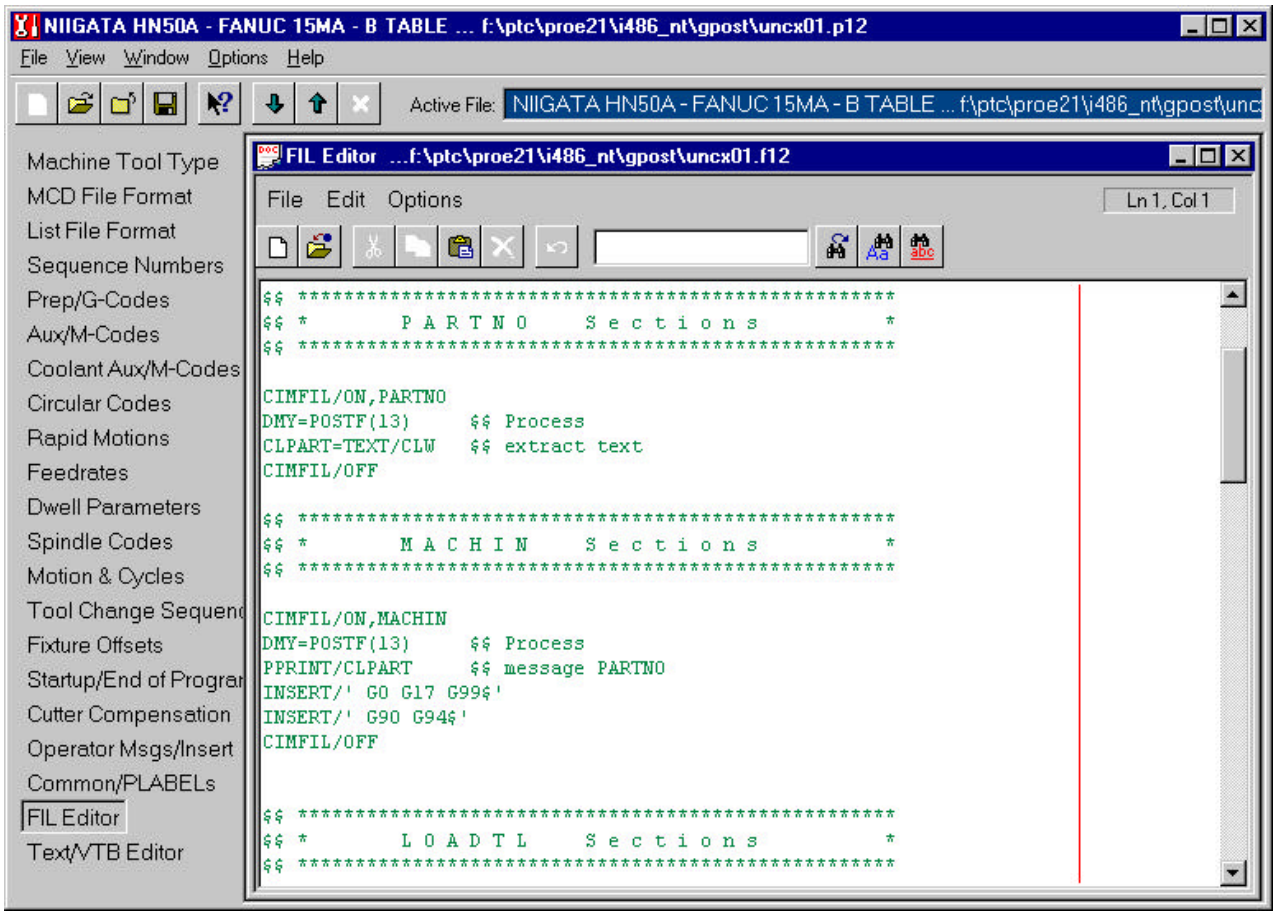
Pro/NC-GPOST

FIL Exercises



Exercise 1

Basic CIMFIL Macros

This exercise presents some of the basic CIMFIL macros.

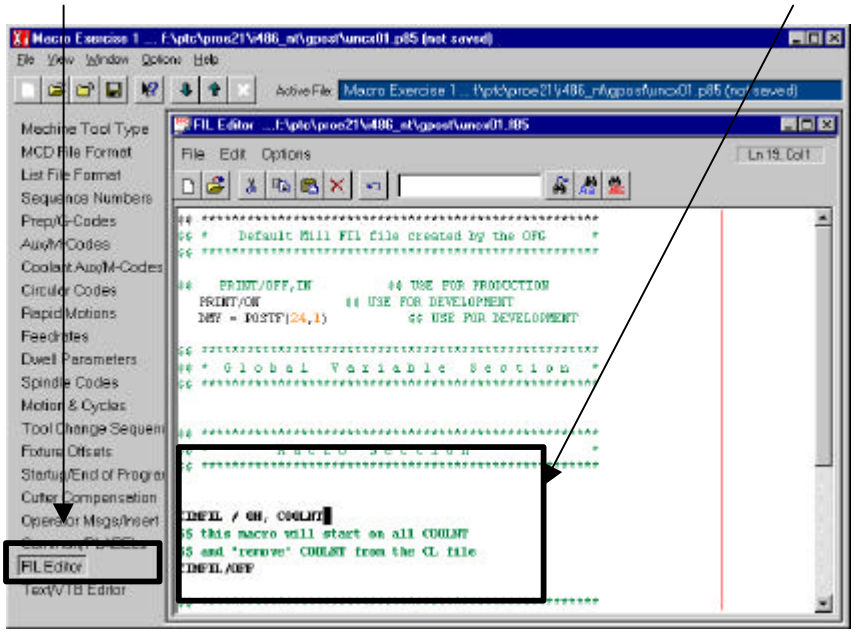
Exercise Steps:

1. Create a generic post-processor for a 3-axes mill with a *Mitsubishi Meldas 520 control,* using the control defaults *Mitsubishi Meldas 500M control*. Save the new post as *UNCX01.P11* and use it to post-process the CL file *pplab1.ncl* from the Pro/NC

2. The first step is to make a macro to ignore all coolant commands. This demonstrates the most basic CIMFIL application : ignore a Post-processor command :

   CIMFIL / ON, COOLNT
   $$ this macro will start on all COOLNT
   $$ and "remove" COOLNT from the CL file
   CIMFIL/OFF

   To add this macro to the post-processorUNCX01.P11, you need to select the **FIL Editor** category and type the text of the macro in the **FIL Editor** window of Optfile :



   Save the post and post-process pplab1.ncl to see the result (coolant code M08 has disappeared)

3. In fact the machine supports coolant but it needs to be started by the operator. Let's modify the COOLNT CIMFIL macro to issue a message to the operator. This demonstrates another type of CIMFIL macro : replace a CL command by another.

In this case the COOLNT macro is going to be :

```
CIMFIL / ON, COOLNT
$$ This macro will start on all COOLNT
ARG1=POSTF(7,4)              $$ first argument of COOLNT
IF (ARG1.EQ.ICODEF(OFF)) THEN
$$ This is COOLNT/OFF
  INSERT/'(OPERATOR PLEASE STOP COOLNT)$'   $$ Message
ELSE
$$ This is ON or FLOOD or MIST, …
  INSERT/'(OPERATOR PLEASE START COOLANT)$' $$ Message
ENDIF
DELAY / 20                   $$ 20 sec. To give time to the operator
CIMFIL / OFF
```

Save the post and post-process pplab1.ncl to see the result :

```
%
O1111
N0010 G20
N0020 T01 M06
N0030 S600 M03
N0040(OPERATOR PLEASE START COOLANT)
N0050 G04 X20.
N0060 G00 X-2.5 Y0.
…
```

4. Now let's make a macro to start or stop the AIR (M12, M13) before the processing of SPINDL. This demonstrates another type of CIMFIL macro : add a post-processor command after an existing command.

```
CIMFIL / ON, SPINDL          $$ Starts on all SPINDL statement
DMY=POSTF(20)                $$ save the SPINDL record in memory
PARAM=POSTF(7,4)             $$ first parameter of SPINDL command
IF (PARAM.EQ.ICODEF(OFF)) THEN
$$ it is SPINDL/OFF, stop air first
  AUXFUN/13                  $$ stop AIR
ELSE
$$ it is SPINDL/ON, or SPINDL/RPM,… start air
  AUXFUN/12                  $$ start AIR
ENDIF
DMY=POSTF(21)                $$ restore the SPINDL record saved in memory
DMY=POSTF(13)                $$ Process the CL record
CIMFIL/OFF                   $$ End of CIMFIL on SPINDL
```

5. Now we want to add some NC codes at the end of the NC program :

```
G91 G28 Z0.
G28 X0. Y0.
M30
```

To do this we are going to make a macro on FINI :

```
CIMFIL / ON, FINI
DMY=POSTF(20)              $$ save the FINI record in memory
INSERT/' G91 G28 Z0.$'
INSERT/' G28 X0. Y0.$'
INSERT/' M30$'
DMY=POSTF(21)              $$ restore the FINI record saved in memory
DMY=POSTF(13)              $$ Process the CL record
CIMFIL/OFF                 $$ End of CIMFIL on FINI
```

Do not forget to add REDEF/ON in the global section of the FIL file to allow redefinition of DMY (as a test you can see what happens if you don't do this).

1. Now, finally, we need to add some safe startup codes at the beginning of the program :
   G90 G80 G40
   (DATE : 12/12/1999 03:20)

   To do this we need to make a macro on MACHIN. It is important to note that tape output is generated **ONLY after the MACHIN statement is processed**.

```
CIMFIL / ON, MACHIN    $$ starts on all MACHIN
DMY=POSTF(13)          $$ process MACHIN
INSERT / ' G90 G80 G40$'       $$ output safe startup block
DVAL = TEXT / TIMES    $$ Extract the system time
INSERT/'( DATE :',DVAL,')$'    $$ insert the message.
CIMFIL / OFF           $$ End of macro on MACHIN
```

Additional tasks

Macro a CIMFIL macro to replace all RAPID by FEDRAT/200,IPM
Modify the SPINDL CIMFIL macro to output the AIR command after the processing of SPINDL.
Remove all the PPRINT with the word COMMENTS in it (see TEXT/CLW, INDXF function)

Exercise 2

Post-processor Macro Variables : Real

This exercise allows a better familiarization with real variables available in the post-processor macro language. The object is to customize a post-processor in order to accommodate a special output requirement.

Exercise Steps:

1. Create a generic post-processor for a 3-axes mill with a *Mitsubishi Meldas 520 control,* using the control defaults *Mitsubishi Meldas 500M control*. Save the new post as *UNCX01.P12* and use it to post-process the CL file *pplab1.ncl* from the Pro/NC manufacturing model pplab1.mfg. Analyze the results (*pplab1.tap*).

The customization requirement for this exercise consists in accommodating a special sequence numbering convention: each NC Sequence must restart block numbering using *multiples of 1000,* continuing in the respective range with an increment of 1 (i.e. N1000, N1001, N1002… for the 1st NC Sequence, N2000, N2001, N2002… for the 2nd one, etc).

---

2. The first step is to create a variable in the global section of FIL for the sequence number multiplier : SEQNB

```
$$ Variable Definition and initialization
SEQNB = 0
```

3. Now we need to create a CIMFIL macro on LOADTL :

```
CIMFIL / ON, LOADTL     $$ starts on all LOADTL
DMY=POSTF(20)           $$ saves current CL record
SEQNO / OFF             $$ Stops sequence numbering
INSERT / ' $'           $$ inserts a blank line
$$ Increment variable SEQNB (NC Sequence count)
SEQNB = SEQNB + 1
$$ Calculate new start value for sequence numbers
STVAL = SEQNB * 1000
$$ Restart sequence numbering using STVAL
SEQNO / STVAL, INCR, 1
SEQNO / ON              $$ restarts sequence numbering
DMY=POSTF(21)           $$ reloads LOADTL record saved
DMY=POSTF(13)           $$ Process LOADTL
CIMFIL/OFF              $$ End of CIMFIL macro on LOADTL
```

4. Save the post and post-process pplab1.ncl to see the result.

Exercise 3

Post-processor Macro Variables : String

This exercise allows a better familiarization with string variables available in the post-processor macro language. The object is to customize a post-processor in order to accommodate a special tape header.

Exercise Steps:

2. Create a generic post-processor for a 3-axes mill with a *FANUC 6M,* using the control defaults *FANUC 6M Control*. Save the new post as *UNCX01.P13* and use it to post-process the CL file *pplab1.ncl* from the Pro/NC manufacturing model pplab1.mfg. Analyze the results (*pplab1.tap*).

The customization requirement for this exercise consists in generating a special header at the beginning of the program. The PARTNO needs to be displayed as message to the operator at the beginning of the program. A safe startup block : G90 G80 G40' needs also to be inserted. The feedrate needs to be defaulted to 10 IPM.

---

3. The first step is to create CIMFIL macro on PARTNO to capture the value of PARTNO :

```
CIMFIL/ON,PARTNO      $$ starts on all PARTNO
DMY = POSTF(13)       $$ process the PARTNO
PTXT = TEXT/CLW       $$ get the text of the PARTNO
PTXT1 = TEXT/OMIT,PTXT,1    $$ Remove trailing blank
CIMFIL / OFF          $$ End of macro on PARTNO
```

4. Now we need to create a CIMFIL macro on MACHIN. Tape output is generated **ONLY after the MACHIN statement is processed**.

```
CIMFIL / ON, MACHIN    $$ starts on all MACHIN
DMY=POSTF(13)          $$ process MACHIN
INSERT / '(',PTXT1,')$'   $$ inserts the PARTNO as a message
INSERT / 'G90 G80 G40$'       $$ output safe startup block
FEDRAT / 10, IPM       $$ default feedrate
CIMFIL / OFF           $$ End of macro on MACHIN
```

4. Save the post and post-process pplab1.ncl to see the result.

Additional tasks

Simplify the PARTNO macro (do not use two text variables : see REDEF command)
Remove the / in the message (see REPLAC command)
Output the CL file name to the tape (see TEXT command)
Output the Date and Time to the tape (see TEXT command)

Exercise 4

Support of sub-program

The objective of this exercise is to customize a post-processor to support sub-program.

Exercise Steps:

1. Create a generic post-processor for a 3-axes mill with a *FANUC 10M,* using the control defaults *FANUC 10M Control*. Save the new post as *UNCX01.P13* and use it to post-process the CL file *pplab14.ncl*.
   Analyze the results (*pplab14.tap*).

2. The first CIMFIL macro to write is on DEFSUB : sub program definition. For FANUC the sub-program definition is done by **O*sub_number***

```
CIMFIL / ON, DEFSUB          $$ Starts on all DEFSUB
SUBNUM=POSTF(7,4)            $$ First parameter of the DEFSUB command
$$ Convert the real SUBNUM to a string O followed by 4 digits
TPNUM=TEXT/'O',CONVF,SUBNUM,4,0,0,1,1,'$'
SEQNO/OFF                    $$ Stop sequence numbering
INSERT/TPNUM                 $$ Insert sub-program number
SEQNO/10,INCR,1              $$ restart sequence numbering
CIMFIL/OFF                   $$ End of macro on DEFSUB
```

3. Next we need to make the macro for the end of the sub-program : ENDSUB. For FANUC the end of the sub-program is defined by M99.

```
CIMFIL / ON, ENDSUB          $$ Starts on all ENDSUB
INSERT / 'M99$'              $$ Inserts end of sub-program
SEQNO / OFF                  $$ Stops sequence numbering
INSERT / '  $'               $$ Inserts blank line
SEQNO/ON                     $$ restarts sequence numbering
CIMFIL / OFF                 $$ End of macro on ENDSUB
```

4. Finally we need to make a macro for the sub-program call : CALSUB. For FANUC the sub-program call is defined by **M98P*sub_number***

```
CIMFIL / ON, CALSUB          $$ Starts on all CALSUB
CNUM=POSTF(7,4)              $$ First parameter of the CALSUB command
$$ Converts the real CNUM to a string M98P followed by program number
TPNUM=TEXT/'M98P',CONVF,CNUM,4,0,0,1,3,'$'
INSERT / TPNUM               $$ Sub-program call
CIMFIL / OFF                 $$ End of macro on CALSUB
```

5. For cosmetic reason we need to detect the first tool change to issue the header for the main program.
   First we need to setup a flag in the global area of the FIL file :

$$ Flag to detect the first tool change
FTOOL = 0

6. Then we need to make a CIMFIL macro on LOADTL :

```
CIMFIL / ON, LOADTL          $$ Starts on all LOADTL
DMY = POSTF(20)              $$ Saves current CL record
IF (FTOOL .EQ. 0) THEN       $$ Checks if first tool change
   FTOOL = 1                 $$ resets the flag
   SEQNO/OFF                 $$ Stops sequence numbering
   INSERT/'  $'              $$ Inserts a blank line
   INSERT/'%$'               $$ inserts a %'
   SEQNO/10,INCR,2           $$ Restarts sequence numbering
   INSERT/'G90G40G80$'       $$ Safe startup
   DATTXT=TEXT/TIMES         $$ Current date and time
   INSERT/'(',DATTXT,')$'    $$ Message to operator
ENDIF
DMY=POSTF(21)                $$ Restores saved record (LOADTL)
DMY=POSTF(13)                $$ Process LOADTL
CIMFIL / OFF                 $$ End of CIMFIL on LOADTL
```

7. Save the post and post-process pplab14.ncl to see the result.
   Many Errors are generated :

```
** Error ** 1007 SYMBOL IS DOUBLY DEFINED--TPNUM
```

We are redefining TPNUM at each DEFSUB, we need to add **REDEF/ON** in the global section of the FIL File.
In general it is always safer to add REDEF/ON in all your FIL files.

Save the post and post-process pplab?.ncl to see the result.

Additional tasks

Clean up the NC tape output using FIL macros or questionnaire modifications:

- Add space between registers where necessary (check the tape output)
- Remove the G20 at the beginning of the tape
- Put the tool name after the tool change (CIMFIL on PPRINT)
- Be sure to start the sub-program with G00 or G01 (reset feedrate mode : POSTF, INTCOM)
- Ignore SPINDL/OFF

Exercise 5

FIL File Optimization

This exercise will demonstrate technique to simplify macro development using macros, ALIAS and INCLUD statements.

Exercise Steps:

1. Create a post-processor for a Makino, using the control defaults *FANUC 16M Control*. Save the new post as *UNCX01.P15* and use it to post-process the CL file *pplab1.ncl* from the Pro/NC manufacturing model pplab1.mfg. Analyze the results (*pplab1.tap*).

2. The first thing to do in the FIL macro is to define some alias for the most common used functions :

```
$$ ALIAS definition
ALIAS / %SAVE_CLREC, DMY=POSTF(20)       $$ Save current CL record
ALIAS / %RESTORE_CLREC, DMY=POSTF(21) $$ Restore saved CL record
ALIAS / %PROCESS_CLREC, DMY=POSTF(13) $$ process current CL record in memory
ALIAS / %OUTPUT, DMY=POSTF(13)              $$ process current  CL record in memory
ALIAS / %FCL1, POSTF(7,4)                   $$ first argument of current CL record
ALIAS / %FCL2, POSTF(7,5)                   $$ second argument of current CL record
ALIAS / %FCL3, POSTF(7,6)                   $$ third argument of current CL record
ALIAS / %FCL4, POSTF(7,7)                   $$ fourth argument of current CL record
ALIAS / %FCL5, POSTF(7,4)                   $$ fifth argument of current CL record
ALIAS / %FCL6, POSTF(7,4)                   $$ sixth argument of current CL record
ALIAS / %FCL7, POSTF(7,4)                   $$ seventh argument of current CL record
ALIAS / %FCL8, POSTF(7,4)                   $$ eigth argument of current CL record
ALIAS / %FIZE, POSTF(5)                     $$ number of argument of current CL record
ALIAS / %GET_TXT, TEXT/CLW                  $$ return argument text of current CL record
ALIAS/%MTNMAC_TRUE,DMY=POSTF(26,5,5,1)      $$ Macro on motion activated
ALIAS/%MTNMAC_FALSE,DMY=POSTF(26,5,5,0)     $$ Macro on motion de-activated
```

3. We can also define some standard variables used in FIL files, like for example all the letter addresses :

```
A=1;B=2;C=3;D=4;E=5;F=6;G=7;H=8;I=9;J=10;K=11;L=12;M=13
N=14;O=15;P=16;Q=17;R=18;S=19;T=20;U=21;V=22;W=23;X=24;Y=25;Z=26
```

4. Let's make a macro to write a machining summary (program size and machining time) at the end of the program. This macro can be used in any post-processor :

```
ENDSUM = MACRO
_TTIME=POSTF(1,3,495)       $$ DBLCOM 495 : total machining time
_TCHAR=POSTF(1,3,494)       $$ DBLCOM 494 : total tape length
PPRINT / 'CYCLE TIME :',_TTIME ,'sec.  BYTES:',_TCHAR
TERMAC
```

   Now to print the machining time at the end of the program we need to make a CIMFIL macro on FINI. We are also going to use the ALIAS defined previously :

```
CIMFIL / ON, FINI          $$ Starts on FINI
%SAVE_CLREC                $$ Save current CL rec.
CALL/ENDSUM                $$ Print time and bytes summary
%RESTORE_CLREC             $$ Restore CL rec. in memory
%PROCESS_CLREC             $$ Process current CL rec.
CIMFIL/OFF                 $$ End of macro on FINI
```

5. Now let's issue a M98P1000 (super GI command for Makino) before the first move after LOADTL. To do this we are going to do a CIMFIL macro on GOTO.
   The first thing is to activate this macro in a CIMFIL on LOADTL :

```
CIMFIL / ON, LOADTL        $$ Starts on all LOADTL
%PROCESS_CLREC             $$ Process LOADTL
%MTNMAC_TRUE               $$ Activate macro on GOTO
CIMFIL/OFF                 $$ End of CIMFIL on LOADTL
```

   Then we can do the CIMFIL macro on GOTO

```
CIMFIL/ON,GOTO             $$ Starts on all GOTO
%SAVE_CLREC                $$ Save the GOTO
INSERT/'M98P1000$'         $$ issue the super GI command
%MTNMAC_FALSE              $$ De-activate macro on GOTO
%RESTORE_CLREC             $$ Restore the saved CL record (GOTO)
%PROCESS_CLREC             $$ Process the GOTO
CIMFIL/OFF                 $$ End of macro on GOTO
```

6. All these ALIAS, variables initialization and macros can be written in a file like **myfil.txt** that you can include in all your post-processors by using :

   INCLUD / myfil.txt

   Another solution is to create your own default startup FIL files **: UNCX01.F00, UNCL01.F00, UNCP01.F00, UNCC01.F00** and **UNCW01.F00** in the ***$PRO_DIRECTORY/$MC/*obj** (for example C:\PTC\PROE2000I\i486_NT\OBJ)

Additional tasks

Rewrite the macro developed in the previous exercises using the ALIAS.

Exercise 6

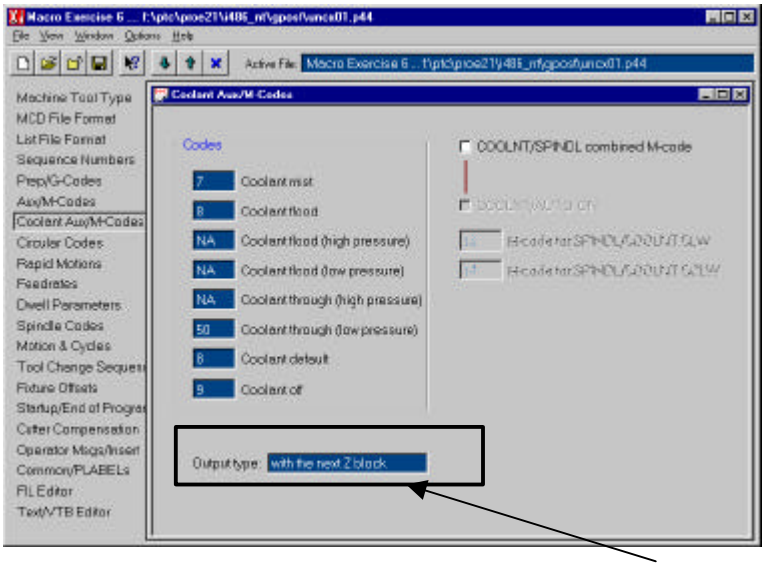
Internal Variables (COMMON)

This exercise will demonstrate techniques to read and modify internal post-processor variables (common) during post-processing execution.

Exercise Steps:

1. Create a post-processor for a Toshiba milling machine, using the control defaults *Toshiba TOSNUC 800 Serie Control*. Save the new post as *UNCX01.P16* and use it to post-process the CL file *pplab1.ncl* from the Pro/NC manufacturing model pplab1.mfg. Analyze the results (*pplab1.tap).*

   If you compare the CL file (pplab1.ncl) and the tape file, you can see that there is no COOLNT/OFF M code (M9), even if COOLNT/OFF is programmed in Pro/NC and the code M9 is defined in the **Coolnt Aux/M-Codes** category of UNCX01.P16 post-processor.

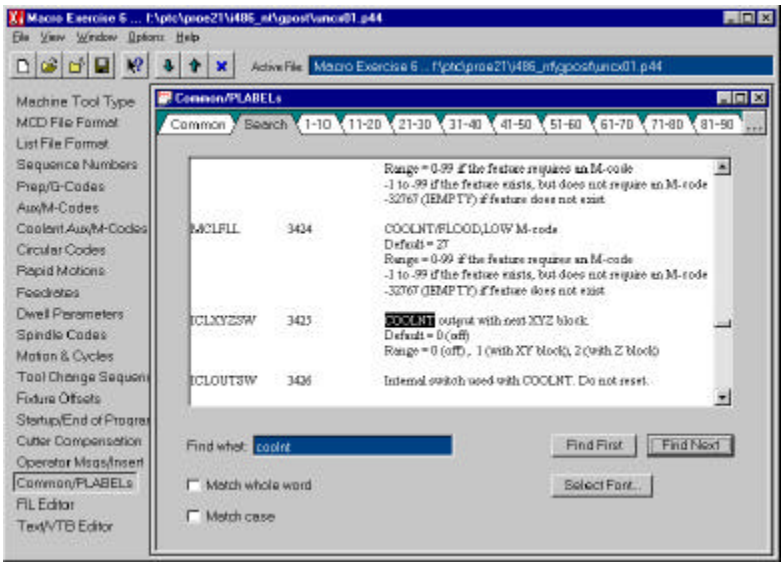


   The processing of the coolant is delayed to the next Z motion, but COOLNT/OFF is not processed with the next Z motion because a COOLNT/ON is encountered by the post-processor before the next motion :

```
GOTO / 9.7500000000, 5.2500000000, -0.2500000000
GOTO / 9.7500000000, 5.2500000000, 0.5000000000
COOLNT / OFF        <- COOLNT programmed wait for next Z move to process
SPINDL / OFF
$$-> END /
$$-> FEATNO / 73
LOADTL / 2 $$-> 1" F.E.M.
$$-> CUTTER / 1.000000
SPINDL / RPM, 600.000000,  CLW
COOLNT / ON         <- COOLNT programmed wait for next Z move to process
RAPID
```

```
GOTO / 9.6250000000, 3.6250, 0.500 <- Z move, process last COOLNT
programmed
```

2. Let's make a CIMFIL macro to process COOLNT/OFF on a line by himself and continue to process the other COOLNT commands with the next Z move.
   The first step is to find the internal common for the COOLNT **Output Type** question.
   To find which internal variable to use we are going to select the **Common/PLABELs** category, sub-panel Search.
   We are looking for the word **COOLNT** :



   The variable is **INTCOM 3425**
   We can now write a CIMFIL macro on COOLNT :

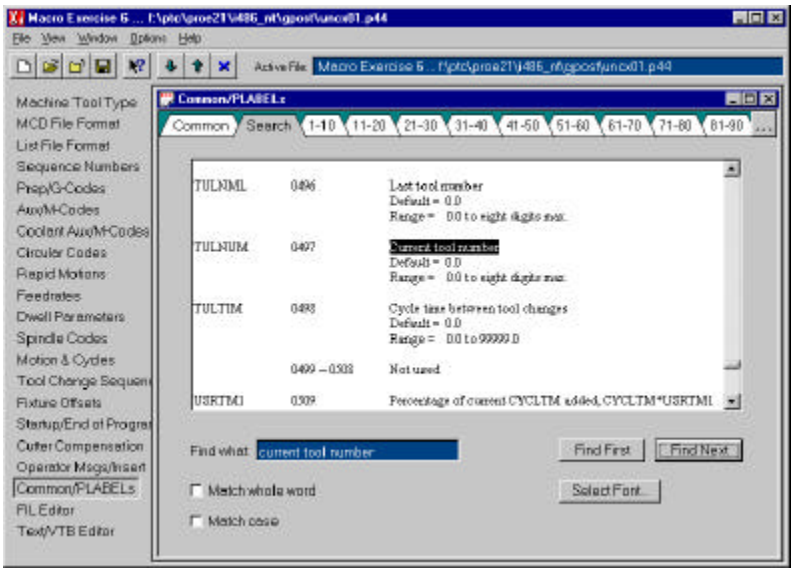
```
CIMFIL / ON, COOLNT             $$ Starts on all COOLNT
COLTYP=POSTF(7,4)               $$ First argument of COOLNT
IF (COLTYP.EQ.ICODEF(OFF)) THEN
$$ This is COOLNT/OFF
  DMY=POSTF(2,1,3425,0)         $$ Set INTCOM 3425 to 0 (output now)
ELSE
$$ This is COOLNT/ON or FLOOD or ….
  DMY=POSTF(2,1,3425,2)         $$ Set INTCOM 3425 to 2 (output with next Z)
ENDIF
DMY=POSTF(13)                   $$ Process COOLNT
CIMFIL / OFF                    $$ End of CIMFIL on COOLNT
```

   Save the post and post-process pplab1.ncl to see the result. COOLNT/OFF is now processed immediately.

   We have just seen **how to change the value of an internal post-processor variable**.

3. Now, one of the tool on our machine is a special tool : TOOL #3 who requires a special coolant code M12.

We are going to make a CIMFIL macro (in fact modify the existing one) to replace M8 by M12 is the current tool is tool #3.

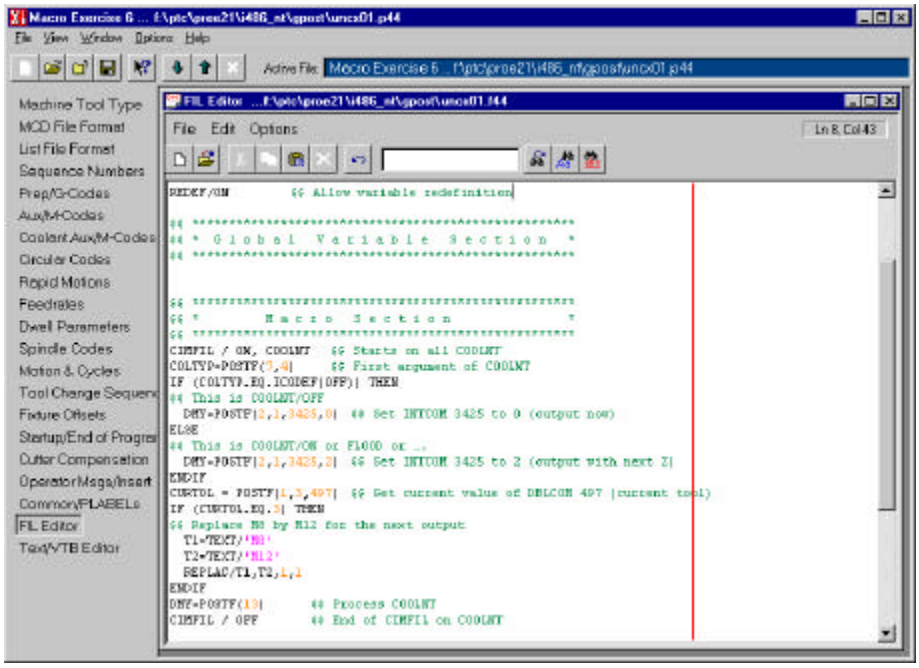


First, let's find the variable for the current tool number : select the **Common/PLABELs** category, sub-panel Search and type "**current tool number"** :

The variable is **DBLCOM 497**
We can now modify our CIMFIL macro on COOLNT :

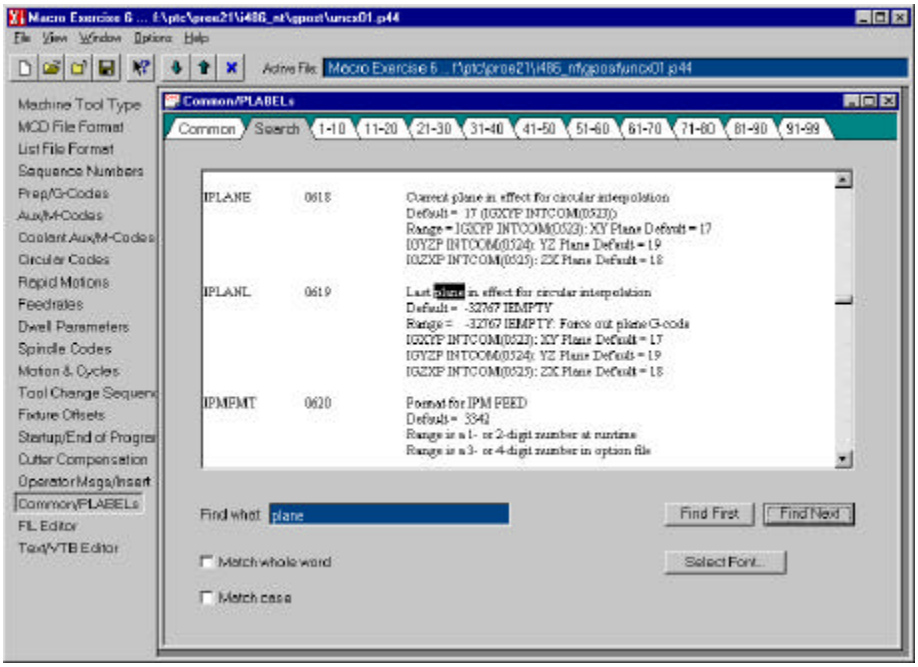
```
CIMFIL / ON, COOLNT            $$ Starts on all COOLNT
COLTYP=POSTF(7,4)              $$ First argument of COOLNT
IF (COLTYP.EQ.ICODEF(OFF)) THEN
$$ This is COOLNT/OFF
  DMY=POSTF(2,1,3425,0)        $$ Set INTCOM 3425 to 0 (output now)
ELSE
$$ This is COOLNT/ON or FLOOD or ….
  DMY=POSTF(2,1,3425,2)        $$ Set INTCOM 3425 to 2 (output with next Z)
ENDIF
CURTOL = POSTF(1,3,497)        $$ Get current value of DBLCOM 497 (current tool)
IF (CURTOL.EQ.3) THEN
$$ Replace M8 by M12 for the next output
  T1=TEXT/'M8'
  T2=TEXT/'M12'
  REPLAC/T1,T2,1,1
ENDIF
DMY=POSTF(13)                  $$ Process COOLNT
CIMFIL / OFF                   $$ End of CIMFIL on COOLNT
```

Do not forget to add **REDEF/ON** in the global area of the FIL file to allow redefinition of variables (T1 and T2).
Save the post and post-process pplab1.ncl to see the result. Check that the coolant code for tool number 3 is M12.



We have just seen **how to check the current value of an internal post-processor variable.**

4. Finally we want to be sure that for a given tool, before the first circular interpolation the G code for the current plane is issued. The default plane is G17, and in our example all



the arcs are in this plane, the post-processor does not generate G17.
Let's find the variable for the last plane code in effect : select the **Common/PLABELs** category, sub-panel Search and type "**plane"** :

The variable is **INTCOM 619**

We can now make a CIMFIL macro on LOADTL to reset this variable at each tool change :

```
CIMFIL / ON, LOADTL          $$ Start on all LOADTL
DMY = POSTF(13)              $$ Process the LOADTL
DMY = POSTF(3,1,619)         $$ Set INTCOM 619 to EMPTY
CIMFIL / OFF
```

Save the post and post-process pplab1.ncl to see the result. Check that G17 is generated before the first circular interpolation of each tool path.

We have just seen **how to set an internal post-processor variable to EMPTY.**

Additional tasks

Find the variable for previous FEDRAT mode
Find the variables for minimum and maximum values of XYZ used for limit checking
Find the variables for last X CL file position

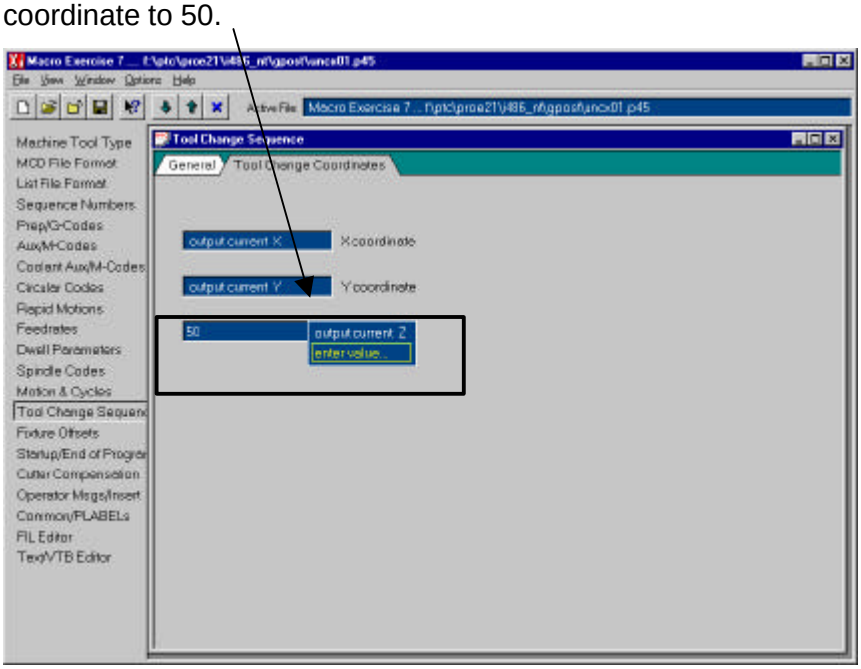
Exercise 7

FILE I/O

This exercise will demonstrate techniques to open, read and write from external files.

Exercise Steps:

1. Create a post-processor for a milling machine, using the control defaults *FANUC 16M Control*. Save the new post as *UNCX01.P17* and use it to post-process the CL file *pplab1.ncl* from the Pro/NC manufacturing model pplab1.mfg. Analyze the results (*pplab1.tap).*
   There is something wrong after the tool change. The machine moves en Z first then in XY. This can be very dangerous. The post-processor is doing what is called “Motion Analysis”, to square off RAPID moves. The problem is that the post does not know the Z value for the tool change.
   To fix this problem we need to modify the Z level of the tool change : Select the **Tool Change Sequence** Category, sub-panel **Tool Change Coordinates** and set Z coordinate to 50.



   Save the post-processor and make a new execution to see the effect of this modification.

The operator of the machine want to have a list of all the tool used during the NC program at the beginning of the tape.
To do this we are going to do a CIMFIL macro on MACHIN and read these tools information from the external file *clfilename.extension.**tl1*** generated by GPOST before the executing the post-processor :

For example ***pplab1.ncl.tl1*** :

```
6
          6   1        1.0000         .0000         .0000         .0000         .0000  0  0
         36   1        2.0000         .0000         .0000         .0000         .0000  0 0
        292   1        3.0000         .0000         .0000         .0000         .0000  0 0
….
```

On the first line we have the total number of tool, **6**.
Each of the following line describes a tool change : CL record number, tool change type (1:LOADTL, 2:TURRET), tool number, offset number, length, X gage length (TURRET) ,Y gage length (TURRET), TURRET index direction (1:CLW, -1:CCLW) and TURRET selected (1:FRONT, 2:REAR, 4:MAIN, …)

```
CIMFIL/ON,MACHIN
DMY=POSTF(13)                $$ Process MACHIN
CLNAME=TEXT/PART             $$ CL file name
T1=TEXT/'.'
NCLPOS=INDXF(CLNAME,T1) $$ find position of . in CL name
$$ remove the version number of CL name
FNAME=TEXT/RANGE,CLNAME,1,NCLPOS+4
FNAME=TEXT/FNAME,'TL1'    $$ Add extension TL1
FEXT=FILEF(1,7,FNAME)        $$ Check if the file exist
IF (FEXT.EQ.1) THEN          $$ Found the file
  INSERT/'( TOOL LIST )$'          $$ message start tool list
  FOPEN=FILEF(1,2,FNAME)    $$ Open the file
  TBUFF=TEXT/READ,1          $$ Read the 1st line of the file
  MXTL=SCALF(TBUFF)          $$ Convert string to real
  IF (MXTL.GT.0) THEN        $$ There are some tools
    DO/ENDDO,I=1,MXTL,1      $$ Loop in the TL1 file
      TBUFF=TEXT/READ,1      $$ Read next line
      TLNUM=TEXT/RANGE,TBUFF,17,30    $$ tool number starts column 17
      TLVAL=SCALF(TLNUM)     $$ Convert to real
         INSERT/'( TOOL ',TLVAL,')$'          $$ Output the TOOL number
    ENDDO) CONTIN            $$ Label for the DO loop
    IF (FOPEN.EQ.0) THEN
      FCLOS=FILEF(1,5)       $$ Close the file
    ENDIF
  ENDIF
  INSERT/'(---------)$'        $$ End tool list
ENDIF
CIMFIL/OFF
```

Save the post and post-process pplab1.ncl to see the result :

```
%
N5 G70
N10( TOOL LIST )
N15( TOOL  1)
N20( TOOL  2)
```

```
N25( TOOL  3)
N30( TOOL  4)
N35( TOOL  5)
N40( TOOL  6)
N45(---------)
N50 T1 M06
….
```

2. Another typicall macro is to prompt the user for something, for example we are going to prompt him for the revision number and display his/her answer with the PARTNO :

   First let's make a macro on PARTNO to store the part name :

```
CIMFIL/ON,PARTNO                $$ Starts on PARTNO
TPART=TEXT/CLW                  $$ Extract text of PARTNO
TPART=TEXT/OMIT,TPART,1         $$ Remove the trailing blank
CIMFIL/OFF
```

   Now we need to add some new lines to our current MACHIN macro :

```
CIMFIL/ON,MACHIN
DMY=POSTF(13)                   $$ PRocess MACHIN
$$
PROMPT=TEXT/'PART',TPART,' - Revision ?:'  $$ Message
RSLT=FILEF(0,1,PROMPT)                     $$ prompt user
STRNG=TEXT/READ,0                          $$ Read his/her answer
SCALR=SCALF(STRNG)                         $$ convert to string
PRGNM=TEXT/CONVF,SCALR,5,0,0,1,1           $$ Format 5 digits
INSERT/('(',TPART,' - REV :',PRGNM,')$'    $$ message in the tape
$$
….
```

Additional tasks

Add the tool offset with the tool number in the tool list at the beginning of the tape.
Remove the **/** in the message with the revision number.

Solution of Additional tasks Exercise 1

Macro a CIMFIL macro to replace all RAPID by FEDRAT/200,IPM
To do this we need to make a CIMFIL macro on RAPID and replace the command by FEDRAT :

```
CIMFIL / ON, RAPID          $$ Starts on all RAPID
FEDRAT / 200, IPM           $$ issue a FEDRAT instead of RAPID
CIMFIL / OFF                $$ End of CIMFIL on RAPID
```

Modify the SPINDL CIMFIL macro to output the AIR command after the processing of SPINDL.

The new CIMFIL macro looks like :

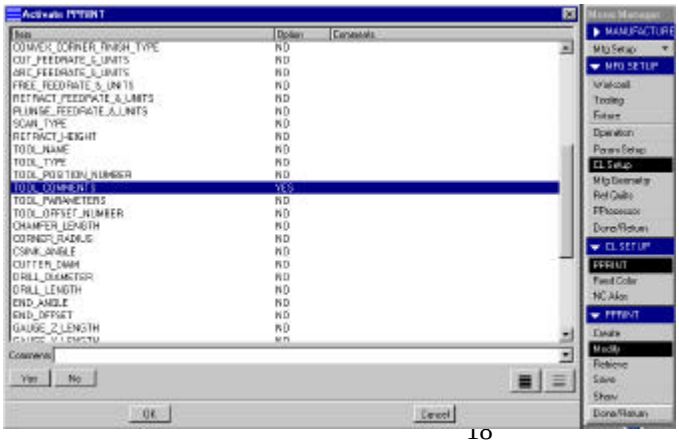
```
CIMFIL / ON, SPINDL                 $$ Starts on all SPINDL statement
DMY=POSTF(13)                       $$ Process SPINDL
PARAM=POSTF(7,4)            $$ first parameter of SPINDL command
IF (PARAM.EQ.ICODEF(OFF)) THEN
$$ it is SPINDL/OFF, stop air first
  AUXFUN/13                 $$ stop AIR
ELSE
$$ it is SPINDL/ON, or SPINDL/RPM,… start air
 AUXFUN/12                  $$ start AIR
ENDIF
 CIMFIL/OFF                 $$ End of CIMFIL on SPINDL
```

Remove all the PPRINT with the word COMMENTS in it.

We need to create a macro on PPRINT :

```
CIMFIL / ON, PPRINT         $$ Starts on all PPRINT
TTEXT = TEXT / CLW          $$ extract the text
TCOM = TEXT / 'COMMENTS'    $$ Comparison value
OK = INDXF(TTEXT,TCOM)      $$ search for TCOM in TTEXT
IF (OK.EQ.0) THEN
$$ TCOM is not found in TTEXT
 DMY=POSTF(13)              $$ Process PPRINT
ENDIF
CIMFIL/OFF                  $$ End of CIMFIL on PPRINT
```

To be able to test this macro, create a PPRINT table in Pro/NC with the pplab1.mfg model, to output **TOOL COMMENTS**



Solution of Additional tasks Exercise 3

Simplify the PARTNO macro (do not use two text variables : see REDEF command)
To allow redefinition of a variable, you need to add the REDEF/ON command in the global section of the FIL macro.
The FIL macros becomes :

```
REDEF/ON

CIMFIL/ON,PARTNO $$ starts on all PARTNO
DMY = POSTF(13)    $$ process the PARTNO
PTXT = TEXT/CLW    $$ gets the text of the PARTNO
PTXT= TEXT/OMIT,PTXT,1  $$ Removes trailing blank
CIMFIL / OFF           $$ End of macro on PARTNO

CIMFIL / ON MACHIN          $$ starts on all MACHIN
DMY=POSTF(13)               $$ process MACHIN
INSERT / '(',PTXT,')$'      $$ inserts the PARTNO as a message
INSERT / 'G90 G80 G40$'     $$ outputs safe startup block
FEDRAT / 10, IPM     $$ default feedrate
CIMFIL / OFF         $$ End of macro on MACHIN
```

Remove the / in the message (see REPLAC command)
You can replace all occurences '**(/**' by '**(**' using the REPLAC command in the global section of the FIL file :

```
T1 = TEXT/'( /'
T2 = TEXT/'('
REPLAC / T1, T2
```

Output the CL file name to the tape (see TEXT command)
The PART modifier returns the CL file name. Add the following lines in the CIMFIL macro on MACHIN :

```
CLNM = TEXT / PART
INSERT / '(',CLNM,')$'
```

Output the Date and Time to the tape (see TEXT command)
The DATE modifier returns the CL file name. Add the following lines in the CIMFIL macro on MACHIN :

```
CURDT = TEXT / DATE
INSERT / '(',CURDT,')$'
```

Solution of Additional tasks Exercise 4

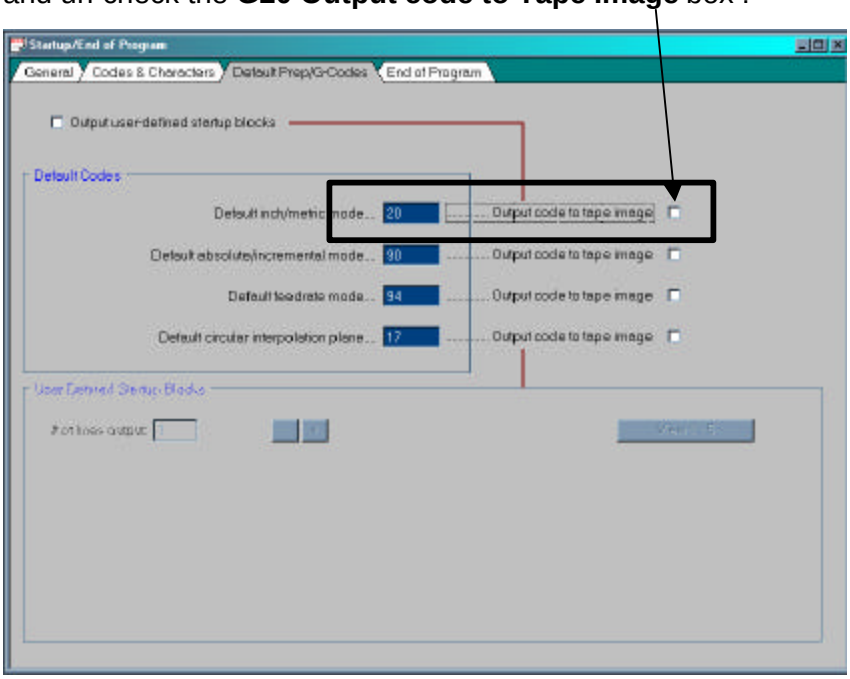
Add space between registers where necessary
If you look at the tape file spaces are missing for M99 and the safe startup block G90G40G80. Replaces the existing INSERTs by the following ones :

INSERT / ‘ M99$’

INSERT / ‘ G90 G40 G80$’

Remove the G20 at the beginning of the tape
The G20 is automatically generated by GPOST. We need to check in Optfile the setting for the start of the program.

1. Starts Optfile and load the UNCX01.P13 post-processor
2. Select the **Startup/End of Program** category, sub-panel **Default Prep./G Codes** and un-check the **G20 Output code to Tape Image** box :



Put the tool name after the tool change
The tool name is coming from a PPRINT in the Pro/NC CL file.
First let’s define two variables in the global section of the FIL file :

OKTOOL=0 $$ to detect if the tool name has been programmed
TOOLNM=TEXT/’NONAME$‘ $$ Default tool name

Then we need to make a CIMFIL macro on PPRINT to catch the tool name :

CIMFIL / ON, PPRINT $$ Starts on all PPRINT
MESS = TEXT/CLW $$ Extract the text of the PPRINT
TESTV=TEXT/’TOOL NAME’ $$ Value to test for
RSLT=INDXF(MESS,TESTV) $$ Test if TESTV is in MESS

```
IF (RSLT.EQ.0) THEN
   PPRINT/MESS          $$ Process the PPRINT
ELSE
   OKTOOL=1             $$ Flag tool name programmed
   TOOLNM=TEXT/MESS          $$ Store tool name
ENDIF
CIMFIL / OFF            $$ End of macro on PPRINT
```

And finally issue the tool name with a CIMFIL on LOADTL. We just need to add the following line at the end of the current CINFIL on LOADTL, just before the CIMFIL/OFF :

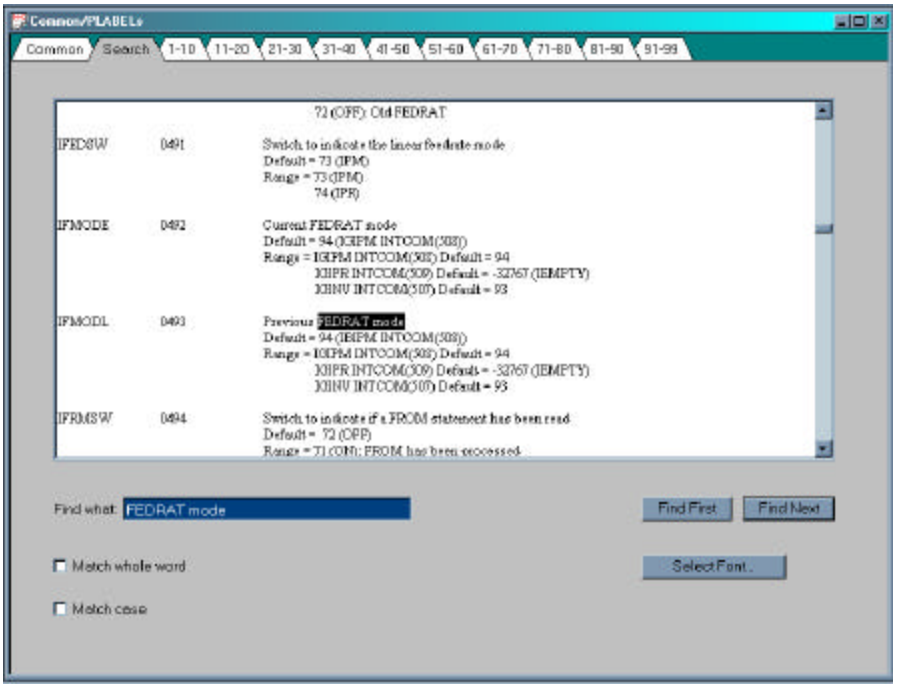
```
IF (OKTOOL.EQ.1) THEN
   OKTOOL=0             $$ Reset the flag
   PPRINT/TOOLNM        $$ Output the message
ENDIF
```

Be sure to start the sub-program with G00 or G01

The post does not process directly the DEFSUB, ENDSUB and CALSUB statement. G codes for linear motion are modals and for the post the first GOTO of the second sub-program is just a continuation of the first sub-program.
The solution is to reset the internal variable for previous feedrate mode at the beginning of each sub-program.
First let's find this variable:

1. Starts Optfile and load the UNCX01.P13 post-processor

2. Select the **Comman/PLABELs** category, sub-panel SEARCH and look for FEDRAT mode :



The variable to reset (set EMPTY) is **INTCOM 493**.

3. Add the following line to the current DEFSUB CIMFIL, just before the CIMFIL/OFF :

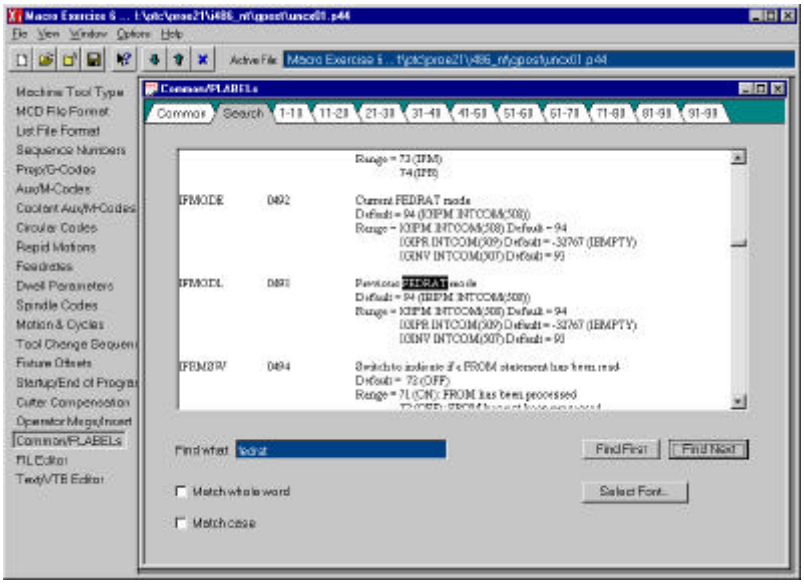
DMY=POSTF(3,1,493) $$ Reset previous feed mode to EMPTY

Ignore SPINDL/OFF
To do this we need to do a CIMFIL macro on SPINDL :

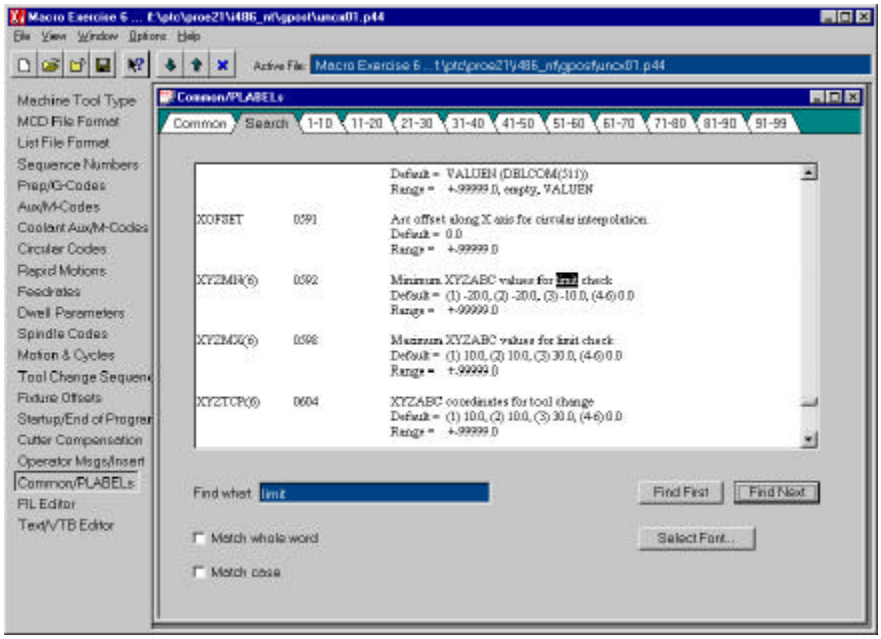
```
CIMFIL / ON, SPINDL              $$ Starts on all SPINDL
ARG1=POSTF(7,4)          $$ First argument
TYP1=POSTF(6,4)          $$ Type of the first argument
$$ Test the first argument to see if it is OFF
IF (TYP1 .EQ. 0 .AND. ARG1 .EQ. (ICODEF(OFF))) THEN
   CONTIN                $$ Do nothing
ELSE
   DMY=POSTF(13)         $$ Process SPINDL
ENDIF
CIMFIL / OFF             $$ end of macro on SPINDL
```

Solution of Additional tasks Exercise 6

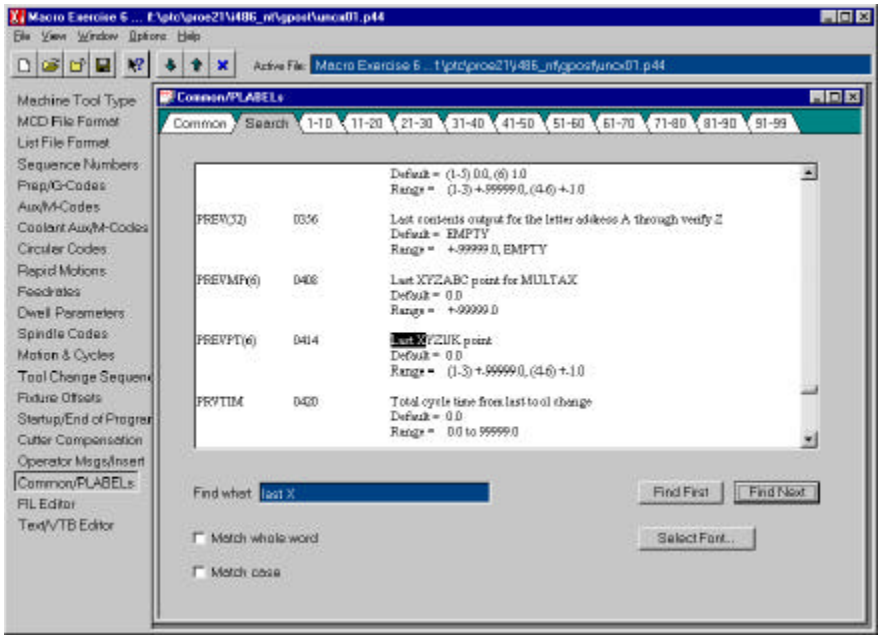
Find the variable for previous FEDRAT mode



Find the variables for minimum and maximum values of XYZ used for limit checking



Find the variables for last X CL file position



Solution of Additional tasks Exercise 7

Add the tool offset with the tool number in the tool list at the beginning of the tape.

The offset number starts at column 31 of the TL1 file.

The new DO loop in the MACHIN macro is now :

```
....
  DO/ENDDO,I=1,MXTL,1          $$ Loop in the TL1 file
     TBUFF=TEXT/READ,1         $$ Read next line
     TLNUM=TEXT/RANGE,TBUFF,17,30    $$ tool number starts column 17
     TLOFF=TEXT/RANGE,TBUFF,31,44    $$ offset number start column 31
     TLVAL=SCALF(TLNUM)        $$ Convert to real
     OFFVAL=SCALF(TLOFF)       $$ Convert to real
     $$ Output the TOOL and OFFSET number
     INSERT/'( TOOL ',TLVAL,' OFFSET ‘,OFFVAL, )$'
   ENDDO) CONTIN               $$ Label for the DO loop
 ...
```

Remove the **/** in the message with the revision number.

There are different techniques to do this :

Use the REPLAC command in the global area of the FIL macro :

```
T1 = TEXT / ‘(  /’
T2 = TEXT /’(‘
REPLAC/T1,T2
```

Or in the PARTNO macro, extract everything after the /

```
CIMFIL/ON,PARTNO                $$ Starts on PARTNO
FPART=TEXT/CLW                  $$ Extract text of PARTNO
FPART=TEXT/OMIT,FPART,1         $$ Remove the trailing blank
T1=TEXT/’/’
SPOS=INDXF(FPART,T1)            $$ Position of the /
LPOS=CANF(FPART,1)              $$ Number of characters of FPART
TPART=TEXT/RANGE,FPART,SPOS+1,LPOS
CIMFIL/OFF
```