ids($id)  {0|1}
    set access($id)        222
    set dialog($id)         {{Additional Offset Setting}}
    set descr($id)          {{A False or True Property}}
    set ui_parent($id)      "cst_tree_group_additional_offset"
    set ui_sequence($id)    -1
}
```

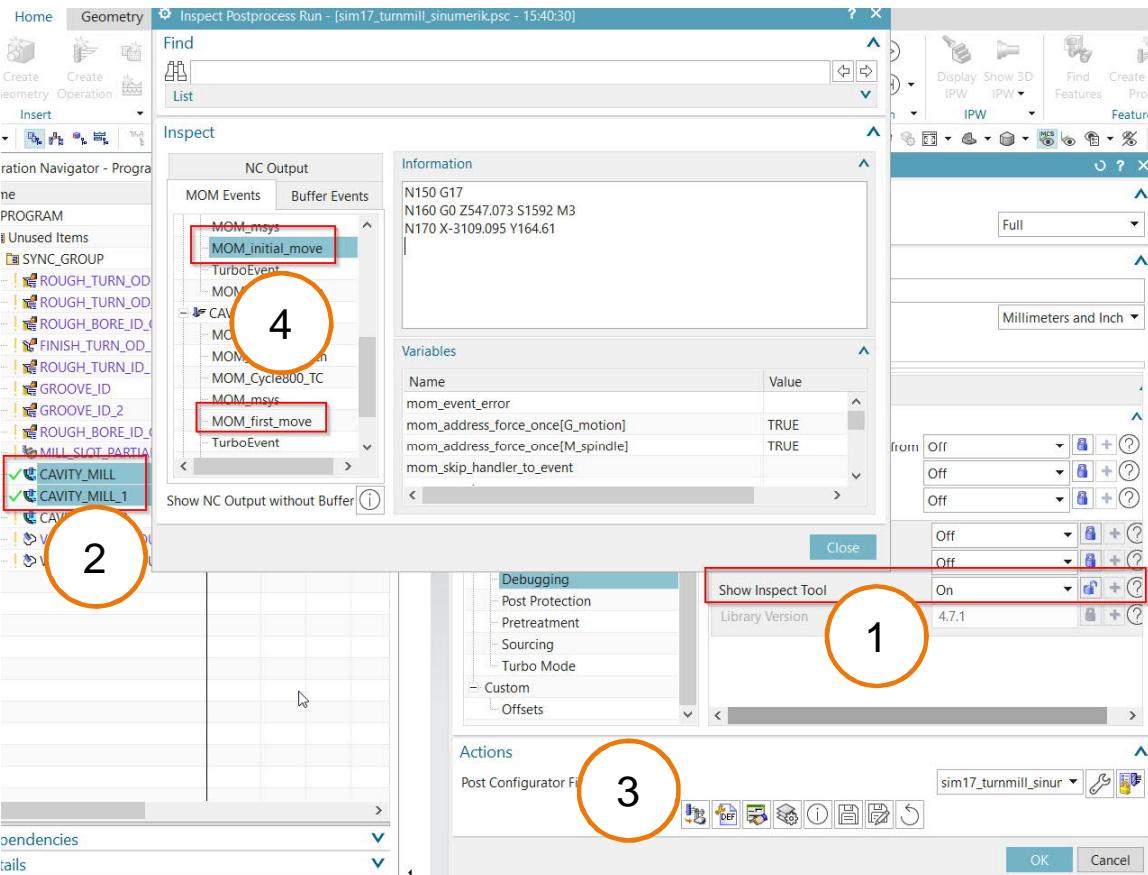


1. Create and define a new custom node and group for property settings
2. Create the property which contains different kind of output for additional offsets
3. Check UI that the property is visible and options ATRANS or WaCo L9958 can be chosen by the user

Task 1

Solution 6/15 – Get Buffers to modify

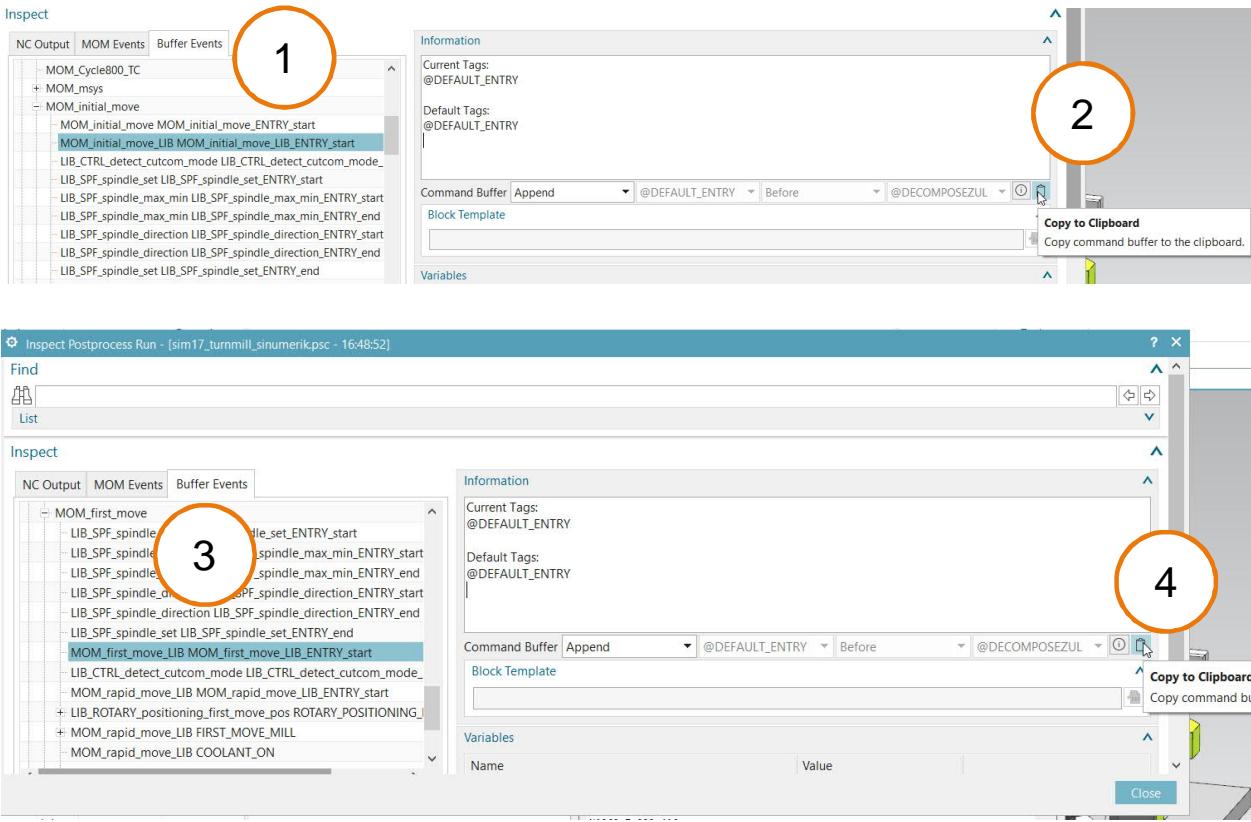
SIEMENS
Ingenuity for life



1. Switch On the Inspect Tool to find the customization point for implementing the Offset output, in this case we should output the additional offsets before the first motion in each operation
2. Select 2 operations, ideally without tool change to get the MOM_initial_move and MOM_first_move event
3. Use integrated post process option
4. Search mentioned MOM events in the MOM Events tab

Task 1

Solution 7/15 – Copy Buffers to modify



1. Because we need both MOM events for the customization use the **MOM_initial_move_LIB_entry_start** with append or prepend command
2. Copy to clipboard and insert into service file (do not close the Inspect window)
3. Search in second operation for the **MOM_first_move_LIB_entry_start** buffer and select the buffer
4. Use copy to clipboard and insert into service file

Task 1

Solution 8/15 – Add new procedure

```

941 newStringProperty    Property   "Creates a default procedure body"
942 newProc               Property   Tcl
943 newUIGroup            UI
944 newUINode             UI
945 newUIObject            UI
946 newObject              Property
947 newBoolProperty       Property
948 newIntProperty        Property
949 new
950
951 new
952

```

1: A context menu is open over the 'new' command, showing options like 'Creates a default procedure body', 'Tcl', 'UI', etc. The 'Tcl' option is selected.

```

946 LIB_GE_command_buffer_edit_append MOM_first_move_LIB MOM_first_move_LIB_ENTRY_start <code> <tag>
947 LIB_GE_command_buffer_edit_append MOM_initial_move_LIB MOM_initial_move_LIB_ENTRY_start <code> <tag>
948
949
950
951 #-
952 proc Cst_AdditionalOffsets {
953 #-
954
955 }
956

```

2: The 'Cst_AdditionalOffsets' procedure has been renamed from 'new'. The original 'new' command and its associated code and tag are highlighted with red boxes.

```

946 LIB_GE_command_buffer_edit_append MOM_first_move_LIB MOM_first_move_LIB_ENTRY_start Cst_AdditionalOffsets _AdditionalOffsetTag
947 LIB_GE_command_buffer_edit_append MOM_initial_move_LIB MOM_initial_move_LIB_ENTRY_start Cst_AdditionalOffsets _AdditionalOffsetTag
948
949
950
951 #-
952 proc Cst_AdditionalOffsets {} {
953 #-
954
955 } MOM_output_literal ";--->Test"
956

```

3: The 'Cst_AdditionalOffsets' procedure has been modified to include a parameter and a call to 'MOM_output_literal'. The 'MOM_output_literal' command and its argument are highlighted with red boxes.

4: The 'MOM_output_literal' command is highlighted with a red box, indicating the final step of testing the temporary output.

1. Create a new procedure body with Autocompletion functionality
2. Rename the procedure, this one is called later in the Buffer
3. Replace <code> and <tag> by the procedure name and the tag which should be have a spoken name
4. For testing work temporary with a MOM_output_literal command to validate with post process that output is on correct place

```

N78 SUPA X=-X_HOME Y=-Y_HOME
N79 G0 Z=0 POS(33000.)
N80 G55
N10 D1
N110 G9 G90
N120 SETHS(1)
N130 M00 CYCLE00001 "BC1_HEAD",0,27,0,0,0,-22.8231,-2.4133,0,0,0,1,0
N150 ;--->Test
N160 G0
N170 G0 Z547.073 S1592 M3
N180 X-3109.095 Y164.61
.....
.....
N680 D1
N690 G17
N760 CYCLE0001 "BC1_HEAD",0,27,0,0,0,0,1.8594,-23.5801,0,0,0,1,0
N710 ;--->Test
N720 G0 Z298.62 G1592 U3
N740 X210.393 Y4015.614
N760 V-110.677 V1442 K00 7.11 A67

```

Task 1

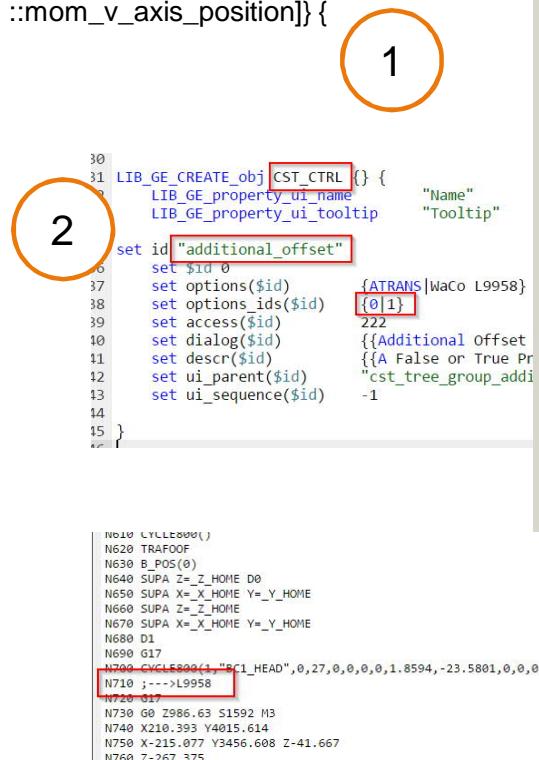
Solution 9/15 – Tcl procedure property logic

```

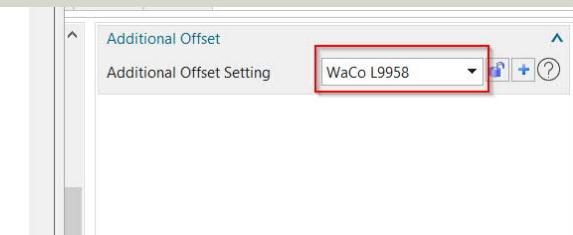
#-----
proc Cst_AdditionalOffsets {} {
#-----
#if UDE is not attached to program or operation, set values to 0 otherwise to avoid Tcl error
if {[!info exists ::mom_w_axis_position] || ![info exists ::mom_v_axis_position]} {
    set ::mom_w_axis_position 0
    set ::mom_v_axis_position 0
}

#ask property value setting to decide which output
switch [CST_CTRL additional_offset] {
    0 {
        #do something with ATRANS
        MOM_output_literal ";--->ATRANS"
    }
    1 {
        #do something with L9958
        MOM_output_literal ";--->L9958"
    }
    default {
        #do something
    }
}
}

```



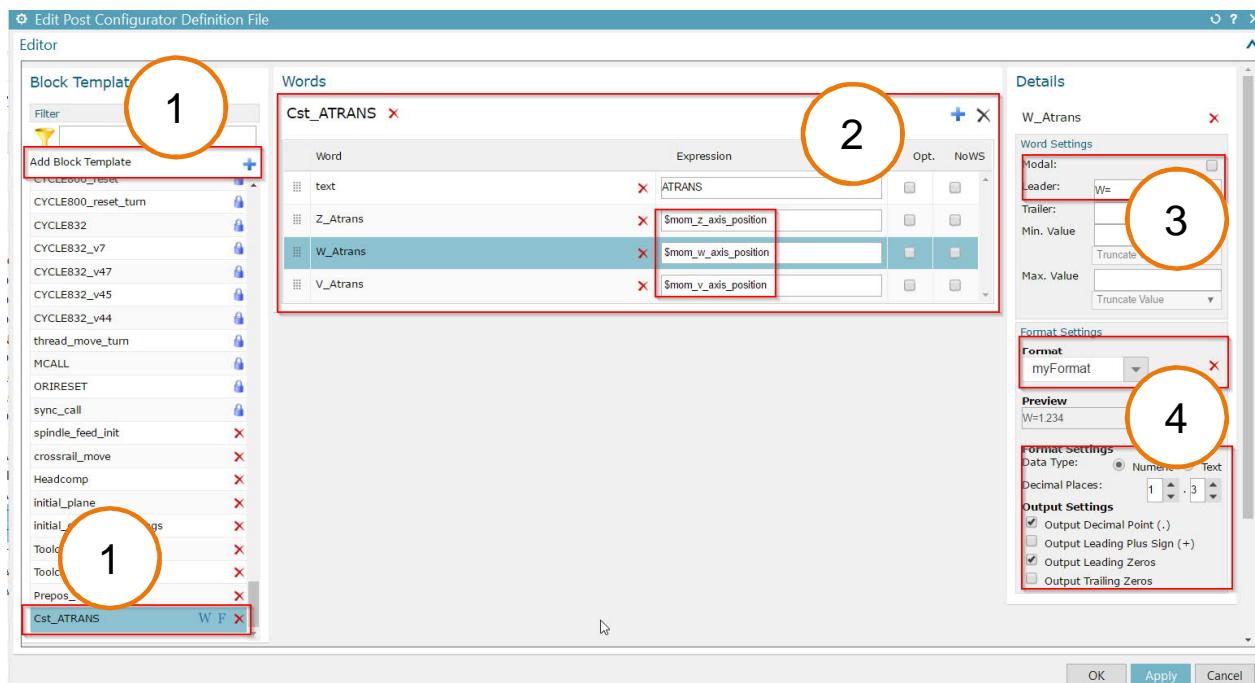
1. First check with a condition if Ude variables are exists, otherwise set those variables to 0 (global variables!)
2. Ask the value of the defined property in a switch block, will return in this case 0 or 1
3. For testing simply output the MOM_output_literal and post process. Change the property to the other value and it should run into second case



Task 1

Solution 10/15 – Add Block Template ATRANS

SIEMENS
Ingenuity for life



1. Add a new Block Template (take care that when creating a new Block Template it will use the last selected)
2. Add text address for ATRANS word and create new address for translation for Z,W,V. The expressions are linked to the mom variables of the UDE and the additional mom_z_axis_position
3. Set the leader and remove modal to get the output everytime even if values are the same
4. Create a new format as double with output leading zeros and assign to all created addresses

Task 1

Solution 11/15 – Add Block Template ATRANS movement

The screenshot shows the Siemens Post Configurator interface. On the left, the 'Editor' pane displays a Tcl script for 'Cst_AdditionalOffsets'. The code includes logic to handle 'ATRANS' movements based on property settings. A large orange arrow points from the script to the 'Output' pane on the right.

Script (Editor):

```

151 #-----#
152 proc Cst_AdditionalOffsets {} {
153 #
154 #if UDE is not attached to program or operation,
155 if {[info exists ::mom_w_axis_position] || ![in:
156     set ::mom_w_axis_position 0
157     set ::mom_v_axis_position 0
158     set ::mom_z_axis_position 0
159 }
160
161 #ask property value setting to decide which output
162 switch [CST_CTRL additional_offset]
163     0 {
164         #do something with ATRANS
165         MOM_output_literal ";"--->ATRANS
166         MOM_do_template Cst_ATRANS
167     }
168     1 {

```

Output (Output pane):

```

N150 ;--->ATRANS
N160 ATRANS Z=0. W=0. V=0.
N170 G17
N180 G0 Z547.073 S1592 M3
N190 X-3109.095 Y164.61

```

Block Template (Bottom Left):

- Block Template list: crossrail_move, Headcomp, initial_plane, initial_operation, Toolchange_L, Toolchange_C, Prepos_Traori, Cst_ATRANS, Cst_ATRANS_pos (highlighted).
- Details panel: Word: Cst_ATRANS_pos, Expression: Smom_sys_rapid_code, Value: 0, Opt.: NoWS.
- Details panel: Word: G_motion, Modal: G (highlighted), Leader: G, Min. Value: Truncate Value, Max. Value: Truncate Value.

Numbered Callouts:

- Callout 1: Points to the line 'set ::mom_z_axis_position 0' in the script.
- Callout 2: Points to the 'MOM_do_template Cst_ATRANS' line in the script.
- Callout 3: Points to the 'Output' pane showing the generated G-code.
- Callout 4: Points to the 'Cst_ATRANS_pos' entry in the Block Template list.
- Callout 5: Points to the 'Word' table in the Block Template details panel.

1. Add additional mom_z_axis_position variable in the code (will be calculated automatically later on)
2. Calling the created Block template in Tcl if property is set to ATRANS
3. Run post process and check first output
4. Create second Block Template for the positioning movement after ATRANS (create copy from existing ATRANS)
5. Add existing G_motion address with standard rapid code (e.g. copy from rapid_move template) and set expressions to zero

Task 1

Solution 12/15 – Add Z compensation logic

The screenshot shows a script editor window with several lines of Tcl code. A dialog box titled "User Defined Events" is open, listing various events like "Position W and V". Three numbered circles (1, 2, 3) point to specific areas: circle 1 points to the "Position W and V" event in the dialog; circle 2 points to the "Events Available" list in the dialog; and circle 3 points to a red box highlighting the line of code where Z-axis compensation is calculated.

```
961 #ask property value setting to decide which output
962 switch [CST_CTRL additional_offset] {
963     0 {
964         #do something with ATRANS
965         MOM_output_literal ";--->ATRANS"
966         MOM_do_template Cst_ATRANS
967         MOM_do_template Cst_ATRANS_pos
968     }
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
961 #ask property value setting to decide which output
962 switch [CST_CTRL additional_offset] {
963     0 {
964         #do something with ATRANS
965         MOM_output_literal ";--->ATRANS"
966         MOM_do_template Cst_ATRANS
967         MOM_do_template Cst_ATRANS_pos
968     }
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
952 proc Cst_AdditionalOffsets {} {
953 #-----
954 #if UDE is not attached to program or operation, set values to 0 otherwise to avoid Tcl error
955 if {![info exists ::mom_w_axis_position] || !*[info exists ::mom_v_axis_position]} {
956     set ::mom_w_axis_position 0
957     set ::mom_v_axis_position 0
958     set ::mom_z_axis_position 0
959 } else {
960     #calculate Z-axis compensation
961     set ::mom_z_axis_position [expr ($::mom_w_axis_position + $::mom_v_axis_position) * -1]
962 }
```

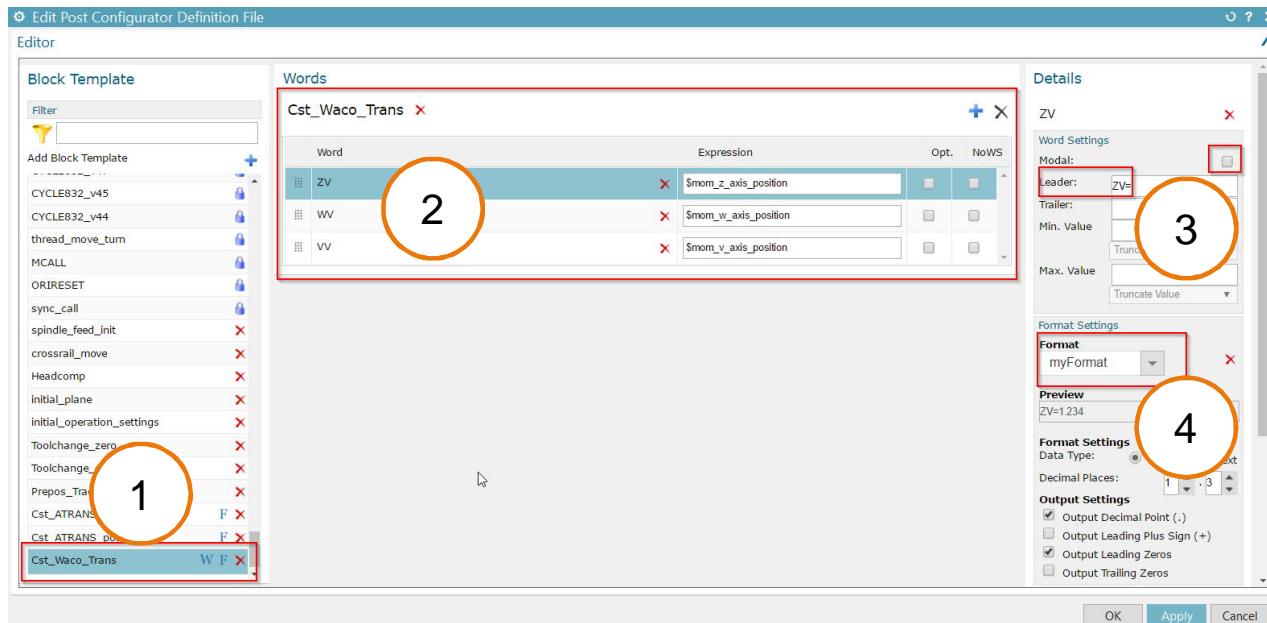
1. Calling additional Block Template in Tcl and run the post process to check result
2. Add the created UDE to the operation and add values for W(200) and V(100)
3. Add additional Tcl code to calculate the Z compensation if the Ude is attached to an operation
4. Check the result. All values of Z, W and V together should be equal 0 in the ATRANS line

The screenshot shows a script editor window with several lines of G-code. A status bar at the bottom displays "N150 ;--->ATRANS N160 ATRANS Z=-300. W=200. V=100. N170 G0 Z=0. W=0. V=0. N180 G17 N190 G0 Z547.073 S1592 M3". A numbered circle (4) points to the "V=100" value in the G-code.

```
N150 ;--->ATRANS
N160 ATRANS Z=-300. W=200. V=100.
N170 G0 Z=0. W=0. V=0.
N180 G17
N190 G0 Z547.073 S1592 M3
```

Task 1

Solution 13/15 – Add Block Template L9958



1. Add new Block Template for L9958 output
2. Create new Addresses for ZV, WV and VV with using same mom-variables
3. Setup the leader and remove modal flag
4. Reuse created format from ATRANS Block template

Task 1

Solution 14/15 – Add L9958 output

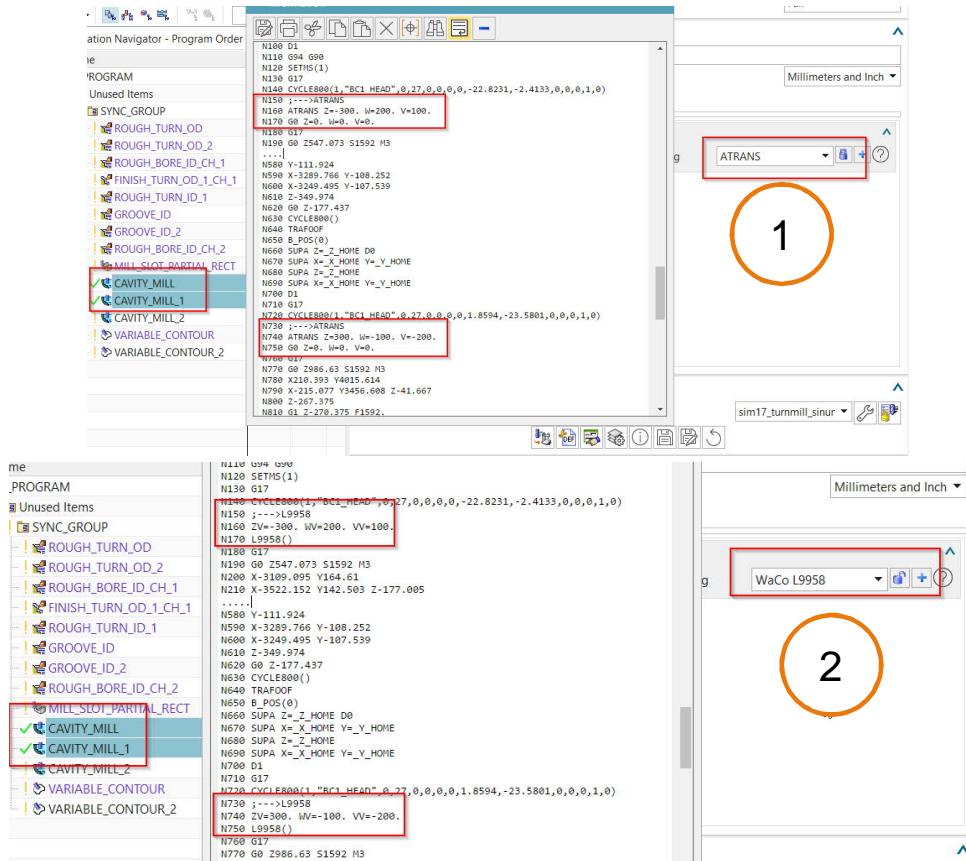
```
963  
964 #ask property value setting to decide what to do  
965 switch [CST_CTRL additional_offset] {  
966     0 {  
967         #do something with ATRANS  
968         MOM_output_literal ";--->ATRANS"  
969         MOM_do_template Cst_ATRANS  
970         MOM_do_template Cst_ATRANS_pos  
971     }  
972     1 {  
973         #do something with L9958  
974         MOM_output_literal ";--->L9958"  
975         MOM_do_template Cst_Waco_Trans  
976         MOM_output_literal "L9958\\(\\)"  
977     }  
978     default {  
979         #do something  
980     }  
981 }
```

1. Add Tcl code in case of L9958 of property definition
2. Execute the created Block template and output the subprogram line

Task 1

Solution 15/15 – Validate settings

SIEMENS
Ingenuity for life



1. Add the UDE to 2 operations with different values and run post process with ATRANS setting and validate result
2. Run post process with L9958 option and validate result

Task 1

Conclusion Solution and Benefits



1. Should be reused in a separate layer to minimize implementation effort for next post processor
2. One time effort higher, benefit of extremely time decrease for next projects
3. Take care that Cycle800 must be turned off when working with ATRANS (not included in this task)
4. Workflow is similar to every other UDE
5. Learned in this task :
 - Using UDE editor
 - Using DEF editor
 - Using Tcl editor and Inspect Tool
 - Create and connect created properties
 - Start thinking about generic solutions

Task 2



Collect the information from the created UDE with pretreatment:

Conditions:

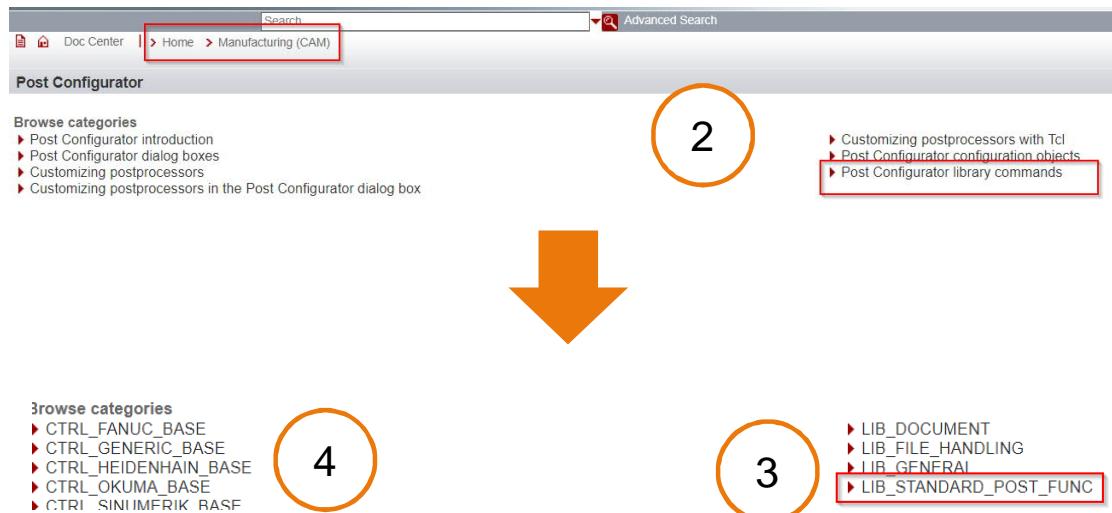
- If the UDE is attached to an operation the values of the transformation are output in the header with operation information (name and used tool)
- The user can control the output with a property to switch on/ off

Task 2

Solution 1/7 – Use PC online documentation

1

https://docs.plm.automation.siemens.com/tdoc/nx/1872/nx_help/#uid:xid1128418:index_mfgpostconfig



1. Use official online documentation for Post Configurator to search for standard procedures for reuse
2. On the Post Configurator documentation page use library commands section
3. Most of standard functionalities are defined in the LIB_STANDARD_POST_FUNC section which are controller independent. It's recommend to take 30min to read thru all of this to get a feeling what is available
4. CTRL specific definitions and general File handling are defined in different areas

Task 2

Solution 2/7 – Use PC online documentation



1. First it's needed to add the new UDE variables to the pretreatment to output them in the header. Open the LIB_SPF_pretreatment_add_var to read documentation and how it should be used
2. To add new variables it's needed to create a new procedure with a fixed name
3. LIB_SPF_get_pretreatment is to get values. Keep it in mind for later from documentation

Task 2

Solution 3/7 – Add variables to pretreatment

```

983 #
984 #-----#
985 proc LIB_SPF_pt_additional_variables_cst {} {
986 #
987 |
988 }
989

Format LIB_SPF_pretreatment_add_var <proc_name> <var_name> <eventnumber>

Parameters proc_name
Procedure where the variable value is memorized.

var_name
Name of the variable to memorize

eventnumber
Valid options are
0 - Memorize variable value. This is the most used case. The current value of the given variable is memorized at the specified event name. Use this option to memorize variables which are only set once per operation. Like additional tool or operation information (for example, mom_tool_diameter) The memorized value can be simply accessed with LIB_SPF_get_pretreatment mom_variable_name. See examples 1 and 2 for more information.
1 - Memorize variable value with event number. This option is used when the specified event might be called several times per operation and when you want to memorize every single variable value from each call. This could be used when you want to access all insert statements that could be placed on an operation. To access the memorized values you have to query directly the pretreatment variables: lib_pretreatment(tag_id,event_number,mom_variable_name). See examples 3 and 4 for more information.
2 - Memorize variable. This option is used to memorize the last value of a given variable in the whole post run. The last memorized value can be simply accessed with LIB_SPF_get_pretreatment mom_variable_name. See example 5 for more information.
This argument is optional. Default value is 0.

984 #-----#
985 proc LIB_SPF_pt_additional_variables_cst {} {
986 #
987 LIB_SPF_pretreatment_add_var MOM_Position_WV mom_w_axis_position 0
988 LIB_SPF_pretreatment_add_var MOM_Position_WV mom_v_axis_position 0
989 LIB_SPF_pretreatment_add_var MOM_Position_WV mom_z_axis_position 0
990 }
991

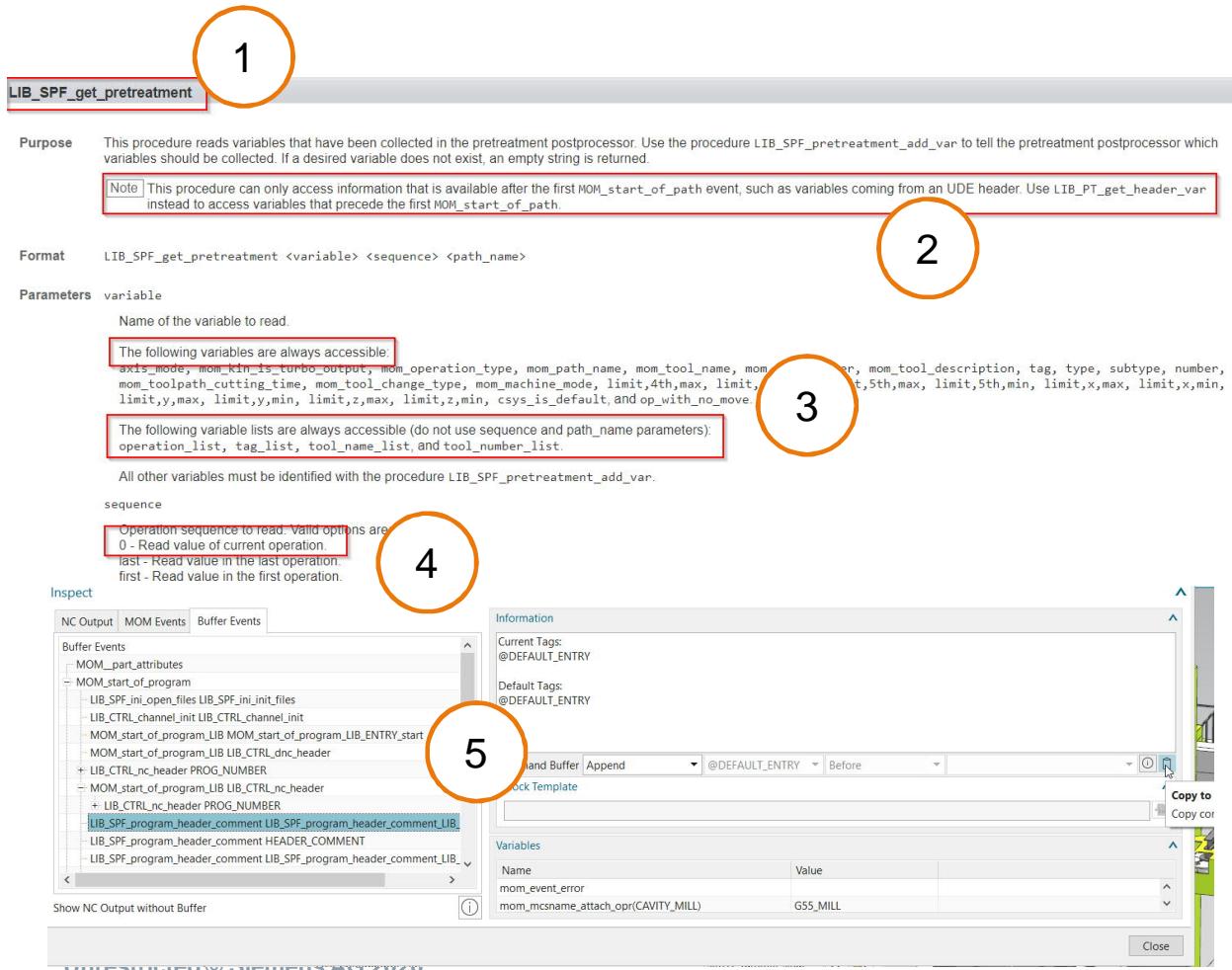
992
888 proc MOM_Position_WV { } {
889     global mom_w_axis_position
890     global mom_v_axis_position
891
892     #Put your UDE Handler Tcl here
893     set ::mom_z_axis_position [expr ($::mom_w_axis_position + $::mom_v_axis_position) * -1]
894 }
895

```

- 1 Create a new procedure with the name from documentation
- 2 Use documentation to find out how to collect and the arguments needed
- 3 Variables can be collected from called MOM events. The UDE is an own event and the variables are accessible first time in this event. Add the MOM event name of the UDE and all variables to collect
- 4 Because the Z-axis compensation was calculated later in the procedure it's needed to add the calculation in the MOM event handler of UDE to have access in pretreatment

Task 2

Solution 4/7 – Output collected values in header



1. Open documentation for LIB_SPF_get_pretreatment
2. Take care of the Note how to collect variables outside of operations (e.g. if the UDE is attached to the program group)
3. A set of variables is always collected and multiple lists are available to cycle thru the collected values. The operation list will be used in next step
4. Argument which values should be collected
5. Run Inspect Tool to get Buffer for the output, e.g. the program header comment buffer

Task 2

Solution 5/7 – Add logic to cycle thru collected values

```
990 J
991 LIB_GE_command_buffer_edit_append LIB_SPF_program_header_comment LIB_SPF_program_header_comment_LIB_ENTRY_start OutputPretreatmentVars AdditionalPTVars
992 #-
993 proc OutputPretreatmentVars {} {
994 #-
995 foreach e [LIB_SPF_get_pretreatment operation_list] {
996 #Format and output listing
997 }
998 }
```

```
#-----
#-----
proc OutputPretreatmentVars {} {
#-----
foreach e [LIB_SPF_get_pretreatment operation_list] {
#Format and output listing
    MOM_output_text ";Operation Name:[LIB_SPF_get_pretreatment mom_path_name 0 $e] Toolname:[LIB_SPF_get_pretreatment mom_tool_name 0 $e]"
    MOM_output_text ";Additive offset Z=[LIB_SPF_get_pretreatment mom_z_axis_position 0 $e]"
    MOM_output_text ";Additive offset V=[LIB_SPF_get_pretreatment mom_v_axis_position 0 $e]"
    MOM_output_text ";Additive offset W=[LIB_SPF_get_pretreatment mom_w_axis_position 0 $e]"
    MOM_output_text ";"
}
}
```

Unrestricted © Siemens AG 2020

1

2

3

4

1. Call the new procedure in the program header comment buffer and add a Tagname
2. With foreach from automatically collected operation list cycle thru all operations
3. In each operation output the mom_path_name and mom_tool_name which are also collected automatically
4. Output the variables added to the pretreatment from the UDE definition

Task 2

Solution 6/7 – Validate Output

```

1
1 ! GROOVE_ID
1 ! GROOVE_ID_2
1 ! ROUGH_BORE_ID_CH_2
1 ! MILL_SLOT_PARTIAL_RECT
1 CAVITY_MILL
1 CAVITY_MILL_1
1 CAVITY_MILL_2
1 VARIABLE_CONTOUR
1 VARIABLE_CONTOUR_2

=====
%_N_var_MPFF
;Operation Name:CAVITY_MILL Toolname:MILL_DIA_50
;Additive offset Z=-300.0
;Additive offset V=100.0
;Additive offset W=200.0
;
;Operation Name:CAVITY_MILL_1 Toolname:MILL_DIA_50
;Additive offset Z=300.0
;Additive offset V=-200.0
;Additive offset W=-100.0
;
N10 DEF REAL _X_HOME, _Y_HOME, _Z_HOME

920 set id "cst_tree_group_additional_offset"
921 set $id "0"
922 set datatype($id) "GROUP"
923 set access($id) 222
924 set dialog($id) {{Additional Offset}}
925 set descr($id) {{Group description}}
926 set group_status($id) 1
927 set ui_parent($id) "cst_tree_node_offset"
928 set ui_sequence($id) -1

929
930 set id "cst_tree_node_header"
931 set $id "0"
932 set datatype($id) "NODE"
933 set access($id) 222
934 set dialog($id) {{Header Settings}}
935 set descr($id) {{Node description}}
936 set group_status($id) 0
937 set ui_parent($id) "cst_tree_root"
938 set ui_sequence($id) -1
939

940 set id "cst_tree_group_header"
941 set $id "0"
942 set datatype($id) "GROUP"
943 set access($id) 222
944 set dialog($id) {{Header Settings}}
945 set descr($id) {{Group description}}
946 set group_status($id) 1
947 set ui_parent($id) "cst_tree_node_header"
948 set ui_sequence($id) -1
949 }

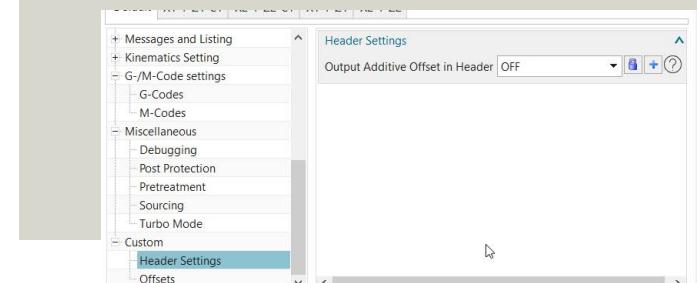
2
950 LIB_GE_CREATE_obj CST_CTRL {} {
951 LIB_GE_property_ui_name "Name"
952 LIB_GE_property_ui_tooltip "Tooltip"
953
954 set id "additional_offset"
955 set $id 0
956 set options($id) {ATRANS|WaCo L9958}
957 set options_ids($id) {0|1}
958 set access($id) 222
959 set dialog($id) {{Additional Offset Setting}}
960 set descr($id) {{A False or True Property}}
961 set ui_parent($id) "cst_tree_group_additional_offset"
962 set ui_sequence($id) -1
963

3
964 set id "header_offset_output"
965 set $id 0
966 set options($id) {OFF|ON}
967 set options_ids($id) {0|1}
968 set access($id) 222
969 set dialog($id) {{Output Additive Offset in Header}}
970 set descr($id) {{Output additive offset values from UDE in header with Tool name and Operation name}}
971 set ui_parent($id) "cst_tree_group_header"
972 set ui_sequence($id) -1
973
974
975 }

```

Unrestricted © Siemens AG 2020

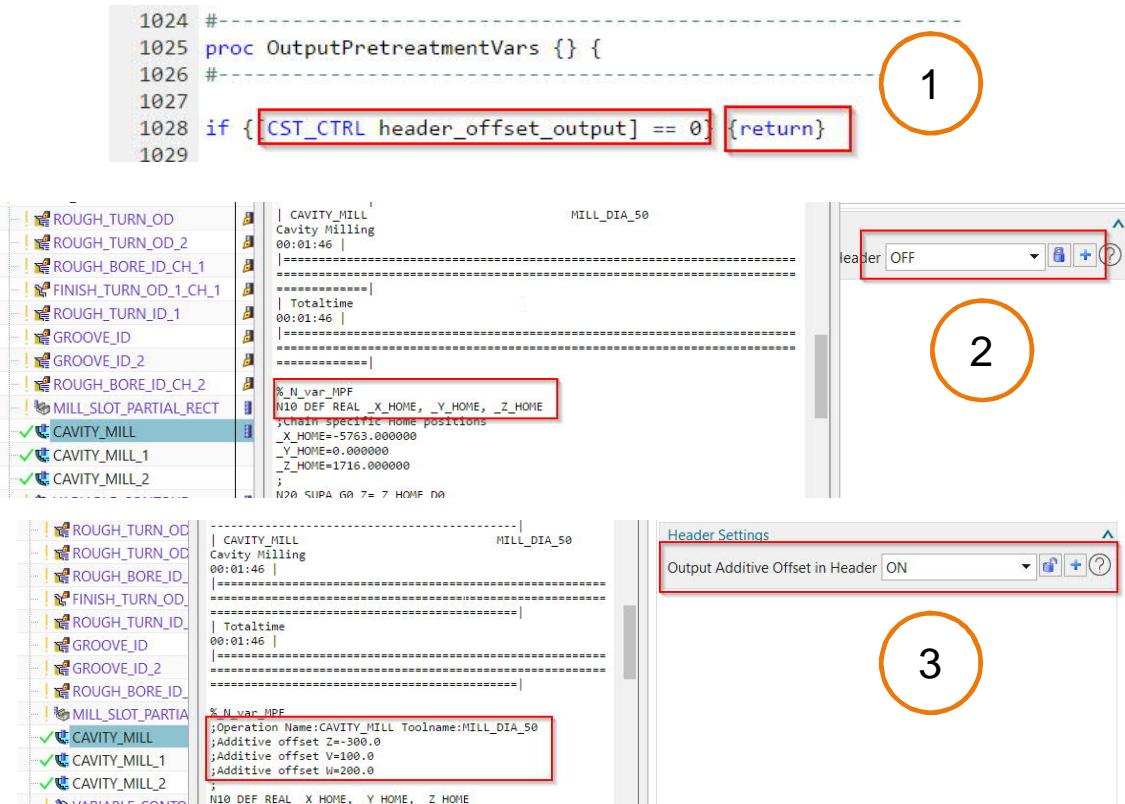
- Run the post process with 2 operations where the UDE is attached. Values with operation name and tool name are output in the header
- Add new group for Header settings in the Custom_tree object definition
- Add a new property to control the output



Siemens PLM Software

Task 2

Solution 7/7 – Connect Property to command and Validate



1. Add logic to return if the created property is set to OFF in the OutputPretreatmentVars procedure which contains the output of values
2. Validate with option OFF, no output should appear
3. Validate with option ON, output should appear if UDE is attached to operation

Task 2

Conclusion Solution and Benefits



1. Should be reused in a separate layer to minimize implementation effort for next post processor
2. One time effort higher, benefit of extremely time decrease for next projects
3. Documentation contains a lot of predefined functions for reuse, take 30min or 1hour to go thru documentation to get a feeling what is available
4. Workflow is similar for each variable which is later available but needed in beginning of program
5. Learned in this task:
 - Using Documentation
 - Using Pretreatment
 - Using Tcl editor and Inspect Tool
 - Create and connect created properties

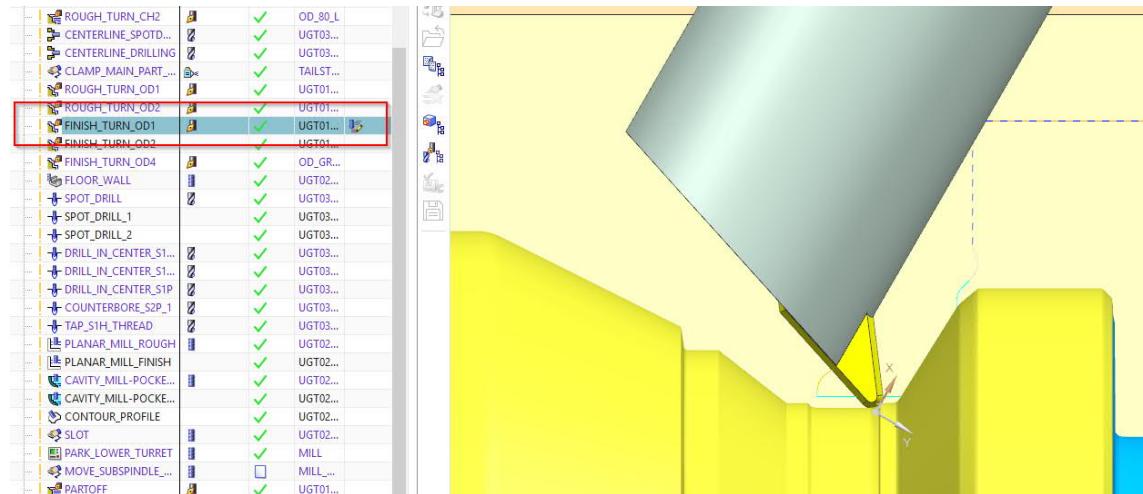
Task 3

SIEMENS
Ingenuity for life

The customer request a B-axis turning capability in the postprocessor (sim15 Sinumerik):

Conditions:

- Use the [...\\Handsout\\CAM_parts\\B_axis Turning\\sim15_millturn_setup_sinumerik_mm_dual_sync.prt](#) from training material
- Option is needed to switch on and off
- Activate and deactivate correctly in NC code, e.g. when switching back to milling mode



Unrestricted © Siemens AG 2020

```
N400 I1(50, /0, 180)
N500 G54
N600 TRANS X0, Y0, Z0.
N700 GETD(C4)
N800 TRAORI
N900 DIAMON
N100 SETMS(4)
N110 SPOS[1]=180.
N120 G0 X110, Y0, Z232.6 D1
N130 G95 S4=2000 M4=3
N140 X58.2
N150 G18 G3 G90 X53.4 Z230.2 -2.4 K0, F4
N160 G1 X52.4 Z230.2 B50.
N170 X52.4 Z235.885 B60.
N180 X52.4 Z244.8 B80.
N190 G3 X54, Z244.8 I0.8 K0.
N200 G1 X68.703 Z244.8 B70.
N210 X72.793 Z244.8
N220 G2 X74.844 Z225.225 I0, K1.45
N230 G1 X76.551 Z246.078
N240 X76.722 Z246.172
N250 G3 X83.484 Z246.467 I1.839 K-1.543 F.5
N260 G0 X110,
N270 Z232,
N280 TRAOF
N290 DIAMOF
N300 TRANS
N310 F4=5
N320 G97
N330 SUPA X598.7 D0
N340 PUSA V400 T400
```

Siemens PLM Software

Task 3

Solution – Implement B-axis turning

The screenshot shows the Siemens PLM Community homepage. At the top, there is a dark blue header with the Siemens logo and the tagline "Ingenuity for life". Below the header is a search bar with the placeholder "Search the community". Underneath the search bar are several navigation links: "Browse Community", "NX Manufacturing", "Forums ▾", "Blogs ▾", "Knowledge Bases ▾", and "Groups ▾". The "Groups" link is underlined, indicating it is the active section. In the main content area, there is a breadcrumb navigation: "Siemens PLM Community > NX Manufacturing > NX CAM Postprocessor Group > Webinar post processing 2019 - Session 04". A circled number "1" is overlaid on the screenshot.

<https://community.plm.automation.siemens.com/t5/NX-CAM-Postprocessor-Group/Webinar-post-processing-2019-Session-04/gpm-p/606377>



1. Webinar session for B-axis turning is available online and how to proceed
2. With NX1872 the sim15 millturn Sinumerik example contains a sample implementation of B-axis turning

A code editor window displays a snippet of G-code. The code includes several procedures and conditional statements related to B-axis turning. A circled number "2" is overlaid on the screenshot.

```
44 LIB_GE_command_buffer_edit_prepend $::mom_end_of_path_LIB $::mom_end_of_path_LIB_ENTRY_start {CheckTurnMode} _CheckMode
45 LIB_GE_command_buffer_edit_prepend $::mom_linear_move_LIB OUTPUT {CalculateBRotation} _CalculateB
47 LIB_GE_command_buffer_edit_prepend $::mom_circular_move_LIB OUTPUT {CalculateBRotation} _CalculateB
48
49 #-----
50 proc Detect_B_axis_mode {} {
51 #-
52 if {[string match "Turn Finishing" $::mom_operation_type]} {
53   if {[info exists ::mom_use_b_axis] && $::mom_use_b_axis == 2} {
54     MOM_output_literal "GETD(44)"
55     MOM_do_template traori
56     MOM_enable_address fourth_axis
57   } else {
58     MOM_disable_address fourth_axis
59   }
60 } else {
61   MOM_disable_address fourth_axis
62 }
63 }
64
65 #-
66 proc CalculateBRotation {} {
67 #-
68 #set initial B-axis init value for turning
69 if {[info exists ::mom_use_b_axis] && $::mom_use_b_axis == 2} {
70   set initial_baxis_value 0
71
72   set B_rotation_calculation [expr $::RAD2DEG*atan2($::mom_tool_axis(0),$::mom_tool_axis(2))]
73   set ::B_rotation [expr $initial_baxis_value + $B_rotation_calculation]
74   MOM_force once X Z
75 }
76 }
77
78 #-
79 proc CheckTurnMode {} {
80 #
81 if {[info exists ::mom_use_b_axis] && $::mom_use_b_axis == 2} {
82   MOM_do_template traorff
83   LIB_TURNING_mode "end"
84 }
85 }
```

Task 4



If the motion type is changing the customer request the output of the current motion type in the NC code.
(e.g. for deposition tools in stepover moves):

Conditions:

- It is not allowed to output this request with **Turbomode off option**
- Option is needed to switch on and off
- Property settings for M-code Deposition tool ON/ OFF needed
- M code is outputted on the correct location in NC code
- Count the numbers of change for stepover, engage, retract, cut, traversal and output this formatted in end of program (for sure option is needed in UI ☺)

Task 4

Solution 1/4 – Add Properties for reuse

```

949 set id "cst_tree_node_deposition"
950 set $id "0"
951 set datatype($id) "NODE"
952 set access($id) 222
953 set dialog($id) {{Deposition Settings}}
954 set descr($id) {{Node description}}
955 set group_status($id) 0
956 set ui_parent($id) "cst_tree_root"
957 set ui_sequence($id) -1
958
959 set id "cst_tree_group_deposition"
960 set $id "0"
961 set datatype($id) "GROUP"
962 set access($id) 222
963 set dialog($id) {{Deposition Settings}}
964 set descr($id) {{Group description}}
965 set group_status($id) 1
966 set ui_parent($id) "cst_tree_node_deposition"
967 set ui_sequence($id) -1
968
969 }

1004 set id "deposition_on_code"
1005 set $id 75
1006 set options($id) {*VALUE*}
1007 set options_ids($id) {*VALUE*}
1008 set datatype($id) "INT"
1009 set access($id) 222
1010 set dialog($id) {{Deposition On Code}}
1011 set descr($id) {{A numeric Property}}
1012 set ui_parent($id) "cst_tree_group_deposition"
1013 set ui_sequence($id) -1
1014
1015 set id "deposition_off_code"
1016 set $id 76
1017 set options($id) {*VALUE*}
1018 set options_ids($id) {*VALUE*}
1019 set datatype($id) "INT"
1020 set access($id) 222
1021 set dialog($id) {{Deposition Off Code}}
1022 set descr($id) {{A numeric Property}}
1023 set ui_parent($id) "cst_tree_group_deposition"
1024 set ui_sequence($id) -1
1025
1026

```

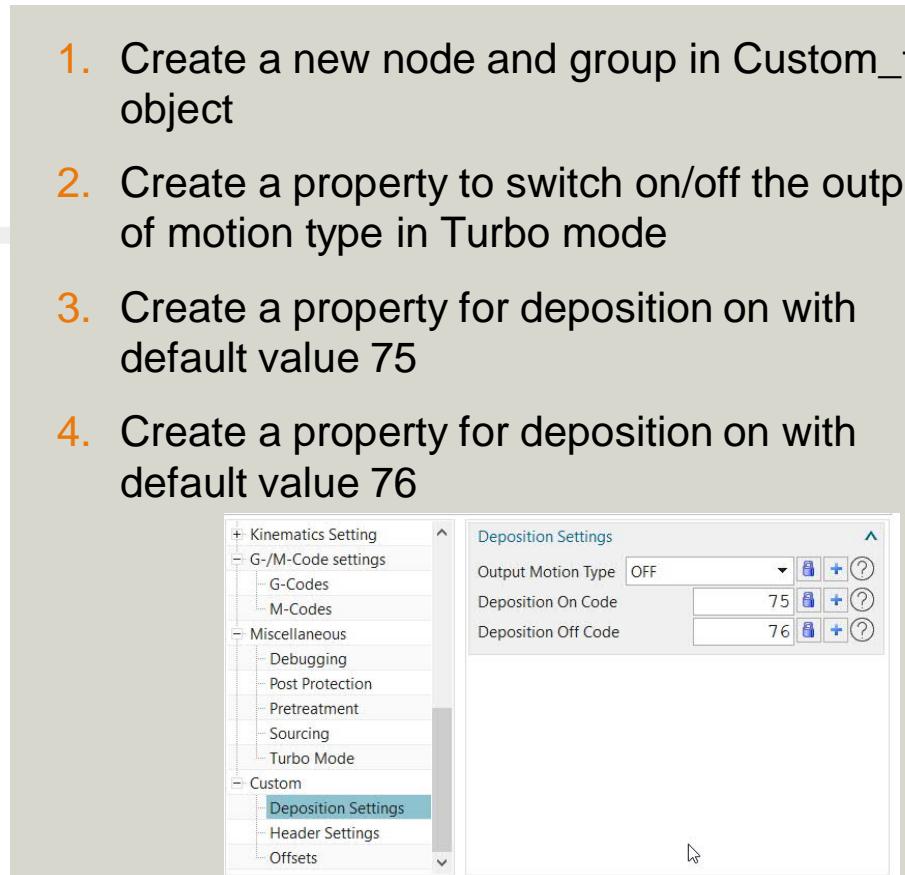
1

2

3

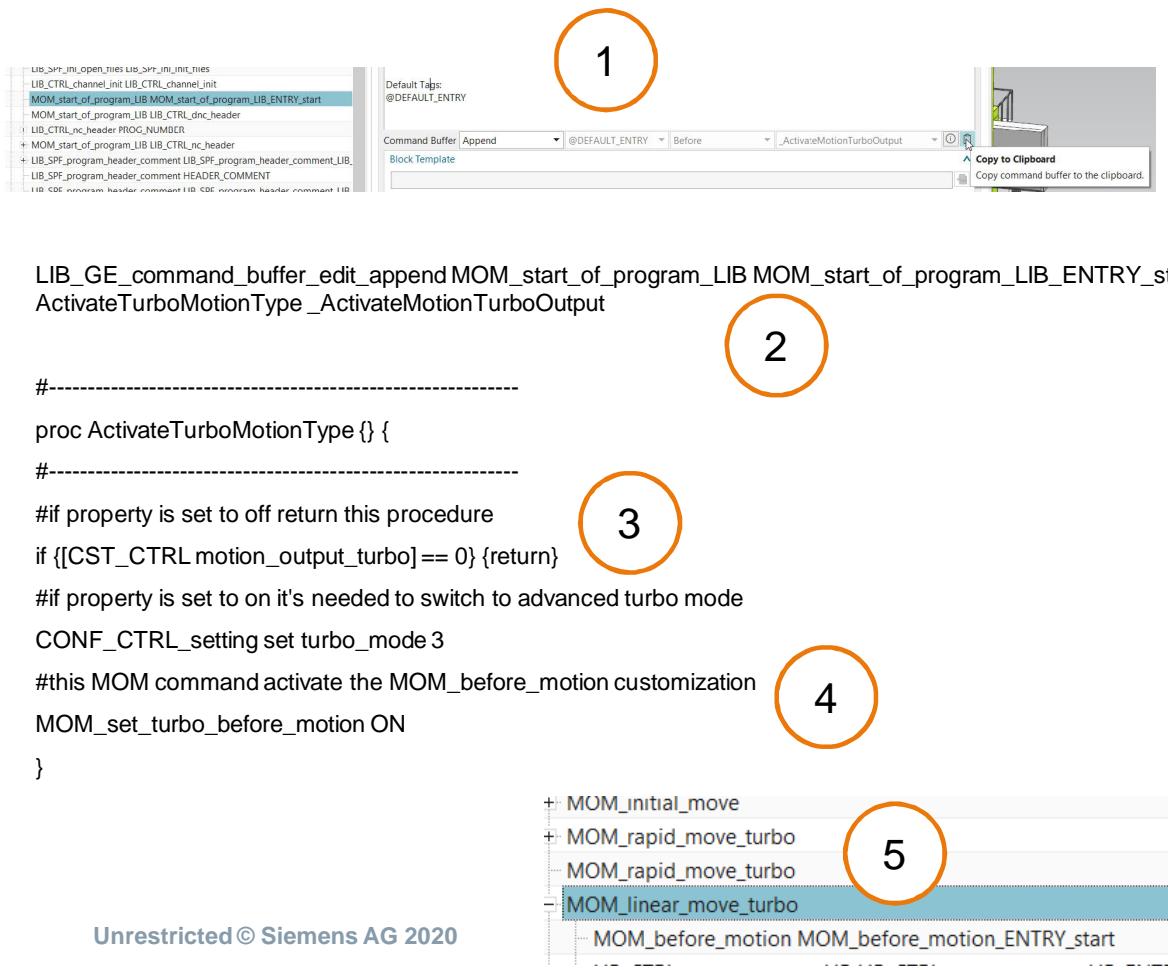
4

1. Create a new node and group in Custom_tree object
2. Create a property to switch on/off the output of motion type in Turbo mode
3. Create a property for deposition on with default value 75
4. Create a property for deposition off with default value 76



Task 4

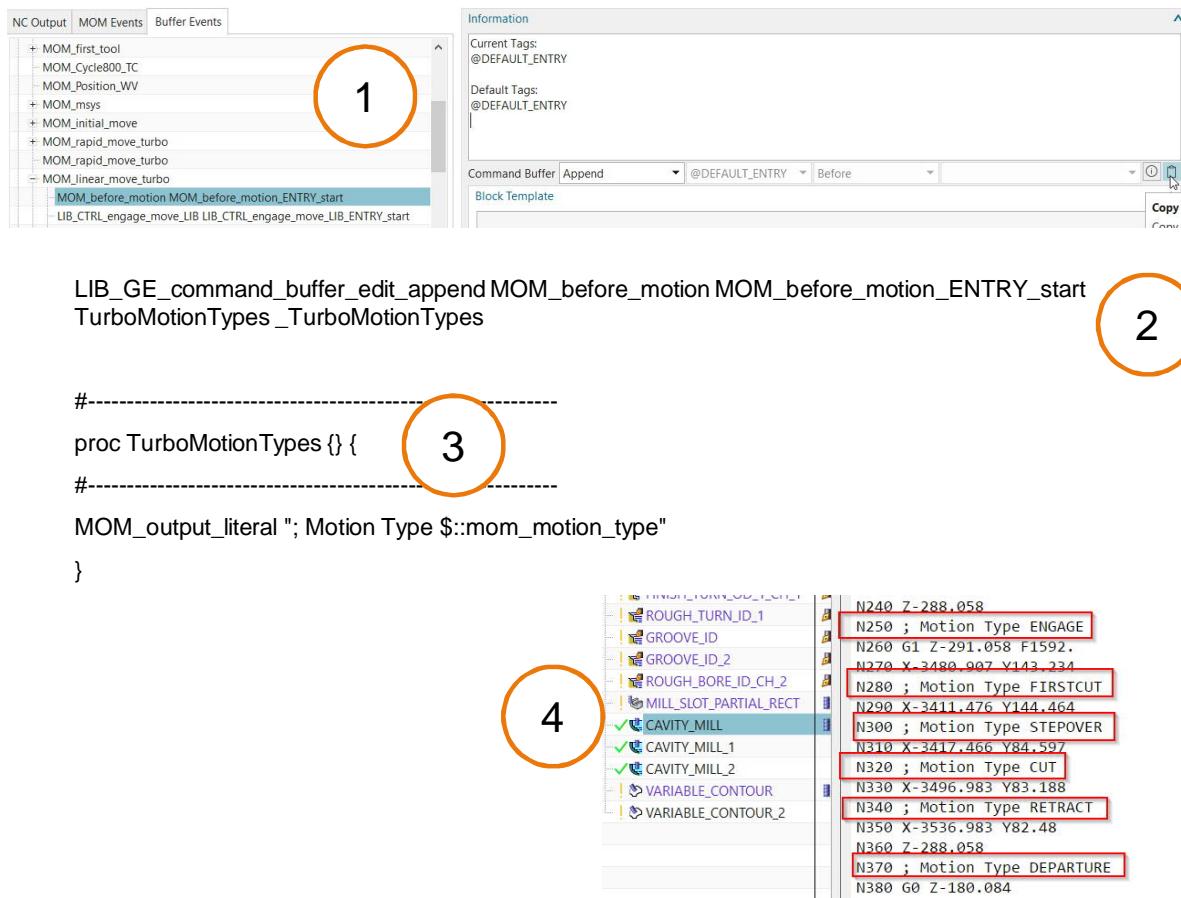
Solution 2/4 – Activate MOM_before_motion in Turbo mode



1. Run Inspect Tool and append new command in start of program to activate extended functionality with Turbo mode
2. Call procedure and define a Tag name with the additional customization
3. Dependent on property setting return this procedure or activate the motion output
4. To influence Turbo mode the turbo mode must switch to Advanced mode, set the property by Tcl and activate the Motion type with MOM_set_turbo_before_motion ON
5. Run the post process with Inspect Tool and instead of Turbo event Turbo events for each motion appears with customization possibilities

Task 4

Solution 3/4 – Output Motion types with Turbo



1. Run post process with Inspect Tool and use MOM_before_motion Buffer from the Turbo event to extend
2. Create procedure and call this in the Buffer. Add a Tag as identifier
3. Output the Motiontype simply by a MOM_output_literal
4. Run post process and check that now each motion change is outputted. Be aware of the fact that only motion changes will be outputted, means if multiple cut motions than only the first one will be outputted if there is no further motion type change

Task 4

Solution 4/4 – Add counting of changed motion types

```

1095 #-----  

1096 proc ActivateTurboMotionType {} {  

1097 #-----  

1098 #if property is set to off return this procedure  

1099 if {[CST_CTRL motion_output_turbo] == 0} {return}  

1100 #if property is set to on it's needed to switch to advanced turbo mode  

1101 CONF_CTRL_setting set turbo_mode 3  

1102 #this MOM command activate the MOM_before_motion customization  

1103 MOM_set_turbo_before_motion ON  

1104  

1105 set ::mom_cst_motionchange_cut 0  

1106 set ::mom_cst_motionchange_firstcut 0  

1107 set ::mom_cst_motionchange_traversal 0  

1108 set ::mom_cst_motionchange_approach 0  

1109 set ::mom_cst_motionchange_departure 0  

1110 set ::mom_cst_motionchange_engage 0  

1111 set ::mom_cst_motionchange_retract 0  

1112 set ::mom_cst_motionchange_stepover 0  

1113 }  

1134  

1135 #-----  

1136 proc Motionchanges {} {  

1137 #-----  

1138 if {[CST_CTRL motion_output_turbo] == 0} {return}  

1139 MOM_output_text "Cut changes: $::mom_cst_motionchange_cut"  

1140 MOM_output_text "Firstcut changes: $::mom_cst_motionchange_firstcut"  

1141 MOM_output_text "Traversal changes: $::mom_cst_motionchange_traversal"  

1142 MOM_output_text "Approach changes: $::mom_cst_motionchange_approach"  

1143 MOM_output_text "Departure changes: $::mom_cst_motionchange_departure"  

1144 MOM_output_text "Engage changes: $::mom_cst_motionchange_engage"  

1145 MOM_output_text "Retract changes: $::mom_cst_motionchange_retract"  

1146 MOM_output_text "Stepover changes: $::mom_cst_motionchange_stepover"  

1147 }

```

1

2

3

4

5

1. Add initial counter in the ActivateTurboMotionType procedure for each motion type
2. In the TurboMotionTypes add a switch condition and increment the counter by one if the motion type is changed
3. Use Inspect Tool and last Buffer in end of program to output the motion type changes
4. Activate this output only if the property is switched On, otherwise it may will result in Tcl error
5. Output all motion type changes counter variables

GROOVE_ID_2
ROUGH_BORE_ID_CH_
MILL_SLOT_PARTIAL_RE
CAVITY_MILL
CAVITY_MILL_1
CAVITY_MILL_2
VARIABLE_CONTOUR
VARIABLE_CONTOUR_2

```

N1070 M30
;Cut changes: 9
;Firstcut changes: 3
;Traversal changes: 0
;Approach changes: 3
;Departure changes: 3
;Engage changes: 3
;Retract changes: 3
;Stepover changes: 9

```

Task 4

Conclusion Solution and Benefits



1. Using advantage of Turbo mode for high performance with possibilities to customize even in the motions, especially for additive
2. Should be reused in a separate layer to minimize implementation effort for next post processor
3. One time effort higher, benefit of extremely time decrease for next projects
4. Further information to Turbo mode can be found in documentation
5. This solution not contains the call of M-commands for deposition but can be easily added by your own now
6. Learned in this task:
 - Using Turbo mode advanced
 - Using Tcl editor and Inspect Tool
 - Create and connect created properties

Task 5



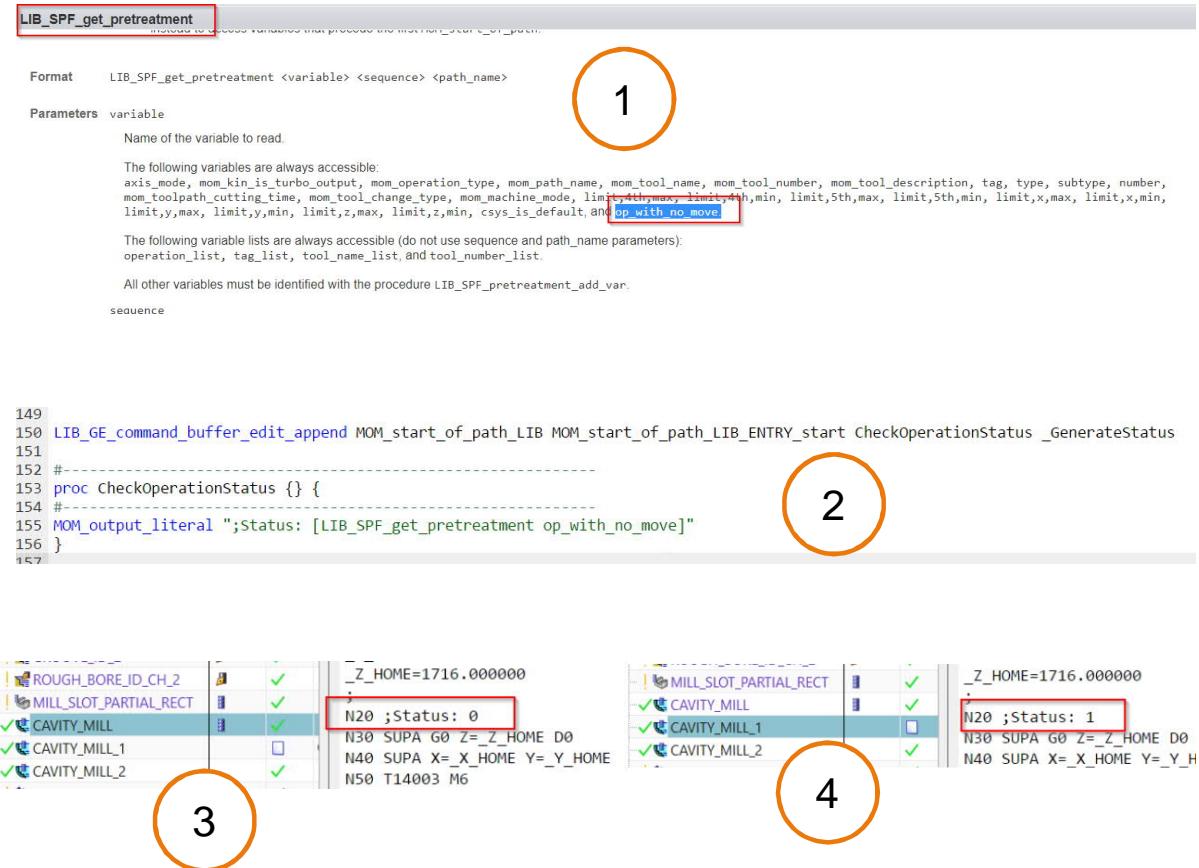
Warning if toolpath is empty:

Conditions:

- Using standard Messagebox from Post Configurator library
- Option is needed in UI

Task 5

Solution 1/3 – Search needed variables



1. Search in NX documentation for variable which contains this information, if not found anything search standard functionality from Post Configurator documentation, in this case in pretreatment there is a variable `op_with_no_moves` which indicates toolpath is empty
2. Add a procedure in Start_of_path buffer and output the value from pretreatment to find out which values are returned
3. Post process an operation with toolpath will return 0
4. Post process an operation without toolpath will return 1

Task 5

Solution 2/3 – Ask status and create Message dialog

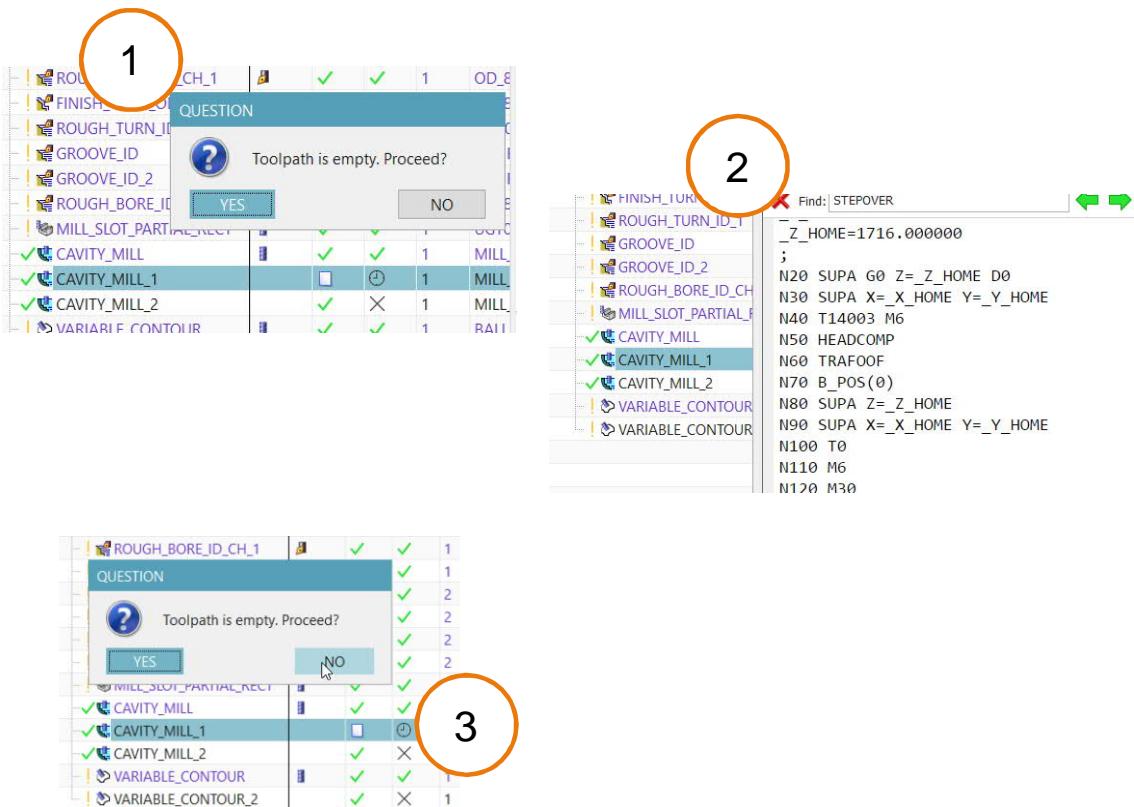
The screenshot shows two main sections. On the left is a help documentation page for the `LIB_GE_message_dialog` procedure. It includes sections for Purpose, Format, Parameters, Return value, and Examples. The Examples section shows a simple error message example and a 1 button dialog box example. On the right is a code editor displaying a Post Configurator script. The script contains a `proc` named `Checkoperationstatus` which checks if a toolpath is empty and displays a message dialog if so.

```
24 #-
25 proc Checkoperationstatus {} {
26 #
27 #MOM_output_literal "Status: [LIB_SPF_get_pretreatment op_with_no_move]"
28
29 if {[LIB_SPF_get_pretreatment op_with_no_move] == 1} {
30     set_response [LIB_GE_message_dialog "Toolpath Empty" "Toolpath is empty. Proceed?" "QUESTION" 2 YES NO]
31     if {$response == 2} {
32         LIB_SPF_abort_postrun
33     }
34 }
```

1. Use documentation of Post Configurator to find standard Message dialogs, `LIB_GE_message_dialog`
2. In the examples copy the sample with 2 buttons into service file
3. Create condition if `op_with_no_moves` the dialog comes up with yes and no question to proceed
4. If response no abort the post process run with additional standard command from Post Configurator libraries

Task 5

Solution 3/3 – Validate modifications



1. Run post process with operation and empty toolpath
2. If question is answered with yes some output should appear
3. If question is answered with no the post process is aborted without any output

Task 5

Conclusion Solution and Benefits



1. Using documentation to find standard commands instead of reinvent the wheel
2. Should be reused in a separate layer to minimize implementation effort for next post processor
3. One time effort higher, benefit of extremely time decrease for next projects
4. Standard methods for messages are available

5. Learned in this task:
 - Using Documentation
 - Using Tcl editor and Inspect Tool

Task 7



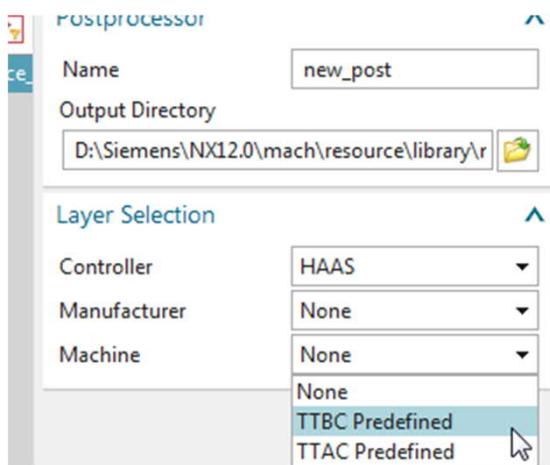
Generic kinematic layer which can be reused to configure predefined kinematic types (HT, HH, TT, 3axis, 2-axis lathe)

Conditions:

- Layer contains all important settings (mom_kin_xxx variables)
- Is integrated in Post create environment in machine section

Task 7

Solution 1/15 – General purpose



Post Configurator use MTB informations. If no model is loaded it will use a 45°-BC Headtable kinematic. To create Postprocessors without MTB but with preconfigured kinematics it's possible to add them during create process. These layers can be reused by different controllers.

Task 7

Solution 2/15 – Additional kinematic layers (TTBC)



1. Create a new file with information about the kinematic, e.g. kinematic type _MinLimit4th_MaxLimit4th_MinLimit5th_MaxLimit5th/nolimits
2. Include this file in the temporary postprocessor behind the mtb layer. When a postprocessor will be created the new one created will be located here automatically
3. Open Post Configurator and select the new created file and open it with the Tcl editor

Task 7

Solution 3/15 – Additional kinematic layers (TTBC)

```

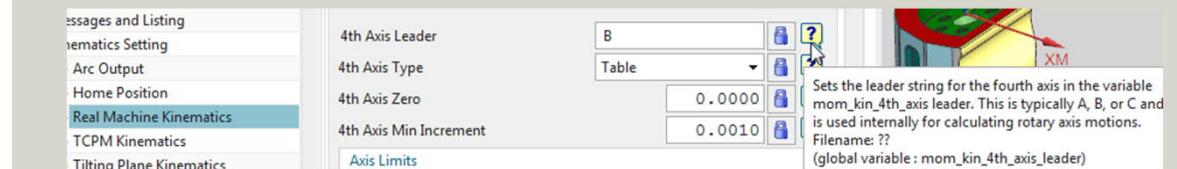
1 ##### Kinematic Layer TTBC #####
2 #
3 # Kinematic Layer TTBC
4 #
5 # Company : Siemens Industry Software
6 # Contact person : Thomas Jenensch
7 # Mail : thomas.jenensch@siemens.com
8 #
9 #####
10 #
11 # Copyright 2018 Siemens Industry Software
12 # All Rights Reserved.
13 #
14 ##### Kinematic type: Table Table #####
15 # Description
16 # This layer contains predefined kinematic condit
17 # kinematic model.
18 #
19 # Kinematic type: Table Table
20 # 4th axis: B
21 # 4th axis Max limit: 110
22 # 4th axis Min limit: -35
23 # 5th axis: C
24 # 5th axis limits: no limits
25 #####
26 # History
27 # 24-07-18 TJ Initial version
28 #####
29 #
30 ### General kinematic settings ###
31 set ::mom_machine_mode MILL
32 set ::mom_kin_machine_type 5_axis_dual_table
33 set mom_sys_leader(X) X
34 set mom_sys_leader(Y) Y
35 set mom_sys_leader(Z) Z
36 set mom_cvc_leader(fourth_axis) n
1 ##### 4th axis settings #####
10 set mom_kin_spindle_axis(2) 1.0
11 ## General kinematic settings ##
12 #
13 ### 4th axis settings #####
14 set mom_kin_4th_axis_leader B
15 set mom_kin_4th_axis_min_incr 0.001
16 set mom_kin_4th_axis_point(0) 0.0
17 set mom_kin_4th_axis_point(1) 0.0
18 set mom_kin_4th_axis_point(2) 0.0
19 set mom_sys_4th_axis_has_limits 1
20 set mom_kin_4th_axis_max_limit 110
21 set mom_kin_4th_axis_soft_max_limit 110
22 set mom_kin_4th_axis_min_limit -35
23 set mom_kin_4th_axis_soft_min_limit -35
24 set mom_kin_4th_axis_type Table
25 set mom_kin_4th_axis_vector(0) 0
26 set mom_kin_4th_axis_vector(1) 1
27 set mom_kin_4th_axis_vector(2) 0
28 set mom_kin_4th_axis_zero 0.0
29 #####
30 ##### 5th axis settings #####
31 set mom_kin_5th_axis_leader C
32 set mom_kin_5th_axis_min_incr 0.001
33 set mom_kin_5th_axis_point(0) 0.0
34 set mom_kin_5th_axis_point(1) 0.0
35 set mom_kin_5th_axis_point(2) 0.0
36 set mom_kin_5th_axis_has_limits 0
37 set mom_kin_5th_axis_max_limit 3600000
38 set mom_kin_5th_axis_soft_max_limit 3600000
39 set mom_kin_5th_axis_min_limit -3600000
40 set mom_kin_5th_axis_soft_min_limit -3600000
41 set mom_kin_5th_axis_type Table
42 set mom_kin_5th_axis_vector(0) 0
43 set mom_kin_5th_axis_vector(1) 0
44 set mom_kin_5th_axis_vector(2) 1
45 set mom_kin_5th_axis_zero 0.0
46 #####

```

1

2

1. Add description of the layer and generic information about the kinematic type
2. Add all necessary mom-variables which are needed for the kinematics. The information which mom-variable is needed is located in the Tooltip help.

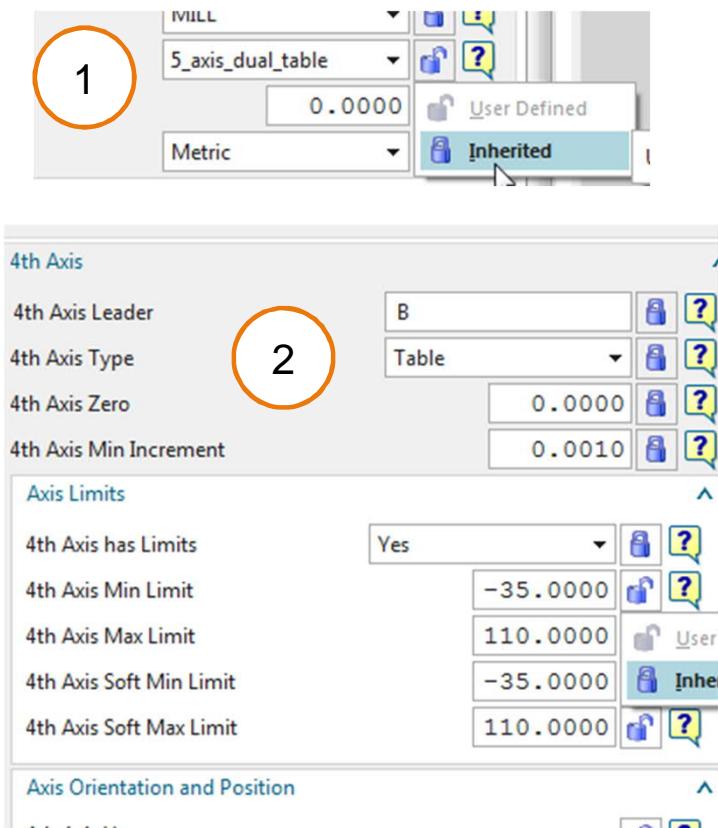


3. After finishing of set up the layer save and close the Tcl-editor.

Task 7

Solution 4/15 – Additional kinematic layers (TTBC)

SIEMENS
Ingenuity for life



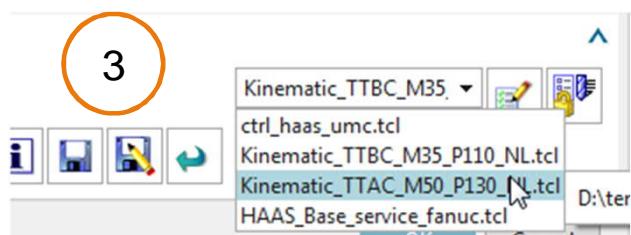
1. Now check that this layer is sourced correctly and all changed properties get now the new default value from the layer for the machine type.
2. Repeat this for all properties which was changed in the UI of Post Configurator (open lock).

Task 7

Solution 5/15 – Additional kinematic layers (TTAC)

HAAS_Base_service.det	18.07.2018 14:47	Export
HAAS_Base_service_fanuc.tcl	18.07.2018 14:47	TCL F
Kinematic_TTAC_M50_P130_NL.tcl	24.07.2018 13:02	TCL F
Kinematic_TTBC_M35_P110_NL.tcl	24.07.2018 13:02	TCL F

```
<?xml version="1.0" encoding="UTF-8"?>
<Configuration>
  <Copyright>Copyright Statement</Copyright>
  <Version>1.0</Version>
  <Controller>Fanuc</Controller>
  <MachineName>HAAS_Base</MachineName>
  <Sourcing>
    <Sequence>
      <Filename Name="ctrl_haas_umc" Processing="true"/>
      <Filename Name="HAAS_Base_mub" Processing="true"/>
      <Filename Name="Kinematic_TTBC_M35_P110_NL" Processing="true"/>
      <Filename Name="Kinematic_TTAC_M50_P130_NL" Processing="true"/>
      <Filename Name="HAAS_Base_service_fanuc" Processing="true"/>
    </Sequence>
  </Sourcing>
</Configuration>
```



Task 7

Solution 6/15 – Additional kinematic layers (TTAC)

```

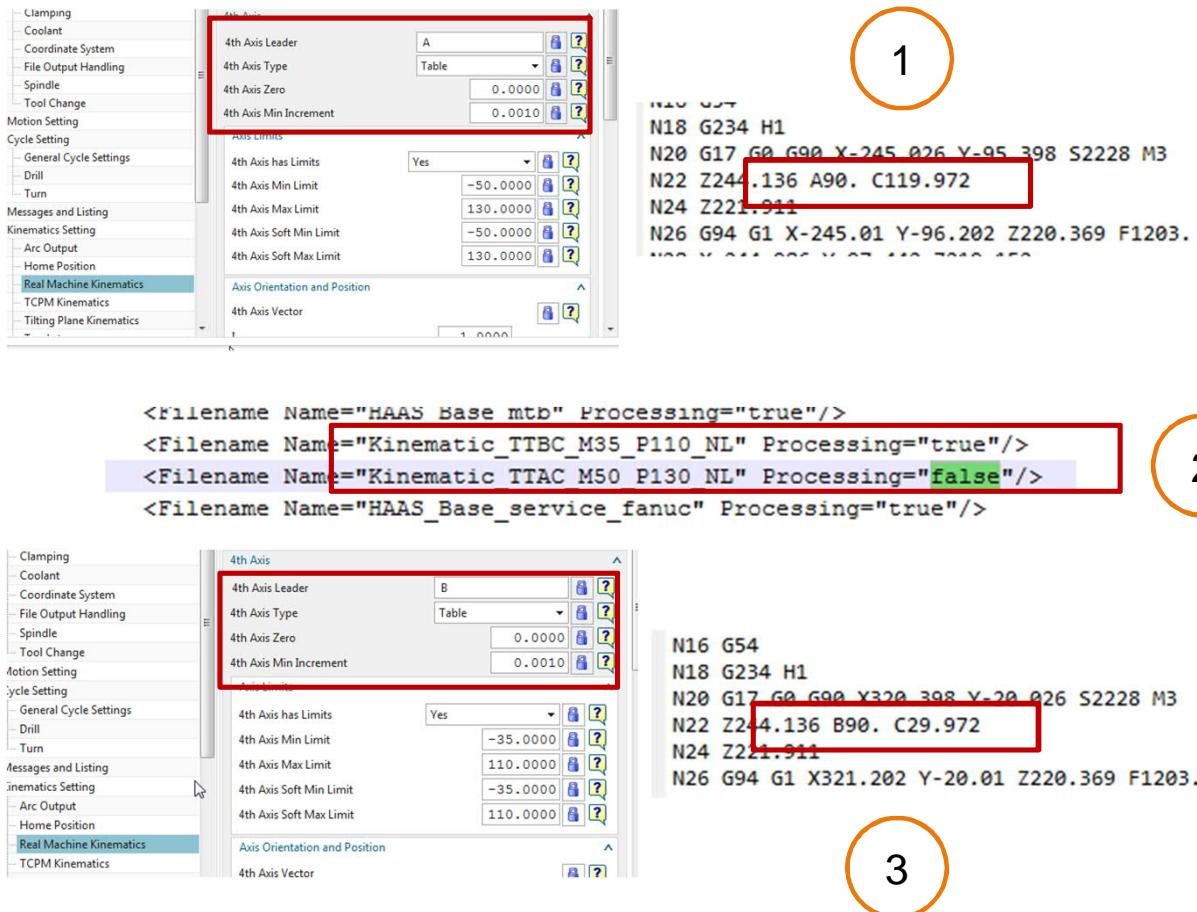
1 ##### Kinematic Layer TTAC #####
2 # Kinematic Layer TTAC
3 #
4 #
5 # Company : Siemens Industry Software
6 # Contact person : Thomas Jenensch
7 # Mail : thomas.jenensch@siemens.com
8 #
9 #####
10 #
11 # Copyright 2018 Siemens Industry Software
12 # All Rights Reserved.
13 #
14 #####
15 # Description
16 # This layer contains predefined kinematic conditions if the post will be created without an
17 # kinematic model.
18 #
19 # Kinematic type: Table Table
20 # 4th axis: A
21 # 4th axis Max limit: 130
22 # 4th axis Min limit: -50
23 # 5th axis: C
24 # 5th axis limits: no limits
25 #####
26 # History
27 # 24-07-18 T3 Initial version
28 #####
29 #
30 ### General kinematic settings #####
31 set ::mom_machine_mode MILL
32 set ::mom_kin_machine_type 5_axis_dual_table
33 set mom_sys_leader(X) X
34 set mom_svs_leader(Y) Y
35 set mom_sys_leader(Z) Z
36 set mom_sys_leader(fourth_axis) A
37 set mom_svs_leader(fifth_axis) C
38 set mom_kin_spindle_axis(0) 0.0
39 set mom_kin_spindle_axis(1) 0.0
40 set mom_kin_spindle_axis(2) 1.0
41 ### General kinematic settings #####
42 #### 4th axis settings #####
43 set mom_kin_4th_axis_leader A
44 set mom_kin_4th_axis_min_incr 0.001
45 set mom_kin_4th_axis_point(0) 0.0
46 set mom_kin_4th_axis_point(1) 0.0
47 set mom_kin_4th_axis_point(2) 0.0
48 set mom_sys_4th_axis_has_limits 1
49 set mom_kin_4th_axis_max_limit 130
50 set mom_kin_4th_axis_soft_max_limit 130
51 set mom_kin_4th_axis_min_limit -50
52 set mom_kin_4th_axis_soft_min_limit -50
53 set mom_kin_4th_axis_type Table
54 set mom_kin_4th_axis_vector(0) 1
55 set mom_kin_4th_axis_vector(1) 0
56 set mom_kin_4th_axis_vector(2) 0
57 set mom_kin_4th_axis_zero 0.0
58 #### 4th axis settings #####
59 #####
60

```

1. Add additional information in header
2. Change general settings, e.g. leader
3. Change all needed mom-variables for 4th axis (leader, vector, limits, type)

Task 7

Solution 7/15 – Additional kinematic layers (TTAC)



Summary:

Now it's possible to switch the kinematics easily.

1. Postprocess an operation with TTAC kinematics file
2. In the psc-file turn off the processing of TTAC kinematics file
3. Reopen Post Configurator and postprocess again

Additional Background:

This can be reused for each predefined kinematic definition. In next topic this will be transferred into standard environment.

Task 7

Solution 8/15 – Create structure in environment



1. Create a Backup of your origin post_registry.xml
2. Create a new folder for the new created layers
3. In this folder create a sub structure, e.g. Controller, machine
4. Copy the created controller file from the temporary postprocessor into the folder

Task 7

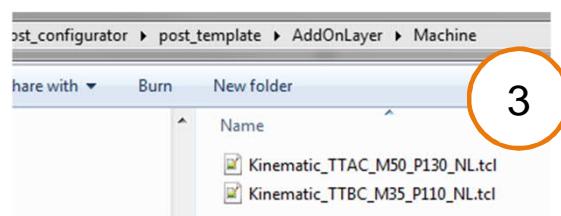
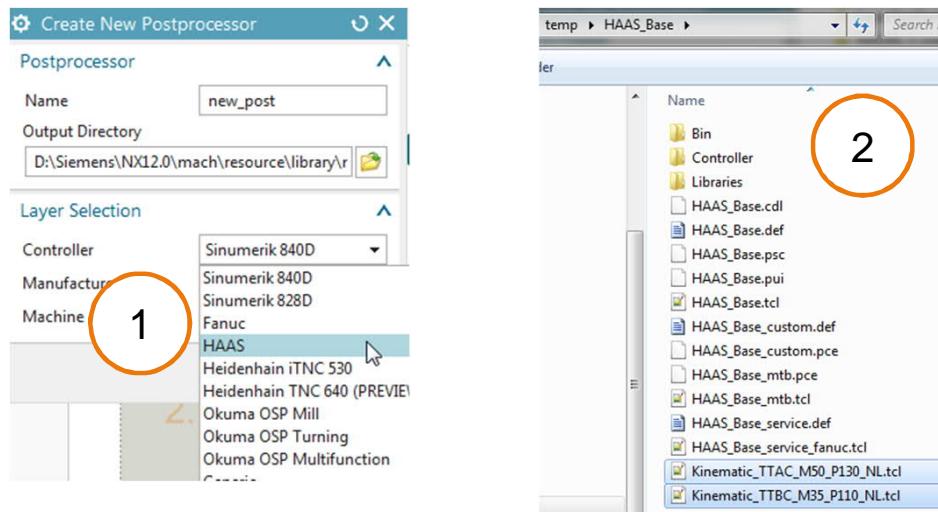
Solution 9/15 – Create structure

1. Open the post_registry.xml
2. Copy the existing Fanuc entry
3. Rename the copy to e.g. HAAS
4. Add the entry for the main-tcl file which will also source the Fanuc controller
5. Save the file

```
</CTRL>
<CTRL>
  <Name>Fanuc</Name>
  <Path>${UGII_CAM_RESOURCE_DIR}post_configurator/post_template/controller/fanuc/base/ctrl_fanuc_base.cdl</Path>
  <Path>${UGII_CAM_RESOURCE_DIR}post_configurator/post_template/controller/fanuc/base/ctrl_fanuc_base.def</Path>
  <Path>${UGII_CAM_RESOURCE_DIR}post_configurator/post_template/controller/fanuc/base/ctrl_fanuc_base.pce</Path>
  <Path>${UGII_CAM_RESOURCE_DIR}post_configurator/post_template/controller/fanuc/base/ctrl_fanuc_base_msg.pce</Path>
  <Service_Path>${UGII_CAM_RESOURCE_DIR}post_configurator/post_template/controller/fanuc/service/template_service_fanuc.pce</Service_Path>
</CTRL>
<CTRL>
  <Name>HAAS</Name>
  <Path>${UGII_CAM_RESOURCE_DIR}post_configurator/post_template/AddOnLayer/Controller/ctrl_haas_umc.tcl</Path>
  <Path>${UGII_CAM_RESOURCE_DIR}post_configurator/post_template/controller/fanuc/base/ctrl_fanuc_base.cdl</Path>
  <Path>${UGII_CAM_RESOURCE_DIR}post_configurator/post_template/controller/fanuc/base/ctrl_fanuc_base.def</Path>
  <Path>${UGII_CAM_RESOURCE_DIR}post_configurator/post_template/controller/fanuc/base/ctrl_fanuc_base.pce</Path>
  <Path>${UGII_CAM_RESOURCE_DIR}post_configurator/post_template/controller/fanuc/base/ctrl_fanuc_base_msg.pce</Path>
  <Service_Path>${UGII_CAM_RESOURCE_DIR}post_configurator/post_template/controller/fanuc/service/template_service_fanuc.pce</Service_Path>
</CTRL>
<CTRL>
```

Task 7

Solution 10/15 – Create structure



1. Open Post Configurator and check that post_registry works fine and the added controller level is visible
2. Add the additional two kinematics files into the AddonLayer/Machine folder from the temporary postprocessor
3. Open the post_registry.xml to add these files as machine layers for this HAAS controller

Task 7

Solution 11/15 – Create structure



1. Copy an existing machine level, e.g. Fanuc
2. Give it a unique name and assign the Base_CTRL (logic which files will be shown when select a controller)
3. Repeat the step for the second predefined kinematic and save the post_registry.xml

The screenshot shows a portion of a XML configuration file for post-processing registries. The file is structured as follows:

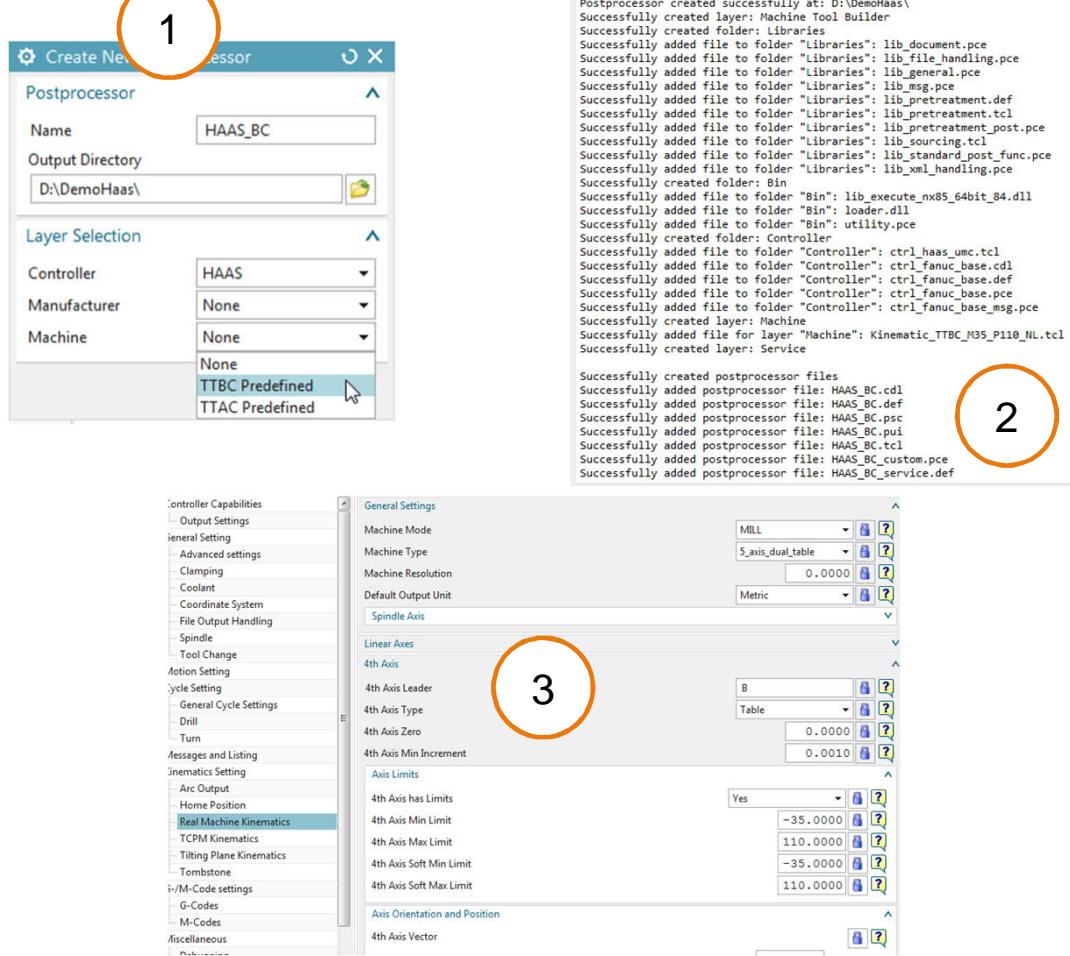
```
<MACHINE>
  <Name>Fanuc Sample Machine Level</Name>
  <Path>${UGII_CAM_RESOURCE_DIR}post_configurator/post_template/controller/fanuc/machine/machine_ootb_5ax_fanuc.pce</Path>
  <Base_CTRL>Fanuc</Base_CTRL>
</MACHINE>
<MACHINE>
  <Name>TTBC Predefined</Name>
  <Path>${UGII_CAM_RESOURCE_DIR}post_configurator/post_template/AddOnLayer/Machine/Kinematic_TTBC_M35_P110_NL.tcl</Path>
  <Base_CTRL>HAAS</Base_CTRL>
</MACHINE>
<MACHINE>
  <Name>TTAC Predefined</Name>
  <Path>${UGII_CAM_RESOURCE_DIR}post_configurator/post_template/AddOnLayer/Machine/Kinematic_TTAC_M50_P130_NL.tcl</Path>
  <Base_CTRL>HAAS</Base_CTRL>
</MACHINE>
```

Three specific sections are highlighted with orange circles and numbered 1, 2, and 3:

1. The first machine entry, which includes the path to the Fanuc template and the assigned base controller.
2. The second machine entry, which includes the path to the TTBC template and the assigned base controller. This entry is enclosed in a red box.
3. The third machine entry, which includes the path to the TTAC template and the assigned base controller. This entry is highlighted with a purple background.

Task 7

Solution 12/15 – Create structure



Unrestricted © Siemens AG 2020

SIEMENS
Ingenuity for life

1. Check that everything is fine and open the Create Postprocessor Dialog
2. Create a new Postprocessor based on this data
3. Check/ recognize that kinematics are set directly correct

Additional Background:

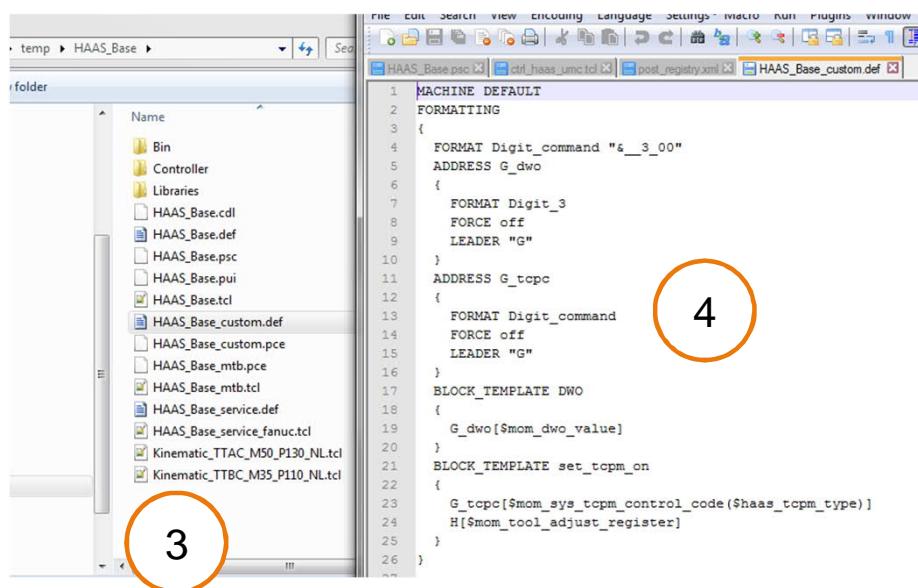
Even if a machine model is loaded now, the predefined kinematics will be used. For this you can build a property which decides use machine model kinematics or not.

Siemens PLM Software

Task 7

Solution 13/15 – Create structure

```
***** MOM: TCL SCRIPT MESSAGE *****
User message: Error code 1745006: MOM given an invalid block name; MOM: DWO is an invalid block template name.
while executing
"MOM_do_template DWO"
invoked from within
"if {1} [subst ${\${subst ::buffer::${00_ section}}::code($00_tag)}\}]"
("foreach" body line 8)
```



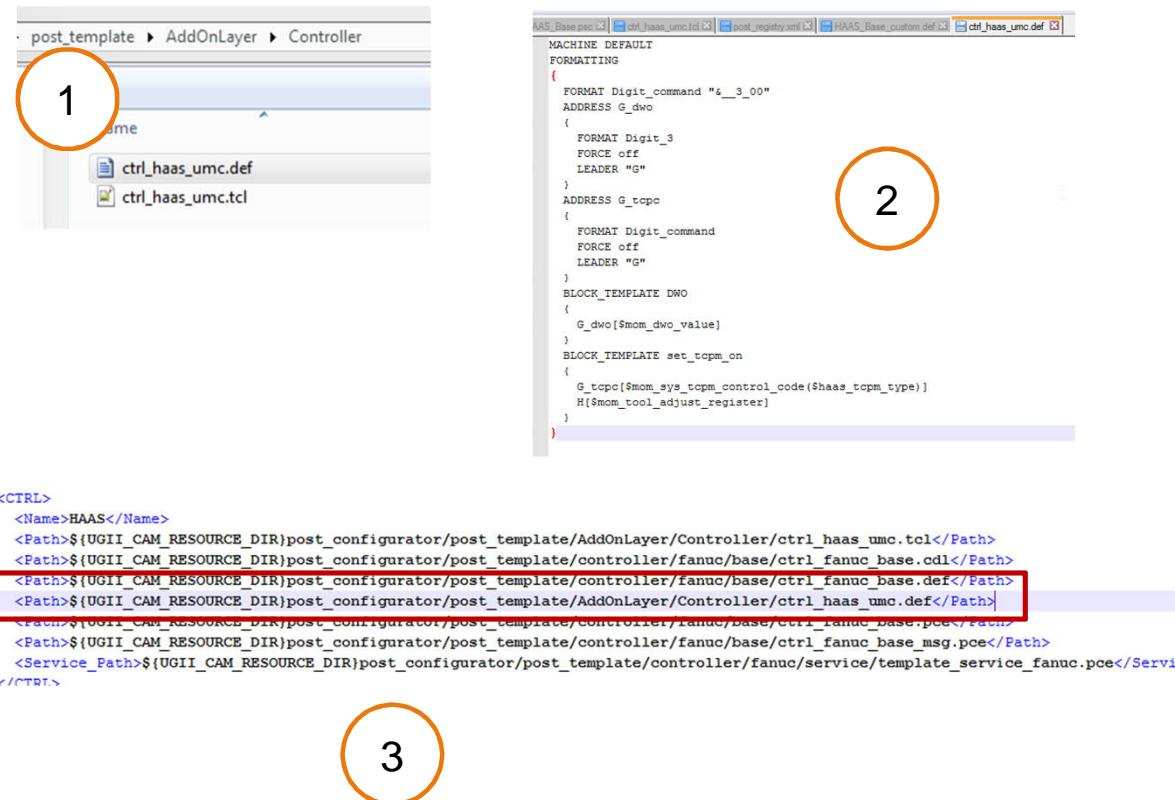
1. Select an operation and try postprocessing. You should get an error 😊.
2. Thinking...
3. Remember, we defined additional Blocktemplates and address.
4. Open the custom.def-file from the temporary postprocessor.

Additional Background:

All customization for address, Blocktemplates, Formats will be written in the xx_custom.def file of the postprocessor. To reuse it we have to copy the content in a def-file controller layer

Task 7

Solution 14/15 – Create structure



1. Create a new def-file in the AddOnLayer/Controller folder
2. Copy the content from temporary postprocessor/ custom def-file into that file
3. Link this file to the controller. Be aware of sourcing order if you overwritten Blocktemplates then the origin layer must be sourced before.

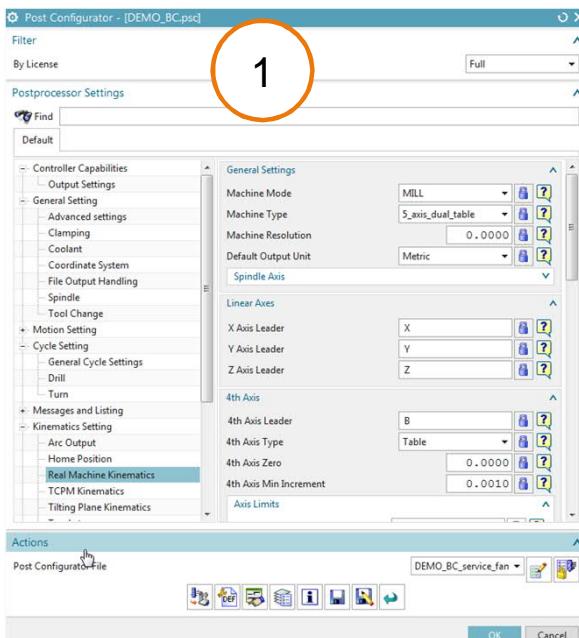
Additional Background:

The same is possible for cdl-files which contains UDE's.

Task 7

Solution 15/15 – Create structure

SIEMENS
Ingenuity for life



The screenshot shows a G-code editor window with a circled '2'. The G-code listed is:

```
(DATE : 24.07.2010 , 14:40  
(PARTNAME : ACCEPTANCE_PART.PRT  
N10 G17 G21 G94 G90  
  
(VARIABLE_STREAMLINE , TOOL : UGTLI0203_052)  
  
N12 G0 G53 Z99999.9  
N14 T00 M6  
N16 G54  
N18 G234 H1  
N20 G17 G0 G90 X320.398 Y-20.026 S2228 M3  
N22 Z244.136 B90. C29.972  
N24 Z221.911  
N26 G94 G1 X321.202 Y-20.01 Z220.369 F1203.  
N28 X322.442 Y-19.986 Z219.152  
N30 X323.998 Y-19.956 Z218.378  
N32 X325.718 Y-19.923 Z218.122  
N34 X327.432 Y-19.89 Z218.411  
N36 X327.166 Y-19.894 Z215.539 B89.442 C29.961  
N38 X326.869 Y-19.9 Z212.597 B88.88 C29.948  
N40 X326.532 Y-19.906 Z209.499 B88.299 C29.936  
N42 X326.15 Y-19.911 Z206.23 B87.698 C29.927  
N44 X325.724 Y-19.913 Z202.827 B87.087 C29.922
```

1. Create a new postprocessor again
2. After creation postprocess an operation.
3. Done.

Q&A



Thomas Jenensch

Product Portfolio Lead NX CAM Infrastructure
Manufacturing Engineering Software

Nonnendammallee 101 5. OG, Bauteil C
D-13629 Berlin, Germany
Tel. :+49 (30) 46777 535

thomas.jenensch@siemens.com
www.siemens.com/plm

Siemens Manufacturing Forum
www.siemens.com/plm/nxmanufacturingforum

Realize Innovation