

Predator Virtual CNC
Reverse Post Reference Manual

1. Introduction 2
1.1 Process and Data Flow Structure 3
1.2 Features Supported: 3
1.2.1 Features already supported: 3
1.2.2 Check Areas: 4
1.2.3 Features not supported: 5
1.3 Reverse Post files: 5
2. Reverse Post File Contents 5
3. RP* Files 6
3.1 Section identifiers: 6
3.2 Patterns – Section Details 10
3.2.1 Function Groups 11
LATHECYCLES 19
PROGRAMMING 19
ARITHMETIC FUNCTIONS 20
NOTE: The * are for ATL and NC file scanning support. 21
3.2.2 Pattern format 21
3.2.3 FORMAT Keywords 21
3.2.5 FUNCTION keywords: 30
3.2.6 REGISTER Keywords: 106
3.2.7 FORMAT keywords 161
3.2.8 FUNCTION PARAMETERS Keywords: 162
3.5 Parameters: 163
3.4. Error messages: 195
3.4.1 Fatal Errors: 195
3.4.2 Non Fatal Errors: 196
4. RP* File Example 197
1.4 Error checking 204
1.4.1 RP* Configuration Load and Syntax Errors 204
1.4.2 NC and CNC Program Syntax and Semantic checking 206
Long Index: 217

July 12th 2005

1. Introduction

Predator Virtual CNC includes an extremely flexible and robust reverse post processor. The purpose of the reverse post processor is to provide Predator Virtual CNC the ability to read and process any CNC program for a wide range of CNC machine tools.

Every machine tool vendor utilizes different CNC standards, methods, languages and G & M codes to run their equipment. The reverse post processor documents and processes the differences so Predator Virtual CNC can provide complete CNC verification and simulation of the manufacturing process.

For example, some G and M codes on a Fanuc are the same as an Okuma while others are very different. A Heidenhain conversational format is unique to Heidenhain CNC controls. Yet, Predator Virtual CNC supports all three and hundreds more. An unlimited number of different CNC controls can be supported by Predator Virtual CNC via an appropriate reverse post.

For every unique CNC control within your manufacturing facility, an appropriate reverse post should be created to address the unique requirements of each CNC control.

The purpose of this document is to provide a reference manual that can be referred to during the creation and fine tuning of each reverse post processor. Due to the technical nature of this process we would prefer that any technical support be initiated via email and sent to support@predator-software.com as needed. Please attach the reverse post, the CNC programs, tools, setup and job files.

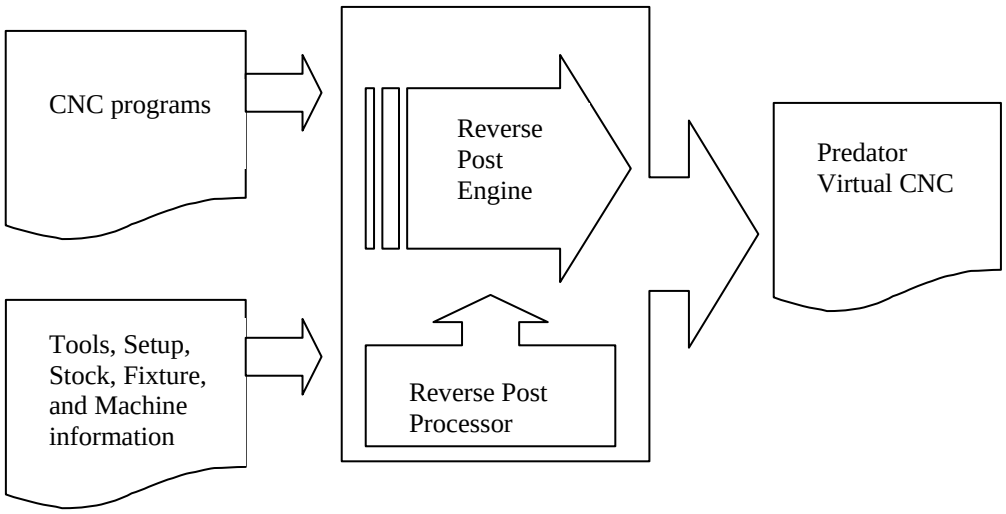
Predator Software does provide a service for the development of reverse post processors. These can be ordered at any time via the following part numbers:

Item #	Description
43020	2 Axis Reverse Post – Lathes
43030	3 Axis Reverse Post – Mills, Routers, Lasers, Plasma, Waterjets or Grinders
43040	4 Axis Reverse Post – Mills, Routers, Lasers, Plasma, Waterjets or Grinders
43050	5 Axis Reverse Post – Mills, Routers, Lasers, Plasma, Waterjets or Grinders

NOTE: Often there can be a fine line between technical support and development of a reverse post. Should this become an issue, we will notify you that your requests require puchase of a reverse post.

Due to the wide range of CNC machine tools, CNC controls, vendors, programming styles and all of the other variables that exist, Predator Software maintains and continues to improve the reverse post processor on a best effort basis. With every release it gets stronger and more flexible. Our goal with the reverse post processor is to provide support for 100% of the functionality in common use by manufacturers worldwide. With an emphasis on the word common. It is an impossible goal to provide 100% of the functionality of every CNC machine tool ever built by every machine tool vendor. At some point in time your requirements may exceed what we currently support. At that point, please contact us via email at support@predator-software.com with the details of what is needed and to discuss the options available.

1.1 Process and Data Flow Structure



The reverse post translates one or more NC and CNC programs into Predator Virtual CNC's internal RVP format. RVP stands for ReVerse-Post, and is a binary equivalent of the CNC program. Virtual CNC not only processes the CNC program but also the setup data in the STP file, before writing the RVP file.

This process requires that the reverse post is properly configured for the particular syntax of the NC and CNC language and the particulars of CNC control. This configuration is implemented in the RPM, RPL or RP5 file. For example, the Fanuc 10A.RPM file contains a complete definition of the syntax and conventions used with this control. With multi-axis machines this process also requires that the machine definition is properly configured for the particular style of multi-axis machine.

The RP* file is loaded before the NC file. Error conditions are determined if the RP* file is incorrect or contains problems.

1.2 Features Supported:

1.2.1 Features already supported:

Linear Interpolation	DONE
Circular Interpolation	DONE
Helical Interpolation	DONE
Spiral Interpolation	DONE
Polar Interpolation	DONE
Cutter Compensation	DONE
Tool Length Compensation	DONE
Absolute vs. Incremental Programming	DONE
Measurement Unit changes	DONE
Feed Rates by IPM, IPR and Inverse Time	DONE
Measurement Unit changes	DONE
Spindle and Coolant Settings	DONE
Work Coordinates	DONE
Global Work Coordinates	DONE
Local Coordinates	DONE
G10 and G54.1 Style Work Coordinates	DONE
Lathe Threading Cycles	DONE

Reference return 1st point	DONE
Canned Holes Cycles:	
Drilling	DONE
Spot Boring	DONE
Pecking	DONE
Tapping	DONE
Boring	DONE
Counter Tapping	DONE
Counter Boring	DONE
Fine Boring	DONE
Ignore Codes	DONE
Comments	DONE
Subroutines	DONE
Sub Programs	DONE
External Sub Programs	DONE
Macro Calls	DONE
Modal Macro Calls	DONE
Custom Macros	DONE
Global and Local Variables	DONE
System and Machine Variables	ONGOING
Expressions	DONE
Lathe Canned Cycles:	
Facing	DONE
Grooving	DONE
Threading	DONE
Roughing	DONE
Finishing	DONE
Scaling	DONE
Rotation	DONE
Square and Rectangular Pockets	DONE
Circular Pockets	DONE
Slot Pockets	DONE
Linear and Grid Hole Cycles	DONE
Arc and Circular Hole Cycles	DONE
IF/THEN and GOTO Conditional Branching	DONE
DO/WHILE and End Branching	DONE
Math Functions	DONE
Trig Functions	DONE
Boolean Functions	DONE
Mill and Lathe Tool Definitions within Comments	DONE
Tool Holders and Tool Kits within Comments	DONE
Stock and Fixture Definitions within Comments	DONE
Diameter, Length and Work Offsets within Comments	DONE
Include other reverse posts	DONE

1.2.2 Check Areas:

Check Spiral Moves	DONE
Check Helical Moves	DONE
Check Tool Loaded	DONE
Check Feed Defined	DONE
Check Spindle Off	DONE
Check Coolant	DONE
Check Cutter Comp Off in G2/G3	ONGOING
Check Cutter Comp Cancel While Ref Point	ONGOING

Check Cutter Comp G41/G42 without G40	ONGOING
Check Length Comp G43/G44 without G49	DONE
Check Travel Limits	DONE
Check Exclusive G & M Codes	DONE

1.2.3 Features not supported:

Exact Stop vs. radius	NOT DONE
2nd Ref Points	NOT DONE
Mill Thread Cutting	NOT DONE
Mirroring	NOT DONE
Functions	NOT DONE

1.3 Reverse Post files:

Each reverse post file is an ASCII file that configures a particular NC control. They have an RP* extension and can be found within one of the subdirectories of the "**Rpost x.0**" directory. By default all reverese posts are located in the following folder:

c:\program files\predator software\common files\rpost 5.1\

NOTE: The version number of the sub directory will change to match the version of Predator Virtual CNC.

The location of the reverse post directory can be specified within Predator Virtual CNC by selecting the Preferences command within the Edit menu. Several sub directories are maintained to group and organize the reverse post by machine class.

..\rpost 5.1\lathe
..\rpost 5.1\mill
..\rpost 5.1\mill-5
..\rpost 5.1\mill-turn

In addition specific extensions are used .RPL is for lathes, .RPM for mills, .RP5 for multi-axis mills and .RMT for mill/turn machines.

NOTE: The MCF and LCF directories and CNC translators belong to previous versions of Predator Virtual CNC and are no longer supported with v4.1 or greater of Predator Virtual CNC. Users upgrading should use an appropriate reverse post vs. the older CNC translator.

2. Reverse Post File Contents

The RP* file defines the relationship between G & M codes in the NC or CNC program and the associated functionality. This is the same process that takes place in the control itself where each NC or CNC code is interpreted before taking effect.

G & M codes are identified through fixed patterns. This reverse-post will identify patterns by comparing NC or CNC program contents with the reverse post's pattern definitions.

When a pattern is identified, a specific functionality is associated to it, and this functionality will generate the actual output tool motion or CNC simulation.

Reverse post files support some useful features to make this process easy:

1) Include files. It is possible to share common CNC control definitions for a family of several specific CNC controls by using a shared RP* file included in several different RP* files. This feature allows nested include files, making it possible to have several levels of shared configurations, from most basic family characteristics to customized controls. The example reverse posts each utilize a single include for common features for scanning tools, machine definitions, offsets, etc.

2) Comments. There are three areas within the RP* file for making comments:
   a) Before the first section identifier
   b) In-line comments after a semicolon ";". A semicolon can be used to start any comment line.
   c) After the [end] statement

3. RP* Files

The RP* file format is ASCII for easy readability and edit ability with text editors such as the Predator CNC Editor. To create a new reverse post step through the following:

1. Copy a similar and existing .RP* file
2. Paste it into the appropriate reverse post directory
3. Rename the new .RP* file appropriately
4. Edit the new .RP* file and implement the necessary changes

RP* files allows reference to include other RP* files. This include capability allows nesting, with a limitation of up to 32 RP* files. All RP* files are concatenated before loaded into the reverse post engine.

All RP* files are loaded by the reverse post engine before the NC or CNC program is read. If there are configuration errors within the RP* files appropriate error messages will be displayed prior to reading any NC or CNC programs.

NOTE: A single .RP* file can be utilized by one or more machine or .MCH files.

3.1 Section identifiers:

RP* files contain information categorized in several sections. Each section has a starting line with the section name in brackets to separate it from previous sections. The only required section is [End] the rest of the sections are optional. Currently there are eight sections allowed within an RP* file:

[Include]	; List additional RP* files to include
[Initializations]	; Specify initial internal CNC control registers and values
[Patterns]	; Define patterns of CNC codes to specific CNC functionality
[Function Groups]	; Map patterns of CNC codes to specific CNC functionality
[Exclusive]	; Specify illegal patterns within a single block or line of code
[Parameters]	; Specify CNC warnings, rules, and control limits
[Arithmetic Functions]	; Map standard arithmetic functions
[Boolean Functions]	; Map boolean or conditional functions
[Trigonometric Functions]	; Map trigonometric functions
[End]	; Specify the end of the RP* file

Additional details on each section are outlined below:

[Include]

This section allows for up to 32 additional RP* files to be included within the current RP* file. Included RP* files have the identical structure including multiple sections. A portion of a Hardinge Fanuc 18.RPL file with an include section follows:

```
[include]          ; List additional RP* files to include
c:\program files\predator software\rpost\mill\fanuc g codes.rpl
c:\program files\predator software\rpost\mill\hardinge m codes.rpl
```

Using the above two include files allows for easier maintenance of a large number of RP* files. If the FANUC G CODES.RPL file is updated then all RP* files that reference it will automatically be updated with the new changes, including the HARDINGE FANUC 18.RPL file.

[Initializations]

This section allows for configuration of Predator Virtual CNC's internal registers and variables by the RP* file. An example of a Fadal.RPL initialization section follows:

```
[INITIALIZATIONS]
"\21"                                   :          space_delimiter
```

[Patterns]

This section maps alphanumeric patterns to specific CNC functionality. By definition the patterns section of an RP* file is very detailed, although considerable effort has been made to keep things simple. A portion of a Fanuc 18.RPM's patterns section follows:

```
[PATTERNS]                                  ; Map patterns of CNC codes to specific CNC functionality
$MOVETO:#10
"X",real          :MoveX                    ; Map X values with a decimal point to X axis motion
"Y",real          :MoveY                    ; Map Y values with a decimal point to Y axis motion
"Z",real          :MoveZ                    ; Map Z values with a decimal point to Z axis motion
"I",real          :MoveI                    ; Map I values with a decimal point to the I register
"J",real          :MoveJ                    ; Map J values with a decimal point to the J register
"K",real          :MoveK                    ; Map K values with a decimal point to the K register
"R",real          :MoveR                    ; Map R values with a decimal point to the R register
"A",int           :MoveA                    ; Map A values without a decimal point to A axis motion
"B",int           :MoveB                    ; Map B values without a decimal point to B axis motion
$MOVETYPE:#100
"G00"             :MoveRapid                ; Map G00 to rapid tool motion
"G0"              :MoveRapid                ; Map G0 to rapid tool motion
"G01"             :MoveLinear               ; Map G01 to linear feed tool motion
"G1"              :MoveLinear               ; Map G1 to linear feed tool motion
"G02"             :MoveCircularCW           ; Map G02 to clockwise circular feed motion
"G2"              :MoveCircularCW           ; Map G2 to clockwise circular feed motion
"G03"             :MoveCircularCCW          ; Map G03 to counter clockwise circular feed motion
"G3"              :MoveCircularCCW          ; Map G3 to counter clockwise circular feed motion
```

Each pattern is enclosed in quotation marks and assigned to a specific function keywords. In the above example both G01 and G1 are mapped to MoveLinear. MoveLinear is a function keyword for linear tool motion. There is no limit to the number of patterns that can be assigned to that same function keyword. If only G01 can be used for linear tool motion, delete or comment out the G1 pattern mapping. Similar patterns are organized into function groups. The above example uses two function groups MOVETO and MOVETYPE to organize the large number of patterns and to assign priorities to patterns.

Pattern priorities are used when multiple patterns are found on a single block or line of code. The higher pattern priorities are processed by Predator Virtual CNC before the lower pattern priorities.

Special characters can be defined within a pattern by using a backslash and the ASCII value 0 to 255. For example, \0 is a null character, \9 is a tab character, \10 is a linefeed character and \13 is a carriage return character. Refer to an ASCII chart for a complete listing of ASCII characters.

Additional details on function groups and priorities are outline further in this chapter.

[Exclusive]

This section allows for exclusive relationships to be defined between specific function keywords. By definition some function keywords must be exclusive of one another. For example, it is illegal to have G0 and G1 codes on the same block or line of code. This exclusive relationship must be defined within the exclusive section. To continue with the example MoveRapid and MoveLinear are exclusive function keywords which maps to the exclusive G codes. A Fanuc 18.RPM's exclusive section follows:

```
[EXCLUSIVE]
MOVETYPE MoveLinear MoveCircularCW MoveCircularCCW MoveRapid
MOVETYPE SelectPlaneXY SelectPlaneYZ SelectPlaneZX
CUTTERCOMP LengthCompCancel CutterCompLeft CutterCompRight LengthCompPlus LengthCompMinus
```

After each functional group mutually exclusive function keywords are specified. In the above example the first exclusive line specifies that only one G0, G1, G2 or G3 code is allowed per block or line. The second specifies that only one G17, G18 or G19 code is allowed per block or line. The third specifies that only one G40, G41, G42, G43 or G44 is allowed per block or line.

[Parameters]

This section allows for a wide range of CNC warnings, rules and control limits to be specified. Many parameters are either On or Off. Others parameters require a number or require specific keywords. A portion of a Fanuc 18.RPM's parameters section follows:

```
[PARAMETERS]
Check_Spindle_Speed=On          ; On requires M3 or M4 before tool motion
Check_Helical_Move=Off          ; Off allows helical interpolation
SubProg_Pos=Bottom              ; Bottom requires all subs to be below the main program
SubProg_Nest=4                  ; Sets the maximum number of nested subs
SubProg_Max=16                  ; Sets the maximum number of subs
```

Each parameter is followed by an equal sign and an appropriate value. The first two are Check parameters and require an On or Off. The third parameter, SubProg_Pos, accepts None, Top, Bottom or Both. The last two SubProg parameters require a number. Details on each parameter are available later in the chapter.

[Arithmetic Functions]

This section allows for mapping of arithmetic functions to be defined within the RP* file. A portion of a Fanuc 18.RPM's arithmetic functions section follows:

```
[ARITHMETIC FUNCTIONS]
"+",#5              : Add()
"-",#5              : Subtract()
"*",#10             : Multiply()
"/",#10             : Divide()
"^",#20             : Exponent()
"MOD",#20           : Module()
```

Each arithmetic functions is followed by a priority number. Priority numbers are defined with a pound sign followed by a number and are used to configure the order of operation. The Exponent and Module functions are assigned a priority of 20 and since they are higher then the others they will be performed first when no parenthesis are used within the CNC program to override the default priorities specified within the RP* file. Multiplication and division are assigned a priority of 10 and these functions will be performed after any exponents and before any additions and subtractions. The add and subtract functions are assigned a priority of 5 and will be performed last. Details on each parameter are available later in the chapter.

[Boolean Functions]

This section allows for mapping of boolean functions to be defined within the RP* file. A portion of a Fanuc 18.RPM's boolean functions section follows:

```
[BOOLEAN FUNCTIONS]
"OR",#30          : Or(double in)
"AND",#30         : And(double in)
"XOR",#30         : Xor(double in)
"EQ",#30          : Equal(double in)
"NE",#30          : NotEqual(double in)
"GT",#30          : Great(double in)
"LT",#30          : Less(double in)
"GE",#30          : GreatOrEqual(double in)
"LE",#30          : LessOrEqual(double in)
```

Boolean functions are also assigned a priority number. In the above case the priority is 30 and since it is higher then all of the arithmetic functions, all boolean functions will be calculated prior to any arithmetic functions. The above example also illustrates how the operators and the operands can be defined for each math function with the DOUBLE, SINGLE, IN and PRE function keywords. Details on each parameter are available later in the chapter.

[Trigonometric Functions]

This section allows for mapping of trigonometric functions to be defined within the RP* file. A portion of a Fanuc 18.RPM's trigonometric functions section follows:

```
"ABS",#10         : Absolute(single pre)
"ACOS",#10        : ArcCosine(single pre)
"ATAN",#10        : ArcTangent(single pre)
"COS",#10         : Cosine(single pre)
"EXP",#10         : Exponent(single pre)
"LN",#10          : Logarithm(single pre)
"ROUND",#10       : Round(single pre)
"SIN",#10         : Sine(single pre)
"SQRT",#10        : SquareRoot(single pre)
"TAN"#10          : Tangent(single pre)
```

Details on each parameter are available later in the chapter.

[End]

This section specifies the end of the reverse post definition and allows for additional comments or a history of changes to be defined within the RP* file. A Fanuc 18.RPM's end section follows:

```
[END]
Date        RPL History
01/01/99    Created original Fanuc 18 definition.
03/30/99    Added support for canned cycles and cutter compensation
04/15/99    Added support for subroutines, variables and expressions
```

05/01/99 Added support for stock and tool definitions within comments

3.2 Patterns – Section Details

The pattern section is organized into multiple areas. Currently there are three areas within a pattern section: multiple pattern definition, function groups, and pattern mapping.

Multiple Pattern Definition

The multiple pattern definition of an RP* file allows nested hierarchical groups of patterns to be defined. Many G & M codes require additional secondary G & M codes. For example, on Fanuc 12 control a subprogram call is an M98 and it requires a secondary P value and an optional L value immediately after the M98 on the same block or line of code. This multiple pattern definition must be defined before the M98 can be assigned to a function keyword. A portion of a Fanuc 18.RPM's multiple pattern definition section follows:

```
[patterns]
@6:#0                                   ; Begin multiple pattern 6 with a priority of 0
"P",int          : r_subprogram_number  ; Map the P value to the SubProgram_Number register
"\0"             : separator            ; Optionally a separator can be used
"L",int          : r_subprogram_times   ; Map the L value to the SubProgram_Times register

$SUBPROGRAM: #5
"O",int          : SubProgramStart      ; Map the O value to a start of a subprogram
":",int          : SubProgramStart      ; Map the : value to a start of a subprogram
"M98",multi@6    : SubProgramCall       ; Map M98 to call a subprogram with pattern 6
"M99"            : SubProgramRet        ; Map M99 to end a subprogram and return
```

Within the SUBPROGRAM function group each line maps CNC codes to function keywords. The first two lines maps either the O and the following number or a : and the following number to indicate the beginning of a subprogram. The next line maps the M98 P and the following number L and the following number to call a subprogram, which subprogram to call and the number of times to execute the subprogram. Pattern 6 provides the secondary mapping for the M98 and must be defined before it can be referenced with the multi@6. Finally the last line maps the M99 to end a subprogram and return.

NOTE: Register keywords begin with an r_register_name and they allow specific G & M codes to be assigned to internal registers within Predator Virtual CNC. The above example uses two registers r_subprogram_number and r_subprogram_times. Although CNC controls don't use register names, these registers are identical to those on real CNC controls. Registers are maintained for modal values and updated as the NC or CNC program is processed just like real CNC controls.

Function Groups:

Function groups organize the large number of pattern mappings and to assign priorities to those pattern mappings. Functions groups must start with a $ followed by the name of the function group a semi colon and a patter priority. Pattern priorities are used when multiple patterns are found on a single block or line of code. A portion of a Fanuc 18.RPM's patterns section follows:

```
[patterns]
$MOVETO:#10                             ; Starts MOVETO function group with a priority 10
"X",real         :MoveX
"Y",real         :MoveY
"Z",real         :MoveZ
$MOVETYPE:#100                          ; Starts MOVETYPE function group with a priority 100
"G0"             :MoveRapid
"G1"             :MoveLinear
```

Function keywords are only supported within its function group. For example, the MoveX function keyword can only be mapped within the MOVETO function group. A complete listing of function groups and their function keywords are provided later in this chapter.

Pattern:

Patterns allows for an unlimited mapping of specific CNC codes to specific CNC functionality. Each pattern is enclosed in quotation marks and assigned to a specific function keywords. A portion of a Fanuc 18.RPM's patterns section follows:

```
[patterns]                              ; Map patterns of CNC codes to specific CNC functionality
$MOVETYPE:#100
"G01"           :MoveLinear             ; Map G01 to linear feed tool motion
"G1"            :MoveLinear             ; Map G1 to linear feed tool motion
```

In the above example both G01 and G1 are mapped to MoveLinear. MoveLinear is a function keyword for linear tool motion. There is no limit to the number of patterns that can be assigned to that same function keyword. If only G01 can be used for linear tool motion, delete or comment out the G1 pattern mapping. Patterns can be defined in upper case, lower case or both. It also may use delimiters, have a strict length or may be specified using multiple lines. Additional details are available in section 3.2.2 and 3.2.4

3.2.1 Function Groups

Function groups have the following syntax:

$functional_group_name:#priority_number

Every function group must begin with a dollar sign followed by the function group name then a colon a number sign and finally the priority number. All of the Fanuc 12.RPL's function groups are listed below:

```
$DATASEPARATOR:#0
$MOVETO:#10
$MOVETYPE:#100
$TOOLCHANGE:#20
$MISCELLANEOUS:#11
$FEEDRATE:#110
$CANNEDCYCLES:#20
$UNSUPPORTED:#1
$IGNORED:#1
$REFPOINTRETURN:#20
$CUTTERCOMP:#20
$WORKCOORD:#30
$ABSOLUTEZERO:#5
$SUBPROGRAM:#5
$VARIABLES:#4
$THREADING:#1
$STOCKDEF:#1
$LATHECYCLES:#1
$PROGRAMMING:#1
$COMMENTS:#1
```

The priority number sets the priority level for all of the function keywords contained within the function group. These priorities are used when multiple G & M codes or function keywords are found on the same NC block or line of code. Higher priority function groups are processed before lower priority function groups. For example, the workcoord function group has a priority of 30 and it's function keywords are calculated and processed prior to the cannedcycles function group which has a priority of 20.

NOTE: If the same function group is repeated the second function group definition will be used by the reverse post and the first definition will be ignored. For example:

```
$TOOLCHANGE:#50
"T",int                         :     SelectTool
$TOOLCHANGE:#40
"M06"                           :     LoadTool
"M6"                            :     LoadTool
"MTOOL",multi@12                :     CreateMillTool
```

The above will not work because the second function group overrides the first. Since the second function group does not contain any tool selection keywords Predator Virtual CNC will display errors when processing a CNC program. The intent of the above example is to establish a higher priority for selecting a tool vs. loading a tool. Since every line can have it's own priority, the correct method looks like the following:

```
$TOOLCHANGE:#50
"T",int,#50                     :     SelectTool
"M06",#40                       :     LoadTool
"M6",#40                        :     LoadTool
"MTOOL",multi@12                :     CreateMillTool
```

The above will work when either T01 M06 or when M06 T1 style programming is used. Because the higher priority is assigned to the T code and a lower priority is assigned to the M06.

All function groups are listed below with their function keywords:

MOVETO

The moveto function group includes all function keywords that map different linear and rotary axis of motion including X, Y, Z, I, J, K, R, A, B and several others. All moveto function keywords are listed below with a short description:

IncMoveX ; Maps the linear horizontal or X axis of incremental motion
IncMoveY ; Maps the linear horizontal or Y axis of incremental motion
IncMoveZ ; Maps the linear horizontal or Z axis of incremental motion
InsertMove ; Maps the insert moves for turning profiles
MoveX ; Maps the linear horizontal or X axis of motion
MoveY ; Maps the linear vertical or Y axis of motion
MoveZ ; Maps the linear depth or Z axis of motion
MoveI ; Maps the circle center horizontal or X axis of motion
MoveJ ; Maps the circle center horizontal or Y axis of motion
MoveK ; Maps the circle center depth or Z axis of motion
MoveR ; Maps the circle radius motion
MoveA ; Maps the rotary horizontal or A axis of motion
MoveB ; Maps the rotary vertical or B axis of motion
Move4thAxis ; Maps the rotary horizontal or A axis of motion
Move5thAxis ; Maps the rotary vertical or B axis of motion
MoveXYZUVW ; Maps the point and tool orientation linear motion
MoveIJKUVW ; Maps the center point and tool orientation circular ; motion
MoveFromXYZ ; Maps the general point and tool orientation linear motion
MoveXYZIJK ; Maps the general point and tool orientation linear motion
MoveU ; Maps the linear horizontal or X axis of incremental motion ; for turning
MoveV ; Maps the linear horizontal or Z axis of incremental motion ; for turning

PolarRadius ; Maps the polar radius axis of motion
PolarAngle ; Maps the polar angle axis of motion

NOTE: Movetype's group internal code is 100100.

See section **3.2.5.1** for more information on each movetype function keywords.

MOVETYPE

The movetype function group includes all function keywords that map different type of tool motion including linear interpolation, circular interpolation, absolute & incremental programming, inch & metric programming, and rotary table motion. All movetype function keywords are listed below with a short description:

AbsoluteCoord ; Maps absolute coordinate programming
CircularMoveSense ; Maps the cw or ccw direction of circular interpolation
CircleCenter ; Maps circle center definition for circular interpolation
CancelRotation ; Maps cancel modal rotation
CancelScaling ; Maps modal cancel scaling
CancelPolarCoord ; Maps cancel for modal polar definition
DefUnits ; Maps either inch or metric coordinate programming
IncrementalCoord ; Maps incremental coordinate programming
Inches ; Maps inch or imperial coordinate programming
Millimeters ; Maps metric coordinate programming
MoveCircular ; Maps circular interpolation
MoveCircularCW ; Maps clockwise circular interpolation
MoveCircularCCW ; Maps counter clockwise circular interpolation
MoveLinear ; Maps linear feed interpolation
MoveRapid ; Maps linear rapid interpolation
Multax ; Maps 4 and 5 axis programming
PolarCenterAbs ; Maps absolute polar center coordinate definition
PolarCenterInc ; Maps incremental polar center coordinate definition
PolarCoord ; Maps modal polar definition
PolarMoveLinear **;** Maps polar definition for linear interpolation
PolarMoveCircular **;** Maps polar definition for circular interpolation
Rotation ; Maps modal rotation
Scaling ; Maps modal scaling
SelectPlaneXY ; Maps XY plane selection for circular interpolation
SelectPlaneZX ; Maps ZX plane selection for circular interpolation
SelectPlaneYZ ; Maps YZ plane selection for circular interpolation
SetABSenseCW ; Maps the cw direction of rotary axis interpolation
SetABSenseCCW ; Maps the ccw direction of rotary axis interpolation
SelectPlane **;** Maps plane selection for circular interpolation

NOTE: Movetype's group internal code is 100000.

See section **3.2.5.2** for more information on each movetype function keywords.

TOOLCHANGE

The toolchange function group includes all function keywords that create tool definitions, load tools, and select tools. All toolchange function keywords are listed below with a short description:

LoadTool ; Maps loading the tool from the tool changer
SelectTool ; Maps selecting the tool from the tool carousel
SelectAndLoadTool ; Maps selecting and loading the tool from the tool changer
CreateMillTool ; Maps tool comments for tool size and shape definitions
CreateLatheTool ; Maps tool comments for tool size and shape definitions

SetTLBFileName* ; Maps specifying TLB filename
LoadTLBFileName* **;** Maps loading TLB filename

NOTE: Toolchange's group internal code is 100200.

See section **3.2.5.3** for more information on each toolchange function keywords.

CANNEDCYCLES

The cannedcycles function group includes all function keywords that map different types of canned cycle tool motion including holes, circles, arcs, lines and bores. Additional canned cycle function keywords include canceling and returning to rapid planes. All cannedcycles function keywords are listed below with a short description:

ArcCycle ; Reserved for future use and has not been implemented
ArcHoleCycle ; Maps arc pattern cycle definition
BoreCycle ; Maps bore cycle definition
CallCycle ; Maps calling cycle
CancelCannedCycle ; Maps canceling the current active canned cycle
CancelPatternCycle ; Maps canceling the current active pattern cycle
CircleCycle ; Reserved for future use and has not been implemented
CircPocketCycle ; Maps circular pocket cycle definition
CircPocketCycleCW ; Maps circular pocket cycle definition
CircPocketCycleCCW ; Maps circular pocket cycle definition
CircPocketCycleDef ; Maps circular pocket cycle definition
DefCycle ; Maps multi line cycle definition
DrillCycle ; Maps drilling, pecking, tapping, and boring cycles
LineCycle ; Maps line pattern cycle definition
MatrixHoleCycle ; Maps matrix pattern cycle definition
PatternCycle ; Maps pattern cycle definition
ReturnToInitial ; Maps returning to initial rapid plane in a canned cycle
ReturnToReference ; Maps returning to reference rapid plane in a canned cycle
RectPocketCycle ; Maps rectangular pocket cycle definition
RectPocketCycleDef ; Maps rectangular pocket cycle definition
SlotPocketCycle ; Maps slot pocket cycle definition

NOTE: Cannedcycles' group internal code is 100250.

See section **3.2.5.4** for more information on each cannedcycles function keywords.

MISCELLANEOUS

The miscellaneous function group includes all function keywords that map standard M codes including spindle direction, spindle stop, program end, optional stop, and coolant status. All miscellaneous function keywords are listed below with a short description:

AirBlast ; Maps enabling/disabling air blast
ChangePallet ; Maps changing pallet
CoolantON ; Maps coolant on
CoolantOFF ; Maps coolant off
CtantSurfSpeed ; Maps enabling constant surface speed
CtantSurfSpeedCancel ; Maps canceling constant surface speed
DisableRotaryComp ; Maps disabling 5 axis rotary tool compensation
EnableRotaryComp ; Maps enabling 5 axis rotary tool compensation
HspindleStop ; Maps horizontal spindle stop
LineNumber ; Maps line number control
MaxSpindleSpeed ; Maps maximum spindle speed

OptStop	; Maps optional stop
Pause0	; Maps program stop
Pause1	; Maps optional stop
ProgramEnd	; Maps end of program
SetMCHFileName	; Maps setting machine file name
SetOrigRelocation	; Maps setting origin relocation
SetMachineDatum	; Maps setting machine datum
SpindleCW	; Maps clockwise spindle motion
SpindleCCW	; Maps counter clockwise spindle motion
SpindleStop	; Maps spindle stop
SpindleSpeed	; Maps spindle speed
Stop	; Maps program stop
ToolClamp	; Maps tool clamp/unclamp
VspindleStop	; Maps vertical spindle stop

NOTE: Miscellaneous' group internal code is 100300.

See section **3.2.5.5** for more information on each miscellaneous function keywords.

FEEDRATE

The feedrate function group includes all function keywords that map the different feed rate codes including feed per minute and feed per revolution. All feedrate function keywords are listed below with a short description:

FeedPerMinute	; Maps feed rate per minute
FeedPerRevolution	; Maps feed rate per revolution
InvTimeFeed	; Maps inverse time feed
OutFeed	; Maps feed rate
SetFeed	; Maps feed rate
SetMaxFeed	; Maps maximum feed rate

NOTE: Feedrate's group internal code is 100400.

See section **3.2.5.6** for more information on each of feedrate's function keywords.

UNSUPPORTED

The unsupported function group includes only one function keyword it maps the unsupported CNC codes that should produce an error within Predator Virtual CNC. The unsupported error function keyword is listed below with a short description:

Error	; Maps CNC codes to an error condition

The above example would create a warning for every user when ever a G19 is specified within the CNC program.

NOTE: The unsupported group internal code is 100500.

See section **3.2.5.7** for more information on the unsupported function keyword.

REFPOINTRETURN

The refpointreturn function group includes the function keywords that map returning to or from a reference point. These refpointreturn function keywords are listed below with a short description:

ReturnFromRefP	; Maps return from reference point

ReturnToRefP ; Maps return to reference point

NOTE: Refpointreturn's group internal code is 100600.

See section **3.2.5.8** for more information on the refpointreturn's function keywords.

CUTTERCOMP

The cuttercomp function group includes all function keywords that map the different cutter compensation codes including diameter comp right, left and cancel. In addition, length comp plus, minus and cancel. All cuttercomp function keywords are listed below with a short description:

CutterCompLeft ; Maps diameter compensation left
CutterCompRight ; Maps diameter compensation right
CutterCompCancel ; Maps canceling diameter compensation
LengthCompPlus ; Maps positive length compensation
LengthCompMinus ; Maps negative length compensation
LengthCompCancel ; Maps canceling length compensation
LengthOffPositive ; Maps increased or positive length compensation
LengthOffNegative ; Maps decreased or negative length compensation
LengthOffDoublePositive ; Maps double positive length compensation
LengthOffDoubleNegative ; Maps double negative length compensation
SetLengthCompH ; Maps the actual length compensation value
SetCutterCompD ; Maps the actual diameter compensation value
SetDiamOffset ; Maps setting the diameter compensation value
SetLengthOffset ; Maps setting the length compensation value

NOTE: Cuttercomp's group internal code is 100700.

See section **3.2.5.9** for more information on cuttercomp's function keywords.

IGNORE

The ignore function group includes only one function keyword it maps the CNC codes that should be ignored by Predator Virtual CNC. The ignore function keyword is listed below with a short description:

Ignore ; Maps CNC codes that should be ignored

NOTE: The ignore group internal code is 100800.

See section **3.2.5.10** for more information on the ignore function keyword.

WORKCOORD

The workcoord function group includes all function keywords that map the different work or fixture offset codes. All workcoord function keywords are listed below with a short description:

AxisRotation ; Maps setting axis rotation
CancelAxisRotation ; Maps canceling axis rotation
ChangeAllWorkCoord ; Maps changing all work coordinates
ChangeOffset ; Maps work or fixture offset's X, Y and Z values
ChangeProgramOrigin ; Maps changing the programming point
DisableWorkCoord ; Maps disabling work coordinates
LocalCoordSys ; Maps a local work offset
SetCoordSystem ; Maps setting system coordinates
SetWorkCoord ; Maps up to 100 work or fixture offsets
SetWorkCoord1 ; Maps first work or fixture offset

SetWorkCoord2	; Maps second work or fixture offset
SetWorkCoord3	; Maps third work or fixture offset
SetWorkCoord4	; Maps fourth work or fixture offset
SetWorkCoord5	; Maps fifth work or fixture offset
SetWorkCoord6	; Maps sixth work or fixture offset
Translate	; Maps a work offset translation

NOTE: Workcoord's group internal code is 101400.

See section **3.2.5.11** for more information on workcoord's function keywords.

ABSOLUTEZERO

The absolutezero function group includes only one function keyword it maps the code to set the absolute zero position. The setabsolutezero function keyword is listed below with a short description:

SetAbsoluteZero	; Maps setting the absolute zero position

NOTE: Absolutezero's group internal code is 101500.

See section **3.2.5.12** for more information on the setabsolutezero function keyword.

THREADING

The threading function group includes only one function keyword it maps the code to set the thread cutting motion. The threadmove function keyword is listed below with a short description:

ThreadCycle	; Maps the thread cutting cycle
ThreadMove	; Maps the thread cutting move

NOTE: The threading group internal code is 101600.

See section **3.2.5.13** for more information on the threadmove function keyword.

SUBPROGRAMS

The subprograms function group includes all function keywords that map the different subprogram codes including starting and ending a subprogram. In addition, calling a subprogram and returning from a subprogram. All subprogram function keywords are listed below with a short description:

EndMacroModalCall	; Maps the end of modal subprogram calling
MacroModalCall	; Maps calling a modal subprogram
ProgramStart	; Maps the start of the main program
SubProgramStart	; Maps the start of each subprograms
SubProgramEnd	; Maps the end of each subprogram
SubProgramCall	; Maps calling a subprogram
SubProgramRet	; Maps returning from a subprogram
SubProgramRetEnd	

NOTE: Subprogram's group internal code is 101700.

See section **3.2.5.14** for more information on subprogram's function keywords.

VARIABLES

The variables function group includes all function keywords that map defining, setting, and getting variables, parameters, global and local variables. All variable function keywords are listed below with a short description:

DefineGlobalVar	; Maps defining a global variable
DefineLocalVar	; Maps defining a local variable
DefineParameter	; Maps defining a parameter
GetActiveTool	; Maps getting active tool
GetActiveDCode	; Maps getting active D offset value
GetCurrentVar	; Maps getting current variable
GetCurrentReg	; Maps getting current register
GetGlobalVar	; Maps getting a global variable
GetLocalVar	; Maps getting a local variable
GetParameter	; Maps getting a parameter
GetVariable	; Maps getting a variable
SetGlobalVar	; Maps setting a global variable
SetLocalVar	; Maps setting a local variable
SetParameter	; Maps setting a parameter
SetVariable	; Maps setting a variable

NOTE: Variables' group internal code is 101800.

See section **3.2.5.15** for more information on variables' function keywords.

DONOTHING

This section is nearly identical to the IGNORE section with one difference. Any G and M codes that are defined within the DONOTHING section will not display a warning message regardless of the ENABLE_WARNING parameter. So some G codes should display a warning and some G codes should be ignored without displaying a warning message. Set the ENABLE_WARNING parameter to ON and add the G codes that should display a warning message in the IGNORE section and add the G codes that should be ignored without displaying a warning message in the DONOTHING section.

DoNothing	; Ignore CNC codes without warning messages

NOTE: The ignore group internal code is 109990.

See section **3.2.5.16** for more information on the Ignore function keyword.

COMMENTS

The comments function group includes all function keywords that map the different comment codes including start, end, and comments at the end of a NC block or line of code. All comments function keywords are listed below with a short description:

CommentFollows	; Maps the remainder of the line as a comment
CommentStarts	; Maps the start of a comment
CommentEnds	; Maps the end of a comment

NOTE: Comments' group internal code is 512.

See section **3.2.5.17** for more information on comments' function keywords.

OPERATORS – Reserved for Backwards Compatibility

This function group is no longer supported and does not work. The operators function group allows for mapping of math operators to be defined within the RP* file.

FUNCTIONS – Reserved for Backwards Compatibility

This function group is no longer supported and does not work. The functions function group allows for mapping of math functions to be defined within the RP* file.

STOCKDEF

The stockdef function group includes all function keywords that map the different stock and fixture creation definitions often contained within comments. Standard stock and fixture shapes include boxes, cylinders, holes, and custom shapes via STL files. All stockdef function keywords are listed below with a short description:

CreateStockBox	; Maps stock box creation
CreateStockHole	; Maps stock box hole creation
CreateStockFixt	; Maps fixture box creation
CreateStockCyl	; Maps stock cylinder creation
CreateStockCylHole	; Maps stock cylinder hole creation
CreateStockCylFixt	; Maps fixture cylinder creation
CreateStockSTL	; Maps STL stock creation
CreateStockSTLFixt	; Maps STL fixture creation
SetSTKFileName*	; Maps setting STK filename

NOTE: The Group internal code is 101900.

This group is used to specify the patterns that will be used to get stock data from the NC file.

See section **3.2.5.20** for more information on stockdef's function keywords.

LATHECYCLES

The lathecycles function group includes all function keywords that map the different lathe canned cycle definitions. Standard OD and ID cycles for roughing, semi-finishing and finishing cycles are supported. In addition, tapping, grooving, face drilling are also supported. All lathecycles function keywords are listed below with a short description:

TurnCycle	; Maps rough and semi-finish cycles
FinishCycle	; Maps finish cycles
RoughZCycle	; Maps Z roughing cycles
RoughXCycle	; Maps X roughing cycles
TurnTapCycle	; Maps tapping cycles
TurnGrooveCycle	; Maps grooving cycles
FaceDrillCycle	; Maps face drilling cycles
StartLabel	; Maps the start of canned cycles
EndLabel	; Maps the end of canned cycles

NOTE: The Group internal code is 102000.

This group is used to specify the patterns that will be used to lathe cycles.

See section **3.2.5.21** for more information on lathecycle's function keywords.

PROGRAMMING

The programming function group includes all function keywords that map the different looping and conditional definitions. IF THENs, DO WHILEs, and GOTOs are supported. All programming function keywords are listed below with a short description:

Condition	; Maps conditional statements

StartLoop	; Maps the beginning of loops
EndLoop	; Maps the end of loops
Goto	; Maps goto style jump statements
SetLabel	; Maps the beginning of labels

NOTE: The Group internal code is 102100.

This group is used to specify the patterns that will be used to lathe cycles.

See section **3.2.5.22** for more information on programming's function keywords.

ARITHMETIC FUNCTIONS

The arithmetic function group includes all function keywords that map the basic math functions. Addition, subtraction, multiplication, division, modulus division and exponentiation are supported. All arithmetic function keywords are listed below with a short description:

Add	; Maps additions
Subtract	; Maps subtractions
Multiply	; Maps multiplications
Divide	; Maps divisions
Module	; Maps modulus
Exponent	; Maps exponentiations

NOTE: The Group internal code is 102200.

This group is used to specify the patterns that will be used for basic math functions.

See section **3.2.5.23** for more information on arithmetic's function keywords.

BOOLEAN FUNCTIONS

The boolean function group includes all function keywords that map the boolean and conditional math functions. And, or, equal and not equal along with many others are supported. All boolean function keywords are listed below with a short description:

And	; Maps ands
Or	; Maps ors
Xor	; Maps xors
Equal	; Maps equals
NotEqual	; Maps not equals
Great	; Maps greater thans
Less	; Maps less thans
GreatOrEqual	; Maps greater than or equals
LessOrEqual	; Maps less thans or equals

This group is used to specify the patterns that will be used for boolean math functions.

See section **3.2.5.24** for more information on boolean's function keywords.

TRIGONOMETRIC FUNCTIONS

The trigonometric function group includes all function keywords that map the trigonometric and advanced math functions. Sin, cosine, tangent, round, and square root along with many others are supported. All trigonometric function keywords are listed below with a short description:

SINE	; Maps sines

COSINE	; Maps cosines
TANGENT	; Maps tangents
ARCTANGENT	; Maps arc tangents
SQUAREROOT	; Maps square roots
ABSOLUTE	; Maps absolutes
ROUND	; Maps rounds
ARCCOSINE	; Maps arc cosines
LOGARITHM	; Maps logarithms
EXPONENT	; Maps exponents

This group is used to specify the patterns that will be used for trigonometric and advanced math functions.

See section **3.2.5.25** for more information on trigonometric's function keywords.

NOTE: The * are for ATL and NC file scanning support.

3.2.2 Pattern format

The general format of a pattern redirector is the following:

Pattern : Function

Where **Pattern** is defined as:

"G & M Codes", format keyword, optional secondary format keywords

and the **Function** is defined as:

function keyword (optional function parameters)

Examples:

"\0"	: DataSeparator
"X",real	: MoveX
"S",int	: SpindleSpeed
"G1"	: MoveLinear
"G01"	: MoveLinear
"G28",multi@5	: ReturnToRefP
"G29",multi@5,delimters	: ReturnFromRefP

See section 3.2.3, 3.2.4 and 3.2.5 for more information

3.2.3 FORMAT Keywords

Format keywords typically specify an additional pattern that follows the main pattern contained in quotes. Using a format keyword allows for definition of specific data types to be assigned to the main pattern. The combination of the main pattern with the format keyword is then assigned to a function keyword. For example, on Fanuc CNC controls feed rates are not just the letter F they are F5.5 or F100 an F followed by a number. In addition, feed rates require a specific type of number, a real number. Defining these precise patterns can be configured by setting the main pattern, within quotes, and then it's format keywords. In the following examples, the main pattern is "F" followed by a comma and a format keyword real. The "F" and the real number that follows it is then assigned to update Predator Virtual CNC's current feed rate with the SetFeed function keyword.

"F",real : SetFeed

All format keywords are listed below with a short description:

Casesens	Allows the pattern to be upper, lower or both cases
Concat	Concatenate multiple lines together for processing
Expression	Math expression follows the main pattern
Group	Group follows the main pattern – Reserved
Int	Integer number follows the main pattern
Ignore_case	Allows the pattern to be any case
Multi	One or more secondary patterns follows the main pattern
Null	Nullifies any data following the main pattern
Optional2	Optional 2 Place factored real number follows the main pattern
Optional3	Optional 3 Place factored real number follows the main pattern
Optional4	Optional 4 Place factored real number follows the main pattern
Real	Real number follows the main pattern
Real2	2 Place factored real number follows the main pattern
Real3	3 Place factored real number follows the main pattern
Real4	4 Place factored real number follows the main pattern
String	Alphanumeric string follows the main pattern
Switch	One of many secondary patterns follows the main pattern
Void	Voids the requirement of a format keyword, default

In addition, register specific format keywords are available when multiple patterns are used to define registers within Predator Virtual CNC and there is no main pattern. All register specific format keywords are listed below with a short description:

Int_reg	Integer number follows the main pattern
Real_reg	Real number follows the main pattern
Real_reg2	2 Place factored real number follows the main pattern
Real_reg3	3 Place factored real number follows the main pattern
Real_reg4	4 Place factored real number follows the main pattern
String_reg	Alphanumeric string follows the main pattern

Casesens

Description: Case independent format

Tech Tips: Allows patterns that are specified in upper, lower case or both to be defined as the same function keyword. If a pattern's case needs to be unique do not use the casesens format keyword. The casesens format keyword is easier than defining multiple patterns just to support different case combinations to the same function keyword. For example, lets assume that sub program names can be specified in any case and the CNC would refer to the same subprogram. Use the casesens format keyword to make CALL POCKET, call pocket or Call Pocket be identical to Predator Virtual CNC. Defining these patterns can be configured by setting the following:

"CALL",string,casesens : SubProgramCall

Modifiers: Refer to section 3.2.2.2

Format code: 1 [0x00000001]

Concat

Description: Concatenate multiple lines together for processing

Tech Tips: Allows a pattern to span across multiple lines. For example, APTCL often defines circular motion with multiple lines. Defining these patterns can be configured by setting the following:

"CIRCLE",multi@10,concat : MoveIJKUVW

Format code: 2048 [0x00000800]

Expression

Description: Expression format
Tech Tips: Identifies that an expression or series of basic mathematical calculations will follow the pattern defined within quotes. An expression can contain a series of +, -, * and / with appropriate values. Typically expressions are used to calculate a numeric value that would otherwise be constant. For example, on Fanuc CNC controls with Custom Macro feed rates can be specified as a constant F10.5 or calculated using an expression F10+10 for a calculated feed rate of F20. Expressions also support variables and advanced math functions. For example, F#100/COS[45]. Defining these patterns can be configured by setting the following:

"F",expression : SetFeed

Modifiers: Refer to section 3.2.2.2
Format code: 8192 [0x2000]

Group – Reserved for future use

Description: Group format
Modifiers: Refer to section 3.2.2.2
Format code: 4096 [0x1000]

Int

Description: Integer number format
Tech Tips: Identifies that an integer number will follow the pattern defined within quotes. An integer number can be positive or negative but can not contain a decimal point or other separator with additional digits. Typical letter addresses that use the int format identifiers include O, N, D, H, S, E and P. For example, on most Fanuc CNC controls D1 – D99 indicate radius or diameter cutter compensation and H1 – H99 indicate tool length or height compensation. Defining these patterns can be configured by setting the following:

"D",int : SetCutterCompD
"H",int : SetCutterCompH

Modifiers: Refer to section 3.2.2.2
Format code: 22 [0x0016]

Int_reg

Description: Integer number format for Format groups only
Tech Tips: Identifies that an integer number will be specified within a format group. An integer number can be positive or negative but can not contain a decimal point or other separator with additional digits. For example, on the Vickers 2100 CNC control the number of times to repeat a sub program can be called with, CLS POCKET,2 which calls a subprogram called pocket twice. Defining these patterns within a format group can be configured by setting the following:

@6:#0
%s,string_reg : r_SubProgram_String

```
","                    : Separator
%d,int_reg             : r_SubProgram_Times
```

Tech Tips: The difference between int_reg and real_reg format identifiers is that int_reg assumes that there will not be a decimal point and additional digits while real_reg assumes that a decimal point and additional digits is possible.
Modifiers: Refer to section 3.2.2.2
Format code: 1040 [0x0410]

Multi@#

Description: Refers to a list of items from which any of the items are allowed and the # specifies the list of items.
Tech Tips: Identifies one of many possible patterns from a list of multiple patterns. An unlimited number of multiple patterns are allowed. For example, on most CNC controls G28 and G29 is followed by one or more X, Y and Z values. The multi format identifier is used to allow any combination of X, Y, and Z values to follow a G28 and G29, return to or from a reference point. The multiple pattern @5 is a list of possible patterns. These patterns are defined as X, Y and Z values. The multi@5 format identifier specifies that one or more patterns from within @5 will follow a G28 or G29. Defining these patterns within the format group can be configured by setting the following:

```
@5:#0
"X",real               : r_reference_pos_X
"Y",real               : r_reference_pos_Y
"Z",real               : r_reference_pos_Z

$REFPOINTRETURN:#20
"G28",multi@5          : ReturnToRefP
"G29",multi@5          : ReturnFromRefP
```

Tech Tips: The difference between switch and multi format identifiers is that switch assumes that there will only be one of the items in the list will follow the pattern while multi assumes that one or more items in the list will follow the pattern.
Modifiers: Refer to section 3.2.2.2
Format code: 38 [0x0026]

Null

Description: Null format
Tech Tips: Identifies that nothing after the patterns should be recognized by Predator Virtual CNC. For example, on most CNC controls comments are enclosed within parenthesis and their contents should be ignored. Defining this pattern can be configured by setting the following:

```
"(",null               : CommentStarts
")",null               : CommentEnds
```

Modifiers: Refer to section 3.2.2.2
Format code: 0 [0x0000]

Optional2

Description: Similar to Real2, but it will optionally be divided by a factor of 100

Tech Tips: Identifies that a real number when a placeholder is present or with 2 assumed decimal places of accuracy when there is no placeholder. An optional2 number can be positive or negative but may or may not contain a decimal point or other separator with additional digits. Typical letter addresses that use the optional2 format identifiers include X, Y, Z, A, B, C, U, W, P, D, H and F. For example, X4 has a value of 0.04 inches while X4. has a value of 4.0 inches. Defining these patterns can be configured by setting the following:

"X",optional2 : MoveX

Modifiers: Refer to section 3.2.2.2
Format code: 6 [0x0016]

Optional3

Description: Similar to Real3, but it will optionally be divided by a factor of 1000
Tech Tips: Identifies that a real number when a placeholder is present or with 3 assumed decimal places of accuracy when there is no placeholder. An optional3 number can be positive or negative but may or may not contain a decimal point or other separator with additional digits. Typical letter addresses that use the optional3 format identifiers include X, Y, Z, A, B, C, U, W, P, D, H and F. For example, Y5 has a value of 0.005 inches while Y5. has a value of 5.0 inches. Defining these patterns can be configured by setting the following:

"Y",optional3 : MoveY

Modifiers: Refer to section 3.2.2.2
Format code: 6 [0x0016]

Optional4

Description: Similar to Real4, but it will optionally be divided by a factor of 10000
Tech Tips: Identifies that a real number when a placeholder is present or with 4 assumed decimal places of accuracy when there is no placeholder. An optional4 number can be positive or negative but may or may not contain a decimal point or other separator with additional digits. Typical letter addresses that use the optional4 format identifiers include X, Y, Z, A, B, C, U, W, P, D, H and F. For example, Z6 has a value of 0.0006 inches while Z6. has a value of 6.0 inches. Defining these patterns can be configured by setting the following:

"Z",optional2 : MoveZ

Modifiers: Refer to section 3.2.2.2
Format code: 6 [0x0016]

Real

Description: Real number format
Tech Tips: Identifies that a real number will follow the pattern defined within quotes. A real number can be positive or negative but can contain a decimal point or other separator with additional digits. Typical letter addresses that use the real format identifiers include X, Y, Z, A, B, C, U, W, and F. For example, on most Fanuc CNC controls feed rates can be within the range of F.1 – F999.9 for milling and F.001 – F999.999 for turning. Defining these patterns can be configured by setting the following:

"F",real : SetFeed

Tech Tips: The difference between int and real format identifiers is that int assumes that there will not be a decimal point and additional digits while real assumes that a decimal point and additional digits is possible.
Modifiers: Refer to section 3.2.2.2
Format code: 6 [0x0016]

Real2

Description: Similar to Real, but it will be divided by a factor of 100
Tech Tips: Identifies that a real number with 2 assumed decimal places of accuracy. A real2 number can be positive or negative but will not contain a decimal point or other separator with additional digits. Typical letter addresses that use the real2 format identifiers include X, Y, Z, A, B, C, U, W, P, D, H and F. For example, P250 has a value of 2.5 seconds. Defining these patterns can be configured by setting the following:

"P",real2 : Dwell

Modifiers: Refer to section 3.2.2.2
Format code: 6 [0x0016]

Real3

Description: Similar to Real, but it will be divided by a factor of 1000
Tech Tips: Identifies that a real number with 3 assumed decimal places of accuracy. A real3 number can be positive or negative but will not contain a decimal point or other separator with additional digits. Typical letter addresses that use the real3 format identifiers include X, Y, Z, A, B, C, U, W, P, D, H and F. For example, P2500 has a value of 2.5 seconds. Defining these patterns can be configured by setting the following:

"P",real3 : Dwell

Modifiers: Refer to section 3.2.2.2
Format code: 6 [0x0016]

Real4

Description: Similar to Real, but it will be divided by a factor of 10000
Tech Tips: Identifies that a real number with 4 assumed decimal places of accuracy. A real4 number can be positive or negative but will not contain a decimal point or other separator with additional digits. Typical letter addresses that use the real4 format identifiers include X, Y, Z, A, B, C, U, W, P, D, H and F. For example, P25000 has a value of 2.5 seconds. Defining these patterns can be configured by setting the following:

"P",real4 : Dwell

Modifiers: Refer to section 3.2.2.2
Format code: 6 [0x0016]

Real_reg

Description: Real number format for Format groups only
Tech Tips: Identifies that an real number will be specified within a format group. A real number can be positive or negative but can contain a decimal point or other separator with additional digits. For example, in an APTCL file the

GOTO statement is followed by six numbers indicating X, Y, Z, U, V and W values. U, V, and W values are also known as I, J, K tool axis normals or vectors. Defining these patterns within a format group can be configured by setting the following:

```
@1:#0
%f,real_reg          : r_Pos_X
%f,real_reg          : r_Pos_Y
%f,real_reg          : r_Pos_Z
%f,real_reg          : r_Pos_U
%f,real_reg          : r_Pos_V
%f,real_reg          : r_Pos_W

$MOVETO:#21
“GOTO”,multi@1       : MoveXYZUVW
```

Modifiers: Refer to section 3.2.2.2
Format code: 1024 [0x0400]

Real_reg2

Description: Similar to Real_reg, but it will be divided by a factor of 100

Real_reg3

Description: Similar to Real_reg, but it will be divided by a factor of 1000

Real_reg4

Description: Similar to Real_reg, but it will be divided by a factor of 10000

String

Description String format
Tech Tips: Identifies that an alphanumeric string will follow the pattern defined within quotes. An alphanumeric string can be any combination of letters or numbers. For example, on the Vickers 2100 CNC control sub program names can be used, DFS POCKET defines the start of a subprogram called pocket. Defining these patterns can be configured by setting the following:

```
“DFS”,string         : SubProgramStart
```

Modifiers: Refer to section 3.2.2.2
Format code: 518 [0x0206]

String_reg

Description String format for Format Groups only
Tech Tips: Identifies that an alphanumeric string will be specified within a format group. An alphanumeric string can be any combination of letters or numbers. For example, on the Vickers 2100 CNC control sub program names can be called with, CLS POCKET,2 which calls a subprogram called pocket twice. Defining these patterns within a format group can be configured by setting the following:

```
@6:#0
%s,string_reg        : r_SubProgram_String
“,”                  : Separator
%d,int_reg           : r_SubProgram_Times
```

Modifiers: Refer to section 3.2.2.2
Format code: 1536 [0x0600]

Switch@#

Description: Refers to a list of items from which only one item is allowed and the # specifies the list of items.
Tech Tips: Identifies only one possible pattern from a list of multiple patterns. An unlimited number of multiple patterns are allowed. For example, on most CNC controls G90 is absolute and G91 is incremental but both can not be used at the same time. The switch format identifier is used to allow either G90 or G91 style coordinates within hole cycles. Two multiple pattern definitions are used. The first @3 is a list of choices from where only one item is allowed G90 or G91. The second @2 is the multiple pattern definition for hole cycles with one of the patterns referring to @3 to allow either a G90 or G91. Defining these patterns within the two format groups can be configured by setting the following:

```
@3:#0
“G90”                   : use_absolute
“G91”                   : use_incremental
@2:#0
“”,switch@3             : r_coord_mode
"X",real                : r_drill_pos_X
"Y",real                : r_drill_pos_Y
"Z",real                : r_peck_Z
"Q",real                : r_peck_inc
"R",real                : r_peck_retract
"L",int                 : r_peck_repeats
"P",int                 : r_dwell_time
```

Modifiers: Refer to section 3.2.2.2
Format code: 262 [0x0106]

Void

Description: Void format
Tech Tips: Identifies that the pattern if fully defined within the quotes. This is the default pattern format identifier and does not need to be specified every time. The example reverse posts included with Predator Virtual CNC use the implied void format and do not specify it each time. For example, on most Fanuc CNC controls G54 – G59 indicate work or fixture offsets. G54, G56 and G58 are defined using the void keyword. G55, G57 and G59 also use the void keyword but it is implied and not implicitly defined. Defining these patterns can be configured by setting the following:

```
“G54”,void      : SetWorkCoord1
“G55”           : SetWorkCoord2
“G56”,void      : SetWorkCoord3
“G57”           : SetWorkCoord4
“G58”,void      : SetWorkCoord5
“G59”,          : SetWorkCoord6
```

Tech Tips: The difference between void and null format identifiers is that null assumes that there is additional information, that should be ignored, while void assumes there is no additional information.
Modifiers: Refer to section 3.2.2.2

Format code: 66 [0x0042]

3.2.4 Multiple Pattern Secondary Format Keywords

Secondary format keywords can optionally follow the Multi@# format keyword to define complex patterns. Using a secondary format keywords can change the default optional definition to require a specific order, length or pattern delimiters. The combination of the main pattern with the Multi@# format keyword with the secondary format keywords are then assigned to a function keyword.

All secondary format keywords are listed below with a short description:

Delimiters	Requires the pattern to use the specified delimiters
Multigstricord	Requires the pattern to be in a specific order
Multigstriclen	Requires the pattern to be a specific length

Delimiters

Description: Requires the multiple patterns to use the specified delimiters.

Tech Tips: Predator Virtual CNC's default separator is a space character. If your CNC programs always have spaces between the G & M codes then the delimiter format keyword is not necessary. Often, CNC programs have no spaces between G & M codes. To address this separators can be specified within the multiple pattern definition or with a $Dataseparator function group. For example, on most CNC controls spaces between the G & M codes may or may not be necessary. The "\0" pattern is a null character and is defined as a separator after the X and Y values within the multiple pattern @5. Defining these patterns within the format group can be configured by setting the following:

```
@5:#0
"X",real            : r_reference_pos_X
"\0"                : separator
"Y",real            : r_reference_pos_Y
"\0"                : separator
"Z",real            : r_reference_pos_Z

$REFPOINTRETURN:#20
"G28",multi@5       : ReturnToRefP
"G29",multi@5       : ReturnFromRefP
```

Second example, if many different separators are necessary; spaces, no spaces or tabs between the G & M codes. The $Dataseparator function group with the delimiter format keyword can be used. Defining these patterns within the format group can be configured by setting the following:

```
@5:#0
"X",real            : r_reference_pos_X
"Y",real            : r_reference_pos_Y
"Z",real            : r_reference_pos_Z

$DATASEPARATOR:#0
"\32"               : dataseparator
"\0"                : dataseparator
"\9"                : dataseparator

$REFPOINTRETURN:#20
"G28",multi@5,delimeters        : ReturnToRefP
```

"G29",multi@5,delimeters : ReturnFromRefP

Format code: 8 [0x00000008]

Multigstrictlen

Format code: 16777216 [0x01000000]
Description: Forces the pattern to have a strict length only works with multi.

Multigstrictord

Description: Requires the pattern and its sub-patterns to follow a strict order only works with multi.
Tech Tips: For example, in an APTCL file the GOTO statement is followed by six numbers indicating X, Y, Z, U, V and W values. U, V, and W values are also known as I, J, K tool axis normals or vectors. APTCL files also require that the order of X, Y, Z, U, V and W are always the same. Adding the secondary format keyword multigstrictord forces the order required within the APTCL file. Defining these patterns within a format group can be configured by setting the following:

```
@1:#0
%f,real_reg          : r_Pos_X
%f,real_reg          : r_Pos_Y
%f,real_reg          : r_Pos_Z
%f,real_reg          : r_Pos_U
%f,real_reg          : r_Pos_V
%f,real_reg          : r_Pos_W

$MOVETO:#21
"GOTO",multi@1,multigstricord          : MoveXYZUVW
```

Format code: 33554432 [0x02000000]

3.2.5 FUNCTION keywords:

Function keywords map the specific CNC function to each pattern. Function keywords typically relate to specific G & M codes or combinations or codes. Functions keywords are organized into function groups. Each function keyword can only be used within it's function group.

Following there is a description of functions per function group.

3.2.5.1 MOVETYPE Function Group

The movetype function group includes all function keywords that map different type of tool motion including linear interpolation, circular interpolation, absolute & incremental programming, inch & metric programming, and rotary table motion. All movetype function keywords are listed below with a short description:

AbsoluteCoord ; Maps absolute coordinate programming
CircularMoveSense ; Maps the cw or ccw direction of circular interpolation
CircleCenter ; Maps circle center definition for circular interpolation
CancelPolarCoord ; Maps cancel modal polar style coordinate programming
CancelRotation ; Maps cancel modal rotation
CancelScaling ; Maps cancel modal scaling
DefUnits ; Maps either inch or metric coordinate programming
Dwell ; Maps dwells
IncrementalCoord ; Maps incremental coordinate programming
Inches ; Maps inch or imperial coordinate programming
Millimeters ; Maps metric coordinate programming

MoveCircular ; Maps circular interpolation
MoveCircularCW ; Maps clockwise circular interpolation
MoveCircularCCW ; Maps counter clockwise circular interpolation
MoveLinear ; Maps linear feed interpolation
MoveRapid ; Maps linear rapid interpolation
Multax ; Maps 4 and 5 axis programming
PolarCenterAbs ; Maps absolute polar style coordinate programming
PolarCenterInc ; Maps incremental polar style coordinate programming
PolarCoord ; Maps modal polar style coordinate programming
PolarMoveLinear ; Maps polar definition for linear interpolation
PolarMoveCircular ; Maps polar definition for circular interpolation
Rotation ; Maps modal rotation
RotationInc ; Maps incremental modal rotation
Scaling ; Maps modal scaling
SelectPlaneXY ; Maps XY plane selection for circular interpolation
SelectPlaneYZ ; Maps YZ plane selection for circular interpolation
SelectPlaneZX ; Maps ZX plane selection for circular interpolation
SetABSenseCW ; Maps the cw direction of rotary axis interpolation
SetABSenseCCW ; Maps the ccw direction of rotary axis interpolation
SelectPlane ; Maps plane selection for circular interpolation

AbsoluteCoord

Description: Maps absolute coordinate programming
Tech Tips: Selects absolute coordinate positioning and overrides previous method of coordinate positioning. Examples using this function keyword include the following:

"G90" : AbsoluteCoord ; Fanuc style
"(90)" : AbsoluteCoord ; Dynapath conversational style
"DIM ABS" : AbsoluteCoord ; Anilam conversational style

Function code: 101
Group code: 100000
Scope: Modal
RVP codes: 7xxx

CircularMoveSense

Description: Maps the cw or ccw direction of circular interpolation
Tech Tips: Selects cw or ccw direction of circular tool motion at the current feed rate. Examples using this function keyword include the following:

"D",int : CircularMoveSense ; Dynapath style

Function code: 6
Group code: 100000
Scope: Modal
RVP codes: 717x, 718x, 719x

CircleCenter

Description: Maps the center of the circle for circular interpolation
Tech Tips: Selects the center of a circle. Examples using this function keyword include the following:

"CC" : CircleCenter ; Heidenhain style

Function code: 6

Group code: 100000
Scope: Modal
RVP codes: 717x, 718x, 719x

CancelPolarCoord

Description: Maps cancellation of polar coordinate style programming
Tech Tips: Selects cancellation of polar coordinate mode. Examples using this function keyword include the following:

“G15” : CancelPolarCoord ; Fanuc style

Function code: 19
Group code: 100000
Scope: Modal
RVP codes: 7xxx

CancelRotation

Description: Maps cancellation of rotation
Tech Tips: Selects cancellation of rotation mode. Examples using this function keyword include the following:

“G69” : CancelRotation ; Fanuc style
“CYCL*DEF*10.1*ROT+0” : CancelRotation ; Heidenhain style

Function code: 15
Group code: 100000
Scope: Modal
RVP codes: 7xxx

CancelScaling

Description: Maps cancellation of scaling
Tech Tips: Selects cancellation of scale mode. Examples using this function keyword include the following:

“G50” : CancelScaling ; Fanuc style or
“CYCL*DEF*26.1*X1*Y1*CCX+0*CCY+0” : CancelScaling ; Heidenhain style or
“G14” : CancelScaling ; Allen Bradley style

Function code: 17
Group code: 100000
Scope: Modal
RVP codes: 7xxx

DefUnits

Description: Maps either inch or metric coordinate programming
Tech Tips: Selects either inch or metric mode. Examples using this function keyword include the following:

"BEGIN",multi@3 : DefUnits ; Heidenhain style

Function code: 7
Group code: 100000
Scope: Modal
RVP codes: 7xxx

Dwell

Description: Maps dwells
Tech Tips: Selects dwells. Examples using this function keyword include the following:

```
"G4",multi@20          : Dwell  ; Fanuc style
"G04",multi@20         : Dwell
"(8)L",real            : Dwell  ; Dynapath style
"(8)*L",real           : Dwell
```

Function code: 11
Group code: 100000
Scope: Modal
RVP codes: 7xxx

IncrementalCoord

Description: Maps incremental coordinate programming
Tech Tips: Selects incremental coordinate positioning and overrides previous method of coordinate positioning. Examples using this function keyword include the following:

```
“G91”          : IncrementalCoord     ; Fanuc style
“(91)”         : IncrementalCoord     ; Dynapath conversational style
 “DIM INC”     : IncrementalCoord     ; Anilam conversational style
```

Function code: 100
Group code: 100000
Scope: Modal
RVP codes: 7xxx

Inches

Description: Maps inch or imperial style coordinate programming
Tech Tips: Selects inch units for all tool motion. Examples using this function keyword include the following:

```
“G20”          : Inches               ; Fanuc style
“G70”          : Inches               ; Multiple definitions are supported
"(S)P0"        : Inches               ; Dynapath style
"(S)*P0"       : Inches
"Unit~IN"      : Inches               ; Anilam style
```

Function code: 300
Group code: 100000
Scope: Modal
RVP codes: 71xx

Millimeters

Description: Maps millimeters or metric style coordinate programming
Tech Tips: Selects millimeter units for all tool motion. Examples using this function keyword include the following:

```
“G21”          : Millimeters          ; Fanuc style
“G71”          : Millimeters          ; Multiple definitions are supported
"(S)P1"        : Millimeters          ; Dynapath style
"(S)*P1"       : Millimeters
```

"Unit~MM" : Millimeters ; Anilam style

Function code: 301
Group code: 100000
Scope: Modal
RVP codes: 71xx
Equivalent Gcodes: FANUC : “G21”

MoveCircular

Description: Maps circular interpolation
Tech Tips: Selects circular tool motion at the current feed rate. Clockwise or counter clockwise direction is determined by a separate pattern mapped to the circularmovesense function keyword. Examples using this function keyword include the following:

“C” : MoveCircular ; Heidenhain style
“CR” : MoveCircular ; Heidenhain style
“(2)” : MoveCircular ; Dynapath style

Function code: 5
Group code: 100000
Scope: Modal
RVP codes: 717x, 718x, 719x

MoveCircularCW

Description: Maps clockwise circular interpolation
Tech Tips: Selects clockwise circular tool motion at the current feed rate. Examples using this function keyword include the following:

“G2” : MoveCircularCW ; Fanuc style
“G02” : MoveCircularCW ; Multiple definitions are supported
“DR-” : MoveCircularCW ; Heidenhain conversational style
“ARC CW” : MoveCircularCW ; Anilam conversational style

Function code: 2
Group code: 100000
Scope: Modal
RVP codes: 7172, 7182, 7192

MoveCircularCCW

Description: Maps counter clockwise circular interpolation
Tech Tips: Selects counter clockwise circular tool motion at the current feed rate. Examples using this function keyword include the following:

“G3” : MoveCircularCCW ; Fanuc style
“G03” : MoveCircularCCW ; Multiple definitions are supported
“DR+” : MoveCircularCCW ; Heidenhain conversational style
“ARC CCW” : MoveCircularCCW ; Anilam conversational style

Function code: 3
Group code: 100000
Scope: Modal
RVP codes: 7173, 7183, 7193

MoveLinear

Description: Maps linear feed interpolation
Tech Tips: Selects linear tool motion at the current feed rate. Examples using this function keyword include the following:

```
"G1"        : MoveLinear        ; Fanuc style
"G01"       : MoveLinear        ; Multiple definitions are supported
"(1)"       : MoveLinear        ; Dynapath conversational style
"L"         : MoveLinear        ; Heidenhain conversational style
"LINE"      : MoveLinear        ; Anilam conversational style
```

Function code: 1
Group code: 100000
Scope: Modal
RVP codes: 7001, 7501, 7601

MoveRapid

Description: Maps linear rapid interpolation
Tech Tips: Selects linear tool motion at the rapid rate. Examples using this function keyword include the following:

```
"G0"        : MoveRapid         ; Fanuc style
"G00"       : MoveRapid         ; Multiple definitions are supported
"(0)"       : MoveRapid         ; Dynapath conversational style
"R F6000"   : MoveRapid         ; Heidenhain conversational style
"RAPID"     : MoveRapid         ; Anilam conversational style
```

Function code: 4
Group code: 100000
Scope: Modal
RVP codes: 6000

Multax

Description Maps 4 and 5 axis programming
Tech Tips: Enables multi-axis or 4 and 5 axis tool motion. Typically this function keyword is used for APTCL style reverse posts only. Examples using this function keyword include the following:

```
"MULTAX",switch@4,casesens  : Multax          ; Enables 4 and 5 axis tool motion
```

Function code: 302
Group code: 100000
RVP codes

PolarCenterAbs

Description: Maps absolute polar style coordinate programming
Tech Tips: Selects absolute polar coordinate positioning and overrides previous method of coordinate positioning. Examples using this function keyword include the following:

```
"G111",multi@42         : PolarCenterAbs                ; Seimens style
```

Function code: 100
Group code: 100000
Scope: Modal

RVP codes: 7xxx

PolarCenterInc

Description: Maps incremental polar style coordinate programming
Tech Tips: Selects incremental polar coordinate positioning and overrides previous method of coordinate positioning. Examples using this function keyword include the following:

```
"G110",multi@42        : PolarCenterInc              ; Seimens style
```

Function code: 100
Group code: 100000
Scope: Modal
RVP codes: 7xxx

PolarCoord

Description: Maps polar style coordinate programming
Tech Tips: Selects polar coordinate positioning and overrides previous method of coordinate positioning. Examples using this function keyword include the following:

```
"G16"                  : PolarCoord                  ; Fanuc style
```

Function code: 100
Group code: 100000
Scope: Modal
RVP codes: 7xxx

PolarMoveLinear

Description: Maps polar linear feed interpolation
Tech Tips: Selects polar linear tool motion at the current feed rate. Examples using this function keyword include the following:

```
"LP",multi@21  : PolarMoveLinear      ; Heidenhain style
```

Function code: 1
Group code: 100000
Scope: Modal
RVP codes: 7001, 7501, 7601

PolarMoveCircular

Description: Maps polar circular feed interpolation
Tech Tips: Selects polar circular tool motion at the current feed rate. Examples using this function keyword include the following:

```
"CP",multi@21  : PolarMoveCircular    ; Heidenhain style
"CTP",multi@21: PolarMoveCircular     ; Heidenhain style
```

Function code: 3
Group code: 100000
Scope: Modal
RVP codes: 7173, 7183, 7193

Rotation

Description: Maps modal rotation

Tech Tips: Selects rotation mode. Examples using this function keyword include the following:

```
"G68",multi@19              : Rotation      ; Fanuc style or
"CYCL*DEF*10.0",multi@24    : Rotation      ; Heidenhain style
"CYCL*DEF*10.1",multi@24    : Rotation      ; Heidenhain style
"ROT",multi@27              : Rotation      ; Seimens style
```

Function code: 101
Group code: 100000
Scope: Modal
RVP codes: 7xxx

RotationInc

Description: Maps incremental modal rotation
Tech Tips: Selects incremental rotation mode. Examples using this function keyword include the following:

```
"AROT",multi@29             : RotationInc   ; Seimens style
```

Function code: 101
Group code: 100000
Scope: Modal
RVP codes: 7xxx

Scaling

Description: Maps modal scaling
Tech Tips: Selects scale mode. Examples using this function keyword include the following:

```
"G51",multi@18              : Scaling       ; Fanuc style or
"CYCL*DEF*26.0",multi@25    : Scaling       ; Heidenhain style or
"CYCL*DEF*26.1",multi@25    : Scaling
"G14.1",multi@19            : Scaling       ; Allen Bradley style
```

Function code: 101
Group code: 100000
Scope: Modal
RVP codes: 7xxx

ScalingInc

Description: Maps incremental modal scaling
Tech Tips: Selects scale mode. Examples using this function keyword include the following:

```
"ASCALE",multi@28           : ScalingInc    ; Seimens style or
```

Function code: 101
Group code: 100000
Scope: Modal
RVP codes: 7xxx

SetABSenseCW

Description: Maps the cw direction of rotary axis interpolation
Function code: 400

Group code: 100000
RVP codes

SetABSenseCCW

Description: Maps the ccw direction of rotary axis interpolation
Function code: 401
Group code: 100000
RVP codes

SelectPlaneXY

Description: Maps XY plane selection for circular interpolation
Tech Tips: Selects XY plane for circular motion. Examples using this function keyword include the following:

```
“G17”       : SelectPlaneXY     ; Fanuc style
“(17)”      : SelectPlaneXY     ; Dynapath style
"PLANE~XY"  : SelectPlaneXY     ; Anilam Style
"PLANEXY"   : SelectPlaneXY
```

Function code: 200
Group code: 100000
Scope: Modal

RVP codes: 71xx

SelectPlaneYZ

Description: Maps YZ plane selection for circular interpolation
Tech Tips: Selects YZ plane for circular motion. Examples using this function keyword include the following:

```
“G19”       : SelectPlaneYZ     ; Fanuc style
“(19)”      : SelectPlaneYZ     ; Dynapath conversational style
"PLANE~YZ"  : SelectPlaneYZ     ; Anilam Style
"PLANEYZ"   : SelectPlaneYZ
```

Function code: 202
Group code: 100000
Scope: Modal
RVP codes: 71xx

SelectPlaneZX

Description: Maps ZX plane selection for circular interpolation
Tech Tips: Selects ZX plane for circular motion. Examples using this function keyword include the following:

```
“G18”       : SelectPlaneZX     ; Fanuc style
“(18)”      : SelectPlaneZX     ; Dynapath conversational style
"PLANE~ZX"  : SelectPlaneZX     ; Anilam Style
"PLANEZX"   : SelectPlaneZX
```

Function code: 201
Group code: 100000
Scope: Modal
RVP codes: 71xx

SelectPlane

Description: Maps plane selection for circular interpolation
Tech Tips: Selects XY, YZ or ZX plane for circular motion. Examples using this function keyword include the following:

"PLANE",int : SelectPlane

Function code: 200
Group code: 100000
Scope: Modal

RVP codes: 71xx

3.2.5.2 GROUP: MOVETO

The moveto function group includes all function keywords that map different linear and rotary axis of motion including X, Y, Z, I, J, K, R, A, B and several others. All moveto function keywords are listed below with a short description:

IncMoveX ; Maps incremental linear horizontal or X axis of motion
IncMoveY ; Maps incremental linear vertical or Y axis of motion
IncMoveZ ; Maps incremental linear depth or Z axis of motion
InsertMove ; Maps lathe insert or chamfer motion within linear motion
Move4thAxis ; Maps the primary rotary axis
Move5thAxis ; Maps the secondary rotary axis
MoveA ; Maps the rotary horizontal or A axis of motion
MoveB ; Maps the rotary vertical or B axis of motion
MoveFromXYZ ; Maps APTCL style FROM statements
MoveInMCHCoord ; Maps tool motion in machine coordinates
MoveIJKUVW ; Maps APTCL style CIRCLE statements
MoveI ; Maps the circle center horizontal or X axis of motion
MoveJ ; Maps the circle center horizontal or Y axis of motion
MoveK ; Maps the circle center depth or Z axis of motion
MoveR ; Maps the circle radius motion
MoveX ; Maps the linear horizontal or X axis of motion
MoveX- ; Maps an inverted linear horizontal or X axis of motion
MoveY ; Maps the linear vertical or Y axis of motion
MoveY- ; Maps an inverted vertical or Y axis of motion
MoveZ ; Maps the linear depth or Z axis of motion
MoveZ- ; Maps an inverted depth or Z axis of motion
MoveXYZIJK ; Maps APTCL style 4 and 5 axis GOTO statements
MoveXYZUVW ; Maps APTCL style 4 and 5 axis GOTO statements
PolarAngle ; Maps the polar angle for polar interpolation
PolarRadius ; Maps the polar radius for polar interpolation
ProcessParamCycle ; Maps a modal parameter for canned cycles

IncMoveX

Description: Maps incremental linear horizontal or X axis of motion
Tech Tips: Selects the incremental X axis of motion. On most CNC controls this is the U register. Examples using this function keyword include the following:

"U",real : IncMoveX ; Fanuc style
"IX",real : IncMoveX ; Heidenhain style

Function code: 1
Group code: 100100

Scope: Non Modal
RVP codes: 7000, 7001, 7172, 7173, 7182, 7183, 7192, 7193,7500, 7501, 7600, 7601

IncMoveY

Description: Maps incremental linear vertical or Y axis of motion
Tech Tips: Selects the incremental Y axis of motion. On most CNC controls this is the V register. Examples using this function keyword include the following:

```
"V",real        : IncMoveY    ; Fanuc style
"IY",real       : IncMoveY    ; Heidenhain style
```

Function code: 2
Group code: 100100
Scope: Non Modal
RVP codes: 7000, 7001, 7172, 7173, 7182, 7183, 7192, 7193,7500, 7501, 7600, 7601

IncMoveZ

Description: Maps incremental linear depth or Z axis of motion
Tech Tips: Selects the incremental Z axis of motion. On most CNC controls this is the W register. Examples using this function keyword include the following:

```
"W",real        : IncMoveZ    ; Fanuc style
"IZ",real       : IncMoveZ    ; Heidenhain style
```

Function code: 3
Group code: 100100
Scope: Non Modal
RVP codes: 7000, 7001, 7172, 7173, 7182, 7183, 7192, 7193,7500, 7501, 7600, 7601

InsertMove

Description: Maps lathe insert or chamfer motion
Tech Tips: Selects the lathe insert or chamfer motion when used on linear interpolation or "G1" moves. On most CNC controls this is the C register. Examples using this function keyword include the following:

```
"C",real        : InsertMove  ; Fanuc style
```

Function code: 17
Group code: 100100
Scope: Non Modal
RVP codes: 7000, 7001, 7172, 7173, 7182, 7183, 7192, 7193

Move4thAxis

Description: Maps the primary rotary axis of motion
Tech Tip: Selects the primary rotary axis as specified within the machine definition. On most CNC controls this is the A register. Examples using this function keyword include the following:

```
"A",real        : Move4thAxis ; Standard style or
"A",expression  : Move4thAxis ; Variables with math functions
```

Function code: 10
Group code: 100100
Scope: Non Modal
RVP codes: 7172, 7173, 7182, 7183, 7192, 7193

Move5thAxis

Description: Maps the secondary rotary axis of motion
Tech Tip: Selects the secondary rotary axis as specified within the machine definition. On most CNC controls this is the B register. Examples using this function keyword include the following:

"B",real : Move5thAxis ; Standard style or
"B",expression : Move5thAxis ; Variables with math functions

Function code: 11
Group code: 100100
Scope: Non Modal
RVP codes: 7172, 7173, 7182, 7183, 7192, 7193

MoveA

Description: Maps the horizontal rotary axis of motion
Tech Tip: Selects the horizontal rotary axis. On most CNC controls this is the A register. Examples using this function keyword include the following:

"A",real : MoveA ; Standard style or
"A",expression : MoveA ; Variables with math functions

Function code: 10
Group code: 100100
Scope: Non Modal
RVP codes: 7172, 7173, 7182, 7183, 7192, 7193

MoveB

Description: Maps the vertical rotary axis of motion
Tech Tips: Selects the vertical rotary axis. On most CNC controls this is the B register. Examples using this function keyword include the following:

"B",real : MoveB ; Standard style or
"B",expression : MoveB ; Variables with math functions

Function code: 11
Group code: 100100
Scope: Non Modal
RVP codes: 7172, 7173, 7182, 7183, 7192, 7193

MoveFromXYZ

Description: Maps APTCL style FROM statements
Tech Tips: Selects reference positions within APTCL style tool motion. Within most APTCL file formats this is the FROM statement followed by X, Y, and Z values. Examples using this function keyword include the following:

"FROM",multi@9,multigstrictord : MoveFromXYZ ; Standard style

Function code: 13
Group code: 100100
Scope: Non Modal
RVP codes: 7172, 7173, 7182, 7183, 7192, 7193

MoveInMCHCoord

Description: Maps tool motion in machine coordinates
Tech Tips: Selects tool motion within machine coordinates. Within most CNCs this is a G53 code followed by X, Y and Z values. Examples using this function keyword include the following:

```
"G53",multi@21              : MoveInMCHCoord          ; Fanuc style
```

Function code: 12
Group code: 100100
Scope: Non Modal
RVP codes: 7172, 7173, 7182, 7183, 7192, 7193

MoveIJKUVW

Description: Maps APTCL style CIRCLE statements
Tech Tips: Selects multi-axis APTCL style tool motion. Within most APTCL file formats this is the CIRCLE statement followed by six numbers. The first three numbers refer to the circle's center coordinates. The last three numbers refer to the circle's normals. Examples using this function keyword include the following:

```
"CIRCLE",multi@10,multigstrictord  : MoveIJKUVW          ; Standard style
```

Function code: 12
Group code: 100100
Scope: Non Modal
RVP codes: 7172, 7173, 7182, 7183, 7192, 7193

MoveI

Description: Maps the circle center horizontal or X axis of motion
Tech Tips: Selects the circle center horizontal axis. On most CNC controls this is the I register. MoveI supports both incremental and absolute coordinates by setting the arccenter_absolute parameter. Examples using this function keyword include the following:

```
"I",real                    : MoveI       ; Standard style or
"I",expression              : MoveI       ; Variables with math functions
```

Function code: 4
Group code: 100100
Scope: Non Modal
RVP codes: 7172, 7173, 7182, 7183, 7192, 7193, 7600, 7601

MoveJ

Description: Maps the circle center vertical or Y axis of motion
Tech Tips: Selects the circle center vertical axis. On most CNC controls this is the J register. MoveJ supports both incremental and absolute coordinates by setting the arccenter_absolute parameter. Examples using this function keyword include the following:

```
"J",real                    : MoveJ       ; Standard style or
"J",expression              : MoveJ       ; Variables with math functions
```

Function code: 5
Group code: 100100

Scope: Non Modal
RVP codes: 7172, 7173, 7182, 7183, 7192, 7193, 7600, 7601

MoveK

Description: Maps the circle center depth or Z axis of motion
Tech Tips: Selects the circle center depth axis. On most CNC controls this is the K register. MoveK supports both incremental and absolute coordinates by setting the arccenter_absolute parameter. Examples using this function keyword include the following:

```
"K",real          : MoveK     ; Standard style or
"K",expression    : MoveK     ; Variables with math functions
```

Function code: 6
Group code: 100100
Scope: Non Modal
RVP codes: 7172, 7173, 7182, 7183, 7192, 7193, 7600, 7601

MoveR

Description: Maps the circle radius
Tech Tips: Selects the circle radius. On most CNC controls this is the R register. Examples using this function keyword include the following:

```
"R",real          : MoveR     ; Fanuc style or
"R",expression    : MoveR     ; Variables with math functions
"L",real          : MoveR     ; Okuma style or
"CR=",real        : MoveR     ; Seimens style
```

Function code: 7
Group code: 100100
Scope: Non Modal

RVP codes: 7172, 7173, 7182, 7183, 7192, 7193

MoveX

Description: Maps the linear horizontal or X axis of motion
Tech Tips: Selects the X axis of motion. On most CNC controls this is the X register. Examples using this function keyword include the following:

```
"X",real          : MoveX     ; Standard style or
"X",expression    : MoveX     ; Variables with math functions
```

Function code: 1
Group code: 100100
Scope: Non Modal
RVP codes: 7000, 7001, 7172, 7173, 7182, 7183, 7192, 7193,7500, 7501, 7600, 7601

MoveX-

Description: Maps an inverted linear horizontal or X axis of motion
Tech Tips: Selects the X axis of motion and flips positive and negative direction. On most CNC controls this is the X register. Examples using this function keyword include the following:

```
"X",real          : MoveX-    ; Standard style or
"X",expression    : MoveX-    ; Variables with math functions
```

Function code: 1
Group code: 100100
Scope: Non Modal
RVP codes: 7000, 7001, 7172, 7173, 7182, 7183, 7192, 7193,7500, 7501, 7600, 7601

MoveY

Description: Maps the linear vertical or Y axis of motion
Tech Tips: Selects the Y axis of motion. On most CNC controls this is the Y register. Examples using this function keyword include the following:

```
"Y",real        : MoveY     ; Standard style or
"Y",expression  : MoveY     ; Variables with math functions
```

Function code: 2
Group code: 100100
Scope: Non Modal
RVP codes: 7000, 7001, 7172, 7173, 7182, 7183, 7192, 7193,7500, 7501, 7600, 7601

MoveY-

Description: Maps an inverted linear vertical or Y axis of motion
Tech Tips: Selects the Y axis of motion and flips positive and negative direction. On most CNC controls this is the Y register. Examples using this function keyword include the following:

```
"Y",real        : MoveY-    ; Standard style or
"Y",expression  : MoveY-    ; Variables with math functions
```

Function code: 2
Group code: 100100
Scope: Non Modal
RVP codes: 7000, 7001, 7172, 7173, 7182, 7183, 7192, 7193,7500, 7501, 7600, 7601

MoveZ

Description: Maps the linear depth or Z axis of motion
Tech Tips: Selects the Z axis of motion. On most CNC controls this is the Z register. Examples using this function keyword include the following:

```
"Z",real        : MoveZ     ; Standard style or
"Z",expression  : MoveZ     ; Variables with math functions
```

Function code: 3
Group code: 100100
Scope: Non Modal
RVP codes: 7000, 7001, 7172, 7173, 7182, 7183, 7192, 7193,7500, 7501, 7600, 7601

MoveZ-

Description: Maps an inverted linear depth or Z axis of motion
Tech Tips: Selects the Z axis of motion and flips positive and negative direction. On most CNC controls this is the Z register. Examples using this function keyword include the following:

```
"Z",real        : MoveZ-    ; Standard style or
"Z",expression  : MoveZ-    ; Variables with math functions
```

Function code: 3
Group code: 100100
Scope: Non Modal
RVP codes: 7000, 7001, 7172, 7173, 7182, 7183, 7192, 7193,7500, 7501, 7600, 7601

MoveXYZIJK

Description: Maps APTCL style 4 and 5 axis GOTO statements
Tech Tips: Selects multi-axis APTCL style tool motion. Within most APTCL file formats this is the GOTO statement followed by X, Y, Z, I, J, and K values. I, J, and K values refer to tool axis normals and do not indicate circle center coordinates. Examples using this function keyword include the following:

"GOTO",multi@1,multigstrictord : MoveXYZIJK ; Standard style

Function code: 9
Group code: 100100
Scope: Non Modal
RVP codes: 7172, 7173, 7182, 7183, 7192, 7193

MoveXYZUVW

Description: Maps APTCL style 4 and 5 axis GOTO statements
Tech Tips: Selects multi-axis APTCL style tool motion. Within most APTCL file formats this is the GOTO statement followed by X, Y, Z, I, J, and K values. I, J, and K values refer to tool axis normals and do not indicate circle center coordinates. Examples using this function keyword include the following:

"GOTO",multi@1,multigstrictord : MoveXYZUVW ; Standard style

NOTE: MoveXYZUVW has been replaced with MoveXYZIJK. MoveXYZUVW can still be used but should be replaced with the newer function keyword MoveXYZIJK.

Function code: 9
Group code: 100100
Scope: Non Modal
RVP codes: 7172, 7173, 7182, 7183, 7192, 7193

PolarAngle

Description: Maps the polar angle for polar interpolation
Tech Tips: Selects the polar angle for polar interpolation. Examples using this function keyword include the following:

"AP=",real : PolarAngle ; Seimens style

Function code: 11
Group code: 100100
Scope: Non Modal
RVP codes: 7172, 7173, 7182, 7183, 7192, 7193

PolarRadius

Description: Maps the polar radius for polar interpolation
Tech Tips: Selects the polar radius for polar interpolation. Examples using this function keyword include the following:

"RP=",real : PolarRadius ; Seimens style

Function code: 11
Group code: 100100
Scope: Non Modal
RVP codes: 7172, 7173, 7182, 7183, 7192, 7193

ProcessParamCycle

Description: Maps a modal parameter for canned cycles
Tech Tips: Selects a modal parameter for canned cycles. On most CNC controls this is the Q register. Examples using this function keyword include the following:

“Q”,real : ProcessParamCycle ; Fanuc style

NOTE: Within the CNC program if different Q values are specified in any secondary hole locations this keyword is required. By utilizing this keyword the proper verification, simulation and cycle time calculations will be calculated. Without this keyword the original Q value within the initial hole cycle will be used for all secondary holes.

Function code: 11
Group code: 100100
Scope: Non Modal
RVP codes: 7xxx

3.2.5.3 GROUP: TOOLCHANGE

The toolchange function group includes all function keywords that create tool definitions, load tools, and select tools. All toolchange function keywords are listed below with a short description:

CreateLatheTool ; Maps turning tool sizes and shape definitions
CreateMillTool ; Maps milling tool sizes and shape definitions
GoHome ; Maps returning the tool to the machine home position
LoadTool ; Maps loading the tool to the first spindle
LoadTool2 ; Maps loading the tool to the second spindle
LoadTool3 ; Maps loading the tool to the third spindle
LoadTool4 ; Maps loading the tool to the fourth spindle
SelectTool ; Maps selecting the tool from the tool carousel
SelectAndLoadTool ; Maps selecting and loading the tool from the tool changer
SetTLBFileName ; Maps loading tool kits or libraries

CreateLatheTool

Description: Maps turning tool sizes and shape definitions
Tech Tips: Selects lathe insert and optional tool holder definitions within comments. Examples using this function keyword include the following:

“LTOOL”,multi@12 : CreateLatheTool ; Fanuc style

NOTE: Refer to the Register Keywords section for more details.

Function code: 4
Group code: 100200
Scope: Non Modal
RVP codes: 20xxx

CreateMillTool

Description: Maps milling tool sizes and shape definitions
Tech Tips: Selects milling tool shapes and optional tool holder definitions within comments. Examples using this function keyword include the following:

"MTOOL",multi@12 : CreateMillTool ; Fanuc style

NOTE: Refer to the Register Keywords section for more details.

Function code: 4
Group code: 100200
Scope: Non Modal
RVP codes: 20xxx

GoHome

Description: Maps returning the tool to the home position
Tech Tips: Maps returning the tool to the machine's home position. Often this can be used to perform a full retract. Examples using this function keyword include the following:

"M26" : GoHome ; Cincinnati 2100 style

Function code: 2
Group code: 100200
RVP codes: 20xxx

LoadTool

Description: Maps loading the tool to the first spindle
Tech Tips: Loads the previously selected tool. On most CNC controls this is the M6 or M06 code. Examples using this function keyword include the following:

"M6" : LoadTool ; Fanuc style
"M06" : LoadTool ; Multiple definitions are supported

NOTE: LoadTool assumes that a SelectTool is also required by the CNC control.

Function code: 2
Group code: 100200
RVP codes: 20xxx

LoadTool2

Description: Maps loading the tool to the second spindle
Tech Tips: Loads the previously selected tool. On most CNC controls this is the M code. Examples using this function keyword include the following:

"M26" : LoadTool2 ; Multiple definitions are supported

NOTE: LoadTool2 assumes that a SelectTool is also required by the CNC control.

Function code: 2
Group code: 100200
RVP codes: 20xxx

LoadTool3

Description: Maps loading the tool to the third spindle

Tech Tips: Loads the previously selected tool. On most CNC controls this is the M code. Examples using this function keyword include the following:

"M36" : LoadTool3 ; Multiple definitions are supported

NOTE: LoadTool3 assumes that a SelectTool is also required by the CNC control.

Function code: 2
Group code: 100200
RVP codes: 20xxx

LoadTool4

Description: Maps loading the tool to the fourth spindle
Tech Tips: Loads the previously selected tool. On most CNC controls this is the M code. Examples using this function keyword include the following:

"M26" : LoadTool4 ; Multiple definitions are supported

NOTE: LoadTool4 assumes that a SelectTool is also required by the CNC control.

Function code: 2
Group code: 100200
RVP codes: 20xxx

SelectTool

Description: Maps selecting the tool from the tool carousel
Tech Tips: Selects the next tool to be loaded. On most CNC controls this is the T register. Examples using this function keyword include the following:

"T",int : SelectTool ; Standard style

NOTE: SelectTool assumes that a LoadTool is also required by the CNC control.

Function code: 1
Group code: 100200
RVP codes

SelectAndLoadTool

Description: Maps selecting and loading the tool from the tool changer
Tech Tips: Selects and loads the tool with a single command. On most CNC lathes and routers this is the T register and no M6 or equivalent is used. Examples using this function keyword include the following:

"T",int : SelectandLoadTool ; Standard style

NOTE: SelectAndLoadTool assumes that a separate LoadTool and SelectTool is not required by the CNC control.

Function code: 3
Group code: 100200
RVP codes: 20xxx

SetTLBFileName

Description: Maps loading tool kits or libraries

Tech Tips: Selects loading an entire tool kit or tool library file with a single command. Example reverse posts specify this value by including a master RP* file. Examples using this function keyword include the following:

“TLB_FILE=”,multi@101 : SetTLBFileName ; Predator style

Function code: 8
Group code: 100200
RVP codes: 20xxx

3.2.5.4 GROUP: CANNEDCYCLES

The cannedcycles function group includes all function keywords that map different types of canned cycle tool motion including holes, circles, arcs, lines and bores. Additional canned cycle function keywords include canceling and returning to rapid planes. All cannedcycles function keywords are listed below with a short description:

ArcCycle ; Reserved for future use and has not been implemented
ArcHoleCycle ; Maps arc, circle or bolt hole pattern definitions
BoreCycle ; Maps bore cycle definition and has not been implemented
CallCycle ; Maps calling the active cycle
CancelCannedCycle ; Maps canceling the current active canned cycle
CancelPatternCycle ; Maps canceling the current active pattern cycle
CircleCycle ; Reserved for future use and has not been implemented
CircPocketCycleDef ; Maps circular pocket cycle definitions
CircPocketCycleCW ; Maps clockwise circular pocket cycle definitions
CircPocketCycleCCW ; Maps counter clockwise circular pocket cycle definitions
CircPocketCycleDef ; Maps circular pocket cycle definition
CycleDepth ; Maps depth of cycle definition
DefCycle ; Maps multiple line cycle definition
DrillCycle ; Maps drilling, pecking, tapping, and boring cycles
LineCycle ; Maps line pattern cycle definition
LinearHoleCycle ; Maps linear hole cycle definition
MatrixHoleCycle ; Maps matrix or grid hole cycle definition
PatternCycle ; Maps pattern cycle definition
ReturnToInitial ; Maps returning to initial rapid plane in a canned cycle
ReturnToReference ; Maps returning to reference rapid plane in a canned cycle
RectInFaceCycle ; Maps rectangular in style facing cycle definition
RectOutFaceCycle ; Maps rectangular out style facing cycle definition
RectPocketCycle ; Maps rectangular pocket cycle definition
RectPocketCycleDef ; Maps rectangular pocket cycle definition
SlotPocketCycle ; Maps slot pocket cycle definition

ArcCycle

Description: Reserved for future use and has not been implemented
Function code: ARCCYCLE
Group code: 100250
RVP codes

NOTE: _ArcCycle is also supported for backwards compatibility.

ArcHoleCycle

Description: Maps arc, circle or bolt hole pattern definitions
Tech Tips: Selects calling the active cycle. Examples using this function keyword include the following:

"G70",multi@70 : ArcHoleCycle ; Haas style or
"HOLES2",multi@40 : ArcHoleCycle ; Siemens style

Function code: 1
Group code: 100250
Scope: Modal
RVP codes

NOTE: _ArcHoleCycle is also supported for backwards compatibility.

BoreCycle

Description: Reserved for future use and has not been implemented
Function code: BORECYCLE
Group code: 100250
RVP codes

NOTE: _BoreCycle is also supported for backwards compatibility.

CircleCycle

Description: Reserved for future use and has not been implemented
Function code: CIRCLECYCLE
Group code: 100250
RVP codes

NOTE: _CircleCycle is also supported for backwards compatibility.

CallCycle

Description: Maps calling the active cycle
Tech Tips: Selects calling the active cycle. Examples using this function keyword include the following:

"CYCL*CALL" : CallCycle ; Heidenhain style

Function code: 1
Group code: 100250
Scope: Modal
RVP codes

NOTE: _CallCycle is also supported for backwards compatibility.

CancelCannedCycle

Description: Maps canceling the current active canned cycle
Tech Tips: Cancels or specifies the end of the current canned cycle. On most CNC controls this is a G80 code. Examples using this function keyword include the following:

"G80" : CancelCannedCycle ; Fanuc style or
"DrillOff" : CancelCannedCycle ; Anilam style

Function code: 1
Group code: 100250
Scope: Modal
RVP codes

CancelPatternCycle

Description: Maps canceling the current active pattern cycle
Tech Tips: Cancels or specifies the end of the current pattern cycle. Examples using this function keyword include the following:

“G37” : CancelCannedCycle ; Vickers 2100 style

Function code: 1
Group code: 100250
Scope: Modal
RVP codes

CircPocketCycle

Description: Maps circular pocket cycle definitions
Tech Tips: Specifies or defines a circular pocketing cycle. Examples using this function keyword include the following:

“POCKET2”,multi@35 : CircPocketCycle( ) ; Siemens 840 style or
"L9801",multi@17 : CircPocketCycle( ) ; Fadal style or
"L9901",multi@17 : CircPocketCycle( )
“G177”,multi@3 : CircPocketCycle( ) ; Bridgeport style

Function code: 1
Group code: 100250
Scope: Modal
RVP codes

NOTE: _CircPocketCycle is also supported for backwards compatibility.

CircPocketCycleCW

Description: Maps clockwise circular pocket cycle definitions
Tech Tips: Specifies or defines a clockwise circular pocketing cycle. Examples using this function keyword include the following:

"G12",multi@16 : CircPocketCycleCW ; Haas style

Function code: 1
Group code: 100250
Scope: Modal
RVP codes

NOTE: _CircPocketCycleCW is also supported for backwards compatibility.

CircPocketCycleCCW

Description: Maps counter clockwise circular pocket cycle definitions
Tech Tips: Specifies or defines a counter clockwise circular pocketing cycle. Examples using this function keyword include the following:

"G13",multi@16 : CircPocketCycleCCW ;Haas style

Function code: 1
Group code: 100250
Scope: Modal
RVP codes

NOTE: _CircPocketCycleCCW is also supported for backwards compatibility.

CirclePocketCycleDef

Description: Maps circular pocket cycle definitions
Tech Tips: Selects and defines circular pocket cycles. Examples using this function keyword include the following:

```
"CYCL*DEF*5.0",multi@22      : CircPocketCycleDef( )        ; Heidenhain style
"CYCL*DEF*214",multi@23      : CircPocketCycleDef( )
```

Function code: 1
Group code: 100250
Scope: Modal
RVP codes

NOTE: _CirclePocketCycleDef is also supported for backwards compatibility.

CycleDepth

Description: Maps the depth of a canned cycle
Tech Tips: Selects the depth of a canned cycle. Examples using this function keyword include the following:

```
"G79",multi@38          : CycleDepth( )          ; Siemens 840 style
```

Function code: 1
Group code: 100250
Scope: Modal
RVP codes

DefCycle

Description: Maps multiple line cycle definition
Tech Tips: Selects the definition of a multiple line cycle. Examples using this function keyword include the following:

```
"CYCL*DEF*1.0",multi@9                    : DefCycle(peck)        ; Heidenhain style
"CYCL*DEF*1.1",multi@10,multigstrictord   : DefCycle(peck)
"CYCL*DEF*1.2",multi@11,multigstrictord   : DefCycle(peck)
"CYCL*DEF*1.3",multi@12,multigstrictord   : DefCycle(peck)
"CYCL*DEF*1.4",multi@13,multigstrictord   : DefCycle(peck)
"CYCL*DEF*1.5"                            : DefCycle(peck)
"CYCL*DEF*2.0",multi@9                    : DefCycle( )
"CYCL*DEF*2.1",multi@10,multigstrictord   : DefCycle( )
"CYCL*DEF*2.2",multi@11,multigstrictord   : DefCycle( )
"CYCL*DEF*2.3",multi@13,multigstrictord   : DefCycle( )
"CYCL*DEF*2.4"                            : DefCycle( )
"V50",multi@12                            : DefCycle(peck)        ; Cincinnati style
"V51",multi@13                            : DefCycle(peck)
```

Function code: 1
Group code: 100250
Scope: Modal
RVP codes

NOTE: _DefCycle is also supported for backwards compatibility.

DrillCycle

Description: Maps drilling, pecking, tapping and boring cycles
Tech Tips: Selects all drill cycles. On most CNC controls these are G81 – G89 codes. Examples using this function keyword include the following:

```
"G73",multi@2          : DrillCycle (peck)                          ; Fanuc style
"G74",multi@2          : DrillCycle (bottomCW retractfeed)
"G76",multi@2          : DrillCycle (oriented_bottomstop)
"G81",multi@2          : DrillCycle ( )
"G82",multi@2          : DrillCycle (dwell)
"G83",multi@2          : DrillCycle (peck dwell)
"G84",multi@2          : DrillCycle (bottomCW dwell retractfeed)
"G85",multi@2          : DrillCycle (retractfeed)
"G86",multi@2          : DrillCycle (bottomstop)
"G87",multi@2          : DrillCycle (bottomstop manualfeed)
"G88",multi@2          : DrillCycle (dwell bottomstop)
"G89",multi@2          : DrillCycle (dwell retractfeed)
“G200”,multi@2         : DrillCycle ( )                             ; Heidenhain style or
“CYCLE81”,multi@37     : DrillCycle ( )                             ; Siemens style or
“BasicDrill”,multi@2   : DrillCycle ( )                             ; Anilam style
"PeckDrill",multi@2    : DrillCycle (peck)
"Tapping",multi@2      : DrillCycle (bottomCW dwell retractfeed)
```

NOTE: By combining the drill cycle parameters additional custom drill cycles can be created.

Function code: DRILLCYCLE
Group code: 100250
Parameters: peck dwell retractfeed manualfeed bottomCW bottomstop oriented_bottomstop
RVP codes

NOTE: _DrillCycle is also supported for backwards compatibility.

LineCycle

Description: Reserved for future use and has not been implemented
Function code: LINECYCLE
Group code: 100250
RVP codes

LinearHoleCycle

Description: Maps a linear canned hole cycle definition
Tech Tips: Specifies a linear hole cycle. Examples using this function keyword include the following:

```
"G72",multi@7          : LinearHoleCycle( )     ; Haas style or
“HOLES1”,multi@20      : LinearHoleCycle( )     ; Siemens 840 style
```

Function code: 1
Group code: 100250
Scope: Modal
RVP codes

MatrixHoleCycle

Description: Maps a matrix canned hole cycle definition

Tech Tips: Specifies a matrix or grid hole cycle. Examples using this function keyword include the following:

“CYCLE801”,multi@41 : MatrixHoleCycle ( ) ; Siemens 840 style

Function code: 1
Group code: 100250
Scope: Modal
RVP codes

PatternCycle

Description: Maps pattern cycle definition
Tech Tips: Specifies or defines a pattern cycle. Examples using this function keyword include the following:

"G70",multi@17 : PatternCycle( ) ; Haas style or
"G71",multi@17 : PatternCycle( )
“G39”,multi@14 : PatternCycle( ) ; Vickers 2100 style

Function code: 1
Group code: 100250
Scope: Modal
RVP codes

NOTE: _PatternCycle is also supported for backwards compatibility.

RectInFaceCycle

Description: Maps rectangular in facing cycles
Tech Tips: Specifies or defines a rectangular in facing cycle. Examples using this function keyword include the following:

“G174”,multi@4 : RectInFaceCycle ( ) ; Bridgeport style

Function code: 1
Group code: 100250
Scope: Modal
RVP codes

RectOutFaceCycle

Description: Maps rectangular out facing cycles
Tech Tips: Specifies or defines a rectangular out facing cycle. Examples using this function keyword include the following:

“CYCLE71”,multi@39 : RectOutFaceCycle ( ) ; Siemens 840 style or
“G173”,multi@4 : RectOutFaceCycle ( ) ; Bridgeport style

Function code: 1
Group code: 100250
Scope: Modal
RVP codes

RectPocketCycle

Description: Maps rectangular pocket cycle definitions
Tech Tips: Specifies or defines a rectangular pocketing cycle. Examples using this function keyword include the following:

```
"POCKET1",multi@34   : RectPocketCycle( )        ; Siemens 840 style or
"L9601",multi@16     : RectPocketCycle( )        ; Fadal style or
"L9701",multi@16     : RectPocketCycle( )
"G172",multi@2       : RectPocketCycle( )        ; Bridgeport style
```

Function code: 1
Group code: 100250
Scope: Modal
RVP codes

NOTE: _RectPocketCycle is also supported for backwards compatibility.

RectPocketCycleDef

Description: Maps rectangular pocket cycle definitions
Tech Tips: Specifies or defines a rectangular pocketing cycle. Heidenhain examples using this function keyword include the following:

```
"CYCL*DEF*4.0",multi@22     : RectPocketCycleDef( )     ;(Rectangular Pocket)
"CYCL*DEF*4.1",multi@22     : RectPocketCycleDef( )     ;(Rectangular Pocket)
"CYCL*DEF*4.2",multi@22     : RectPocketCycleDef( )     ;(Rectangular Pocket)
"CYCL*DEF*4.3",multi@22     : RectPocketCycleDef( )     ;(Rectangular Pocket)
"CYCL*DEF*4.4",multi@22     : RectPocketCycleDef( )     ;(Rectangular Pocket)
"CYCL*DEF*4.5",multi@22     : RectPocketCycleDef( )     ;(Rectangular Pocket)
"CYCL*DEF*4.6",multi@22     : RectPocketCycleDef( )     ;(Rectangular Pocket)
"CYCL*DEF*212",multi@23     : RectPocketCycleDef( )     ;(Rectangular Pocket)
```

Function code: 1
Group code: 100250
Scope: Modal
RVP codes

NOTE: _RectPocketCycleDef is also supported for backwards compatibility.

ReturnToInitial

Description: Maps returning to initial rapid plane in a canned cycle
Tech Tips: Returns to initial rapid plane in current cycle. On most CNC controls this is a G98 code. An Example using this function keyword include the following:

```
“G98”        : ReturnToInitial        ; Fanuc style
```

Function code: 90
Group code: 100250
RVP codes

ReturnToReference

Description: Maps returning to reference rapid plane in a canned cycle
Tech Tips: Returns to reference rapid plane in current cycle. On most CNC controls this is a G99 code. An Example using this function keyword include the following:

```
“G99”        : ReturnToReference     ; Fanuc style
```

Function code: 91
Group code: 100250
RVP codes

SlotPocketCycle

Description: Maps slot pocket cycle definitions
Tech Tips: Specifies or defines a slot pocketing cycle. Examples using this function keyword include the following:

"G88.1",multi@7 : SlotPocketCycle ; Allen Bradley style

Function code: 1
Group code: 100250
Scope: Modal
RVP codes

NOTE: _SlotPocketCycle is also supported for backwards compatibility.

For more information refer to section 3.2.6

3.2.5.5 GROUP: MISCELLANEOUS

The miscellaneous function group includes all function keywords that map standard M codes including spindle direction, spindle stop, program end, optional stop, and coolant status. All miscellaneous function keywords are listed below with a short description:

AirBlast ; Maps enabling/disabling air blasts
ChangePallet ; Maps changing pallets
CoolantON ; Maps coolant on
CoolantOFF ; Maps coolant off
CtantSurfSpeed ; Maps constant surface speed
CtantSurfSpeedCancel ; Maps canceling constant surface speed
DisableRotaryComp ; Maps disabling 5 axis rotary tool compensation
EnableRotaryComp ; Maps enabling 5 axis rotary tool compensation
HSpindleStop ; Maps horizontal spindle stop
LineNumber ; Maps line or sequence numbers
LoadMachine ; Maps enabling a second machine
MaxSpindleSpeed ; Maps maximum spindle speed
OptStop ; Maps optional stop
Pause0 ; Maps program stop
Pause1 ; Maps optional stop
ProgramEnd ; Maps end of program
RestoreMachine ; Maps disabling a second machine
SetMachineDatum ; Maps setting machine datum
SetMCHFileName ; Maps setting machine filename
SetOrigRelocation ; Maps setting origin relocation
SpindleCW ; Maps clockwise spindle motion
SpindleCCW ; Maps counter clockwise spindle motion
SpindleSpeed ; Maps spindle speed
SpindleStop ; Maps the spindle stop
SpindleStop2 ; Maps a second spindle stop
SpindleStop3 ; Maps a third spindle stop
SpindleStop4 ; Maps a fourth spindle stop
Stop ; Maps program stop
ToolClamp ; Maps tool clamp/unclamp
VSpindleStop ; Maps vertical spindle stop
ZHome ; Maps rapiding the tool home

The miscellaneous function group includes optional parameters that can be combined with the primary keyword to assign advanced miscellaneous codes. All optional miscellaneous parameters are listed below with a short description:

Coolant_Off ; Maps coolant off
End_Program ; Maps program end
FloodCoolant_On ; Maps flood coolant on
MistCoolant_On ; Maps mist coolant on
Opt_Stop ; Maps optional stop
Program_Stop ; Maps program stop
Spindle_Cw ; Maps spindle clockwise
Spindle_Ccw ; Maps spindle counter clockwise
Spindle_Off ; Maps spindle off
Tool_Change ; Maps tool change

One or more of the above miscellaneous parameters can be added to any of the following miscellaneous keywords. For example:

"M88" : SpindleCW (FloodCoolant_On) ; Haas advanced M codes
"M89" : Spindlestop (Coolant_Off)

On a Haas a M88 turn the spindle clockwise and also turns on flood coolant while a M89 stops the spindle and also turns off coolant. Several of the following miscellaneous keywords have examples with miscellaneous parameters.

AirBlast

Description: Maps enabling/disabling air blasts
Tech Tips: Selects and defines enabling and disabling air blast. On most CNC controls this is a M code. Although not simulated by Predator Virtual CNC this is used for accurate cycle time calculation. Examples using this function keyword include the following:

“M50” : AirBlast ; Varies per machine builder
“M87” : AirBlast ; Siemens style

NOTE: The time for an air blast can be configured within the machine definition.

Function code: 4
Group code: 100300
RVP codes

ChangePallet

Description: Maps changing pallets
Tech Tips: Selects and defines changing pallets. On most CNC controls this is a M code. Although not simulated by Predator Virtual CNC this is used for accurate cycle time calculation. Examples using this function keyword include the following:

“M64” : ChangePallet ; Varies per machine builder
“M36” : ChangePallet ; Siemens style

NOTE: The time for changing pallets can be configured within the machine definition.

Function code: 4
Group code: 100300
RVP codes

CoolantON

Description: Maps coolant on
Tech Tips: Selects coolant on. On most CNC controls this is a M7 code. Examples using this function keyword include the following:

```
"M7"        : CoolantON        ; Fanuc style
"M07"       : CoolantON        ; Multiple definitions are supported
"(9)M8"     : CoolantON        ; Dynapath style
"MCode~8"   : CoolantON        ; Anilam style
```

Function code: 4
Group code: 100300
RVP codes

CoolantOFF

Description: Maps coolant off
Tech Tips: Selects coolant off. On most CNC controls this is a M9 code. Examples using this function keyword include the following:

```
"M9"        : CoolantOFF       ; Fanuc style
"M09"       : CoolantOFF       ; Multiple definitions are supported
"(9)M9"     : CoolantOFF       ; Dynapath style
"MCode~9"   : CoolantOFF       ; Anilam style
```

Function code: 5
Group code: 100300
RVP codes

CtantSurfSpeed

Description: Maps constant surface speed
Tech Tips: Selects or enables constant surface speed. On most CNC controls this is a G96 code. Examples using this function keyword include the following:

```
"G96"       : CtantSurfSpeed       ; Fanuc style
```

Function code: 5
Group code: 100300
RVP codes

CtantSurfSpeedCancel

Description: Maps canceling constant surface speed
Tech Tips: Cancels or disables constant surface speed. On most CNC controls this is a G97 code. Examples using this function keyword include the following:

```
"G97"       : CtantSurfSpeedCancel     ; Fanuc style
```

Function code: 5
Group code: 100300
RVP codes

DisableRotaryComp

Description: Maps canceling 5 axis rotary compensation

Tech Tips: Cancels or disables 5 axis rotary compensation. This is only available in newer 5 axis CNC machines. On most CNC controls this is a G code. Examples using this function keyword include the following:

"G97" : DisableRotaryComp ; Fidia style
"TRAOFF" : DisableRotaryComp ; Seimens style

Function code: 5
Group code: 100300
RVP codes

EnableRotaryComp

Description: Maps starting 5 axis rotary compensation
Tech Tips: Starts 5 axis rotary compensation. This is only available in newer 5 axis CNC machines. On most CNC controls this is a G code. Examples using this function keyword include the following:

"G96" : EnableRotaryComp ; Fidia style
"TRAORI" : EnableRotaryComp ; Seimens style

Function code: 5
Group code: 100300
RVP codes

HSpindleStop

Description: Maps horizontal spindle stop
Tech Tips: Selects horizontal spindle stop. On most CNC controls this is a M code. Examples using this function keyword include the following:

"M118" : HSpindleStop ; Varies per machine tool builder

Function code: 1
Group code: 100300
RVP codes

LineNumber

Description: Maps line or sequence numbers
Tech Tips: Selects and defines line or sequence numbers. On most CNC controls this is a N number. Examples using this function keyword include the following:

"N",int : LineNumber ; Fanuc style
"N",real : LineNumber ; Optional method

Function code: 1
Group code: 100300
RVP codes

LoadMachine

Description: Maps loading a second machine definition
Tech Tips: Selects and defines when to load a second machine definition with enough physicall differences to justify it. For example, attaching a right angle head on a 4 axis VMC to perform HMC work. On most CNC controls this is an M or G code. Examples using this function keyword include the following:

"G154" : LoadMachine ; SNK style

NOTE: This keyword complements RestoreMachine.

Function code: 1
Group code: 100300
RVP codes

MaxSpindleSpeed

Description: Maps maximum spindle speed
Tech Tips: Selects maximum spindle speed. On most CNC controls this is the S register. An example using this function keyword includes the following:

```
"S10000"      : MaxSpindleSpeed      ; Standard style
```

Function code: 10
Group code: 100300
RVP codes

OptStop

Description: Maps optional stop
Tech Tips: Selects optional stop. On most CNC controls this is a M1 code. Examples using this function keyword include the following:

```
"M1"          : OptStop              ; Fanuc style
"M01"         : OptStop              ; Multiple definitions are supported
"(9)M1"       : OptStop              ; Dynapath style
```

Function code: 8
Group code: 100300
RVP codes

Pause0

Description: Maps program stop
Tech Tips: Selects program stop. On most CNC controls this is a M0 code. Examples using this function keyword include the following:

```
"M0"          : Pause0               ; Fanuc style
"M00"         : Pause0               ; Multiple definitions are supported
```

NOTE: Pause0 has been replaced with Stop. Pause0 can still be used but should be replaced with the newer function keyword Stop.

Function code: 7
Group code: 100300
RVP codes

Pause1

Description: Maps optional stop
Tech Tips: Selects optional stop. On most CNC controls this is a M1 code. Examples using this function keyword include the following:

```
"M1"          : Pause1               ; Fanuc style
"M01"         : Pause1               ; Multiple definitions are supported
```

NOTE: Pause1 has been replaced with OptStop. Pause1 can still be used but should be replaced with the newer function keyword OptStop.

Function code: 7
Group code: 100300
RVP codes

ProgramEnd

Description: Maps end of program
Tech Tips: Selects the end of the CNC program or file. On most CNC controls this is a M2 or M30. Examples using this function keyword include the following:

```
“M2”          : ProgramEnd        ; Fanuc style
“M02”         : ProgramEnd        ; Multiple definitions are supported
“M30”         : ProgramEnd
"END",null    : ProgramEnd        ; Heidenhain style
"(9)M2"       : ProgramEnd        ; Dynapath style
"END"         : ProgramEnd
"EndMain"     : ProgramEnd        ; Anilam style
```

Function code: 7
Group code: 100300
RVP codes

RestoreMachine

Description: Maps unloading a second machine definition
Tech Tips: Selects and defines when to unload the second machine definition and return to its original machine. For example, attaching a right angle head on a 4 axis VMC to perform HMC sooner or later needs to be unloaded returning the machine to a VMC. On most CNC controls this is an M or G code. Examples using this function keyword include the following:

```
“G155”        : RestoreMachine          ; SNK style
```

NOTE: This keyword complements LoadMachine.

Function code: 1
Group code: 100300
RVP codes

SetMachineDatum

Description: Maps selecting a machine datum
Tech Tips: Selects a machine datum or point within comments. Examples using this function keyword include the following:

```
“MCH_DATUM=”,multi@106    : SetMachineDatum     ; Predator style
```

Function code: 2
Group code: 100300
RVP codes

SetMCHFileName

Description: Maps selecting a machine

Tech Tips: Selects a machine from the library of machines within comments. Machines or .MCH files store in the ..\common files\machines directory. Examples using this function keyword include the following:

“MCH_FILE=”,multi@103 : SetMCHFileName ; Predator style

Function code: 2
Group code: 100300
RVP codes

SetOrigRelocation

Description: Maps selecting an origin relocation
Tech Tips: Selects a programming origin or point within comments. Examples using this function keyword include the following:

“ORIG_REL=”,multi@106 : SetOrigRelocation ; Predator style

Function code: 2
Group code: 100300
RVP codes

SpindleCW

Description: Maps spindle clockwise
Tech Tips: Selects spindle clockwise. On most CNC controls this is a M3 code. Examples using this function keyword include the following:

“M3”	: SpindleCW	; Fanuc style
“M03”	: SpindleCW	; Multiple definitions are supported
"(9)M3"	: SpindleCW	; Dynapath style
"M88"	: SpindleCW (FloodCoolant_On)	; Haas advanced M code
"M13"	: SpindleCW (FloodCoolant_On)	; Maho advanced M code

Function code: 2
Group code: 100300
RVP codes

SpindleCCW

Description: Maps spindle counter clockwise
Tech Tips: Selects spindle counter clockwise. On most CNC controls this is a M4 code. Examples using this function keyword include the following:

“M4”	: SpindleCCW	; Fanuc style
“M04”	: SpindleCCW	; Multiple definitions are supported
"(9)M4"	: SpindleCCW	; Dynapath style
"M14"	: SpindleCCW (FloodCoolant_On)	; Maho advanced M code

Function code: 3
Group code: 100300
RVP codes

SpindleSpeed

Description: Maps spindle speed
Tech Tips: Selects spindle speed. On most CNC controls this is the S register. An example using this function keyword includes the following:

“S”,int : SpindleSpeed ; Fanuc style

Function code: 9
Group code: 100300
RVP codes

SpindleStop

Description: Maps spindle stop
Tech Tips: Selects spindle stop for the first spindle. On most CNC controls this is a M5 code. Examples using this function keyword include the following:

“M5” : SpindleStop ; Fanuc style
“M05” : SpindleStop ; Multiple definitions are supported
"M89" : Spindlestop (Coolant_Off) ; Haas advanced M code

Function code: 1
Group code: 100300
RVP codes

SpindleStop2

Description: Maps a second spindle stop
Tech Tips: Selects spindle stop for the second spindle. On most CNC controls this is a M code. Examples using this function keyword include the following:

“M25” : SpindleStop2 ; Multiple definitions are supported
"M89" : Spindlestop2 (Coolant_Off) ; Advanced M code

Function code: 1
Group code: 100300
RVP codes

SpindleStop3

Description: Maps a third spindle stop
Tech Tips: Selects spindle stop for the third spindle. On most CNC controls this is a M code. Examples using this function keyword include the following:

“M35” : SpindleStop3 ; Multiple definitions are supported
"M89" : Spindlestop3 (Coolant_Off) ; Advanced M code

Function code: 1
Group code: 100300
RVP codes

SpindleStop4

Description: Maps a fourth spindle stop
Tech Tips: Selects spindle stop for the second spindle. On most CNC controls this is a M code. Examples using this function keyword include the following:

“M25” : SpindleStop4 ; Multiple definitions are supported
"M89" : Spindlestop4 (Coolant_Off) ; Advanced M code

Function code: 1
Group code: 100300
RVP codes

Stop

Description: Maps program stop
Tech Tips: Selects program stop. On most CNC controls this is a M0 code. Examples using this function keyword include the following:

```
"M0"        : Stop          ; Fanuc style
"M00"       : Stop          ; Multiple definitions are supported
"STOP"      : Stop          ; Heidenhain style
"(9)*M0"    : Stop          ; Dynapath style
```

Function code: 6
Group code: 100300

ToolClamp

Description: Maps enabling/disabling tool clamping
Tech Tips: Selects and defines enabling and disabling tool clamping. On most CNC controls this is a M code. Although not simulated by Predator Virtual CNC this is used for accurate cycle time calculation. Examples using this function keyword include the following:

```
"M60"       : ToolClamp     ; Varies per machine builder
"M82"       : ToolClamp     ; Haas style or
"M86"       : ToolClamp
"M25"       : ToolClamp     ; Siemens style
"M26"       : ToolClamp
```

NOTE: The time for a tool clamp can be configured within the machine definition.

Function code: 4
Group code: 100300
RVP codes

VSpindleStop

Description: Maps vertical spindle stop
Tech Tips: Selects vertical spindle stop. On most CNC controls this is a M code. Examples using this function keyword include the following:

```
"M19"       : VSpindleStop  ; Varies per machine tool builder
```

Function code: 1
Group code: 100300
RVP codes

ZHome

Description: Maps rapiding the tool home
Tech Tips: Selects rapiding the tool to the machine's tool change position's Z value. In other words the tool retracts in the Z axis only. The current X and Y axis values do not change. On most CNC controls this is a M code. Examples using this function keyword include the following:

```
"M25"       : ZHome         ; Bridgeport Style
```

Function code: 1
Group code: 100300
RVP codes

3.2.5.6 GROUP: FEED RATE

The feedrate function group includes all function keywords that map the different feed rate codes including feed per minute and feed per revolution. All feedrate function keywords are listed below with a short description:

FeedPerMinute ; Maps feed rate per minute
FeedPerRevolution ; Maps feed rate per revolution
InvTimeFeed ; Maps inverse time feed
OutFeed ; Maps feed rate
SetFeed ; Maps feed rate
SetMaxFeed ; Maps maximum feed rate

FeedPerMinute

Description: Maps feed rate per minute
Tech Tips: Selects feed rate per minute. On most CNC controls this is the G94 code. An example using this function keyword includes the following:

"G94" : FeedPerMinute ; Fanuc style
"G98" : FeedPerMinute

Function code: 1
Group code: 100400
RVP codes

FeedPerRevolution

Description: Maps feed rate per revolution
Tech Tips: Selects feed rate per minute. On most CNC controls this is the G94 code. An example using this function keyword includes the following:

"G95" : FeedPerRevolution ; Fanuc style
"G99" : FeedPerRevolution

Function code: 1
Group code: 100400

InvTimeFeed

Description: Maps inverse time feed rates
Tech Tips: Selects inverse time feed rates. On most CNC controls this is the G93 code. An example using this function keyword includes the following:

"G93" : InvTimeFeed ; Fanuc style

Function code: 1
Group code: 100400
RVP codes

OutFeed

Description: Maps feed Rate
Tech Tips: Selects feed rate. On most CNC controls this is the F register. An example using this function keyword includes the following:

"F",real : OutFeed ; Fanuc style

NOTE: OutFeed has been replaced with SetFeed. OutFeed can still be used but should be replaced with the newer function keyword SetFeed.

Function code: 1
Group code: 100400
RVP codes

SetFeed

Description: Maps feed rates
Tech Tips: Selects feed rates. On most CNC controls this is the F register. An example using this function keyword includes the following:

"F",real : SetFeed ; Fanuc style
"Feed~",real : SetFeed ; Anilam style

Function code: 1
Group code: 100400
RVP codes

SetMaxFeed

Description: Maps a maximum feed rate
Tech Tips: Selects feed rate. An example using this function keyword includes the following:

"FMAX" : SetMaxFeed(non_modal) ; Heidenhain style
"F*MAX" : SetMaxFeed(non_modal)

Function code: 1
Group code: 100400
RVP codes

3.2.5.7 GROUP: UNSUPPORTED

The unsupported function group includes only one function keyword it maps the unsupported CNC codes that should produce an error within Predator Virtual CNC. The unsupported error function keyword is listed below with a short description:

Error ; Maps CNC codes to an error condition

Error

Description: Maps CNC codes to an error condition
Tech Tips: Select CNC codes that Predator Virtual CNC should report as an error. This varies with each CNC control. Often this function keyword can be used to enforce particular programming practices within a programming department. Examples using this function keyword include the following:

"G9" : Error
"G09" : Error
"G10" : Error
"G11" : Error

NOTE: User definable error messages can be added to specific G and M codes

"M27" : Error (ErrorMsg:"Chip conveyer not supported")
"G60" : Error (ErrorMsg:"Single direction positioning not supported")

Function code: 0
Group code: 100800

RVP codes

3.2.5.8 GROUP: REFERENCE POINT RETURN

The refpointreturn function group includes the function keywords that map returning to or from a reference point. These refpointreturn function keywords are listed below with a short description:

ReturnToRefP ; Maps return to reference point
ReturnFromRefP ; Maps return from reference point

ReturnToRefP

Description: Maps returning to a reference point
Tech Tips: Returns the tool to the reference point's specified axes through an intermediate point. On most CNC controls this is a G28 code. An example using this function keyword includes the following:

"G28",multi@5 : ReturnToRefP ; Fanuc style

NOTE: The most popular usage of reference moves is G28G91Z0 it is important that either the TOOL_CHANGE or MACHINE_HOME parameter, position and pattern be defined for proper simulation.

Function code: 1
Group code: 100600
RVP codes

ReturnFromRefP

Description: Maps returning from a reference point
Tech Tips: Returns the tool from the reference point's specified axes through an intermediate point. On most CNC controls this is a G29 code. An example using this function keyword includes the following:

"G29",multi@5 : ReturnFromRefP ; Fanuc style

Function code: 2
Group code: 100600
RVP codes

3.2.5.9 GROUP: CUTTER COMPENSATION

The cuttercomp function group includes all function keywords that map the different cutter compensation codes including diameter comp right, left and cancel. In addition, length comp plus, minus and cancel. All cuttercomp function keywords are listed below with a short description:

NOTE: Cutter compensation for turning was added in v5.1 of Predator Virtual CNC.

CutterCompCancel ; Maps canceling diameter compensation
CutterCompLeft ; Maps diameter compensation left
CutterCompRight ; Maps diameter compensation right
LengthCompCancel ; Maps canceling length compensation
LengthCompMinus ; Maps negative length compensation
LengthCompPlus ; Maps positive length compensation
LengthOffNegative ; Maps decreased or negative length compensation
LengthOffPositive ; Maps increased or positive length compensation
LengthOffDoubleNegative ; Maps double negative length compensation
LengthOffDoublePositive ; Maps double positive length compensation
SetCutterCompD ; Maps the diameter compensation offset register

SetDiamOffset ; Maps setting the diameter compensation values
SetLengthCompH ; Maps the length compensation offset register
SetLengthOffset **;** Maps setting the length compensation values

CutterCompCancel

Description: Maps canceling diameter compensation
Tech Tips: Selects cancel tool diameter compensation. On most CNC controls this is a G40 code. An example using this function keyword includes the following:

```
"G40"            : CutterCompCancel    ; Fanuc style or
"C0"             : CutterCompCancel    ; Dynapath style or
"ToolComp~Off"   : CutterCompCancel    ; Anilam style or
"R0"             : CutterCompCancel    ; Heidenhain style
```

Function code: 22
Group code: 100700
RVP codes

CutterCompLeft

Description: Maps diameter compensation left
Tech Tips: Selects tool diameter compensation left. On most CNC controls this is a G41 code. An example using this function keyword includes the following:

```
"G41"            : CutterCompLeft      ; Fanuc style or
"C1"             : CutterCompLeft      ; Dynapath style or
"ToolComp~Left"  : CutterCompLeft      ; Anilam style or
"RL"             : CutterCompLeft      ; Heidenhain style
```

Sets left Tool compensation
Function code: 20
Group code: 100700
RVP codes

CutCompRight

Description: Maps diameter compensation right
Tech Tips: Selects tool diameter compensation right. On most CNC controls this is a G42 code. An example using this function keyword includes the following:

```
"G42"            : CutterCompRight     ; Fanuc style or
"C2"             : CutterCompRight     ; Dynapath style or
"ToolComp~Right" : CutterCompRight     ; Anilam style or
"RR"             : CutterCompRight     ; Heidenhain style
```

Function code: 21
Group code: 100700
RVP codes

LengthCompCancel

Description: Maps canceling length compensation
Tech Tips: Selects cancel cutter length compensation. On most CNC controls this is a G40 code. An example using this function keyword includes the following:

```
"G40"            : LengthCompCancel    ; Fanuc style
```

Function code: 3
Group code: 100700
RVP codes

LengthCompMinus

Description Maps negative length compensation
Tech Tips: Selects negative cutter length compensation. On most CNC controls this is a G44 code. An example using this function keyword includes the following:

“G44” : LengthCompMinus ; Fanuc style

Function code: 2
Group code: 100700
RVP codes

LengthCompPlus

Description: Maps positive length compensation
Tech Tips: Selects positive cutter length compensation. On most CNC controls this is a G43 code. An example using this function keyword includes the following:

“G43” : LengthCompPlus ; Fanuc style

Function code: 1
Group code: 100700
RVP codes

LengthOffDoubleNegative

Description: Maps double negative length compensation
Tech Tips: Selects double negative cutter length compensation. On most CNC controls this is a G48 code. An example using this function keyword includes the following:

“G48” : LengthOffDoubleNegative ; Fanuc style

Function code: 6
Group code: 100700
RVP codes

LengthOffDoublePositive

Description: Maps double positive length compensation
Tech Tips: Selects double positive cutter length compensation. On most CNC controls this is a G47 code. An example using this function keyword includes the following:

“G47” : LengthOffDoublePositive ; Fanuc style

Function code: 6
Group code: 100700
RVP codes

LengthOffNegative

Description: Maps decreased or negative length compensation

Tech Tips: Selects negative cutter length compensation. On most CNC controls this is a G46 code. An example using this function keyword includes the following:

“G46” : LengthOffNegative ; Fanuc style

Function code: 4
Group code: 100700
RVP codes

LengthOffPositive

Description: Maps increased or positive length compensation
Tech Tips: Selects positive cutter length compensation. On most CNC controls this is a G45 code. An example using this function keyword includes the following:

“G45” : LengthOffPositive ; Fanuc style

Function code: 4
Group code: 100700
RVP codes

SetCutterCompD

Description: Maps the diameter compensation offset register
Tech Tips: Selects the tool diameter cutter compensation offset. On most CNC controls this is a D register. An example using this function keyword includes the following:

“D”,int : SetCutterCompD ; Fanuc style or
“R”,expression : SetCutterCompD ; Heidenhain style

Function code: 23
Group code: 100700
RVP codes

SetDiamOffset

Description: Maps setting the diameter compensation values
Tech Tips: Selects the tool diameter cutter compensation values that should be applied within the CNC program. On most CNC controls these are entered into an offsets table directly on the CNC control. An example using this function keyword includes the following:

"DIAM_OFFSET",multi@104 : SetDiamOffset ; Predator style or
"OFFN=",expression : SetDiamOffset ; Siemens style

Function code: 23
Group code: 100700
RVP codes

SetLengthOffset

Description: Maps setting the length compensation values
Tech Tips: Selects the tool length cutter compensation values that should be applied within CNC comments. On most CNC controls these are entered into an offsets table directly on the CNC control. An example using this function keyword includes the following:

"LENGTH_OFFSET",multi@105: SetLengthOffset ; Predator style

Function code: 23
Group code: 100700
RVP codes

SetLengthCompH

Description: Maps the length compensation offset register
Tech Tips: Selects the tool length cutter compensation offset. On most CNC controls this is an H register. An example using this function keyword includes the following:

“H”,int : SetLengthCompH ; Fanuc style or
“L”,expression : SetLengthCompH ; Heidenhain style

Function code: 8
Group code: 100700
RVP codes

3.2.5.10 GROUP: IGNORE

The ignore function group includes only one function keyword it maps the CNC codes that should be ignored by Predator Virtual CNC. The ignore function keyword is listed below with a short description:

Ignore ; Maps CNC codes that should be ignored

Ignore

Description: Maps CNC codes that should be ignored
Tech Tips: Select CNC codes that Predator Virtual CNC should ignore. This varies with each CNC control. Often this function keyword can be used to enforce particular programming practices within a programming department. Examples using this function keyword include the following:

"G4" : Ignore
"G04" : Ignore
"G7" : Ignore
"G07" : Ignore

NOTE: User definable warning messages can be added to specific G and M codes

"M27" : Ignore (WarningMsg:”Chip conveyer not supported”)
"G60" : Ignore (WarningMsg:”Single direction positioning not supported”)

Function code: 0
Group code: 100800
RVP codes

3.2.5.11 GROUP: WORK COORDINATES

The workcoord function group includes all function keywords that map the different work or fixture offset codes. All workcoord function keywords are listed below with a short description:

AxisRotation ; Maps setting axis rotation
CancelAxisRotation ; Maps canceling axis rotation
ChangeOffset ; Maps work or fixture offset’s X, Y and Z values
ChangeAllWorkCoord ; Maps changing all work coordinates

ChangeProgramOrigin ; Maps changing the programming point
DisableWorkCoord ; Maps disabling work coordinates
LocalCoordSys ; Maps a local work offset
SetCoordSystem ; Maps setting system coordinates
SetWorkCoord1 ; Maps first work or fixture offset
SetWorkCoord2 ; Maps second work or fixture offset
SetWorkCoord3 ; Maps third work or fixture offset
SetWorkCoord4 ; Maps fourth work or fixture offset
SetWorkCoord5 ; Maps fifth work or fixture offset
SetWorkCoord6 ; Maps sixth work or fixture offset
SetWorkCoord7 - 100 ; Maps the 7th through 100th work or fixture offset
SetWorkCoord ; Maps up to 100 work or fixture offsets
SetWorkCoordNext ; Maps setting next work coordinates
Translate ; Maps a work offset translation
TranslateInc ; Maps an incremental work offset translation

AxisRotation

Description: Maps setting axis rotation
Tech Tips: Selects setting axis rotation. An example using this function keyword includes the following:

“M19.1” : AxisRotation

Function code: 6
Group code: 101400
RVP codes

CancelAxisRotation

Description: Maps canceling axis rotation
Tech Tips: Selects canceling axis rotation. An example using this function keyword includes the following:

“M19.2” : AxisRotation

Function code: 6
Group code: 101400
RVP codes

ChangeOffset

Description: Maps work or fixture offset’s X, Y, and Z values
Tech Tips: Loads a work or fixture offset’s X, Y, or Z coordinates. On most CNC controls this is a G10 followed by an L, P, X, Y and Z values. An example using this function keyword with an appropriate pattern definition follows:

```
“G10”,multi@13              : ChangeOffset      ; Fanuc style or
"CYCL*DEF*7.0",multi@19     : ChangeOffset      ; Heidenhain style or
"CYCL*DEF*7.1",multi@19     : ChangeOffset
"CYCL*DEF*7.2",multi@19     : ChangeOffset
"L5",multi@28               : ChangeOffset      ; Siemens style
```

Function code:
Group code: 101400
RVP codes

ChangeAllWorkCoord

Description: Maps changing all work coordinates
Tech Tips: Selects changing all work coordinates. An example using this function keyword follows:

“G66” : ChangeAllWorkCoord

Function code:
Group code: 101400
RVP codes

ChangeProgramOrigin

Description: Maps changing the programming point
Tech Tips: Selects a new programming point. For example, on older lathes the programming point has to be defined prior to each tool change. On a Fanuc this is often a G50. An example using this function keyword follows:

“G50”,multi@17 : ChangeProgramOrigin ; Fanuc style

Function code:
Group code: 101400
RVP codes

DisableWorkCoord

Description: Maps disabling work coordinates
Tech Tips: Selects disabling work coordinates. An example using this function keyword includes the following:

“G53” : DisableWorkCoord ; Siemens style

Function code: 6
Group code: 101400
RVP codes

LocalCoordSys

Description: Maps a local work offset
Tech Tips: Selects a local work offset. An example using this function keyword includes the following:

“G52”,multi@15 : LocalCoordSys ; Fanuc style

Function code: 6
Group code: 101400
RVP codes

SetCoordSystem

Description: Maps setting work coordinates
Tech Tips: Selects setting work coordinate values within CNC comments. On most CNC controls these values are entered in a table on the CNC. An example using this function keyword includes the following:

"COORD_SYS",multi@107 : SetCoordSystem ; Predator style

Function code: 6
Group code: 101400

RVP codes

SetWorkCoord1

Description: Maps the first work or fixture offset
Tech Tips: Selects the first work offset. On most CNC controls this is a G54 code. An example using this function keyword includes the following:

```
"G54"       : SetWorkCoord1     ; Fanuc style or
"E1"        : SetWorkCoord1     ; Fadal style or
"E01"       : SetWorkCoord1     ; Multiple definitions are supported
"G15*H1"    : SetWorkCoord1     ; Okuma style
```

Function code: 6
Group code: 101400
RVP codes

SetWorkCoord2

Description: Maps the second work or fixture offset
Tech Tips: Selects the second work offset. On most CNC controls this is a G55 code. An example using this function keyword includes the following:

```
"G55"       : SetWorkCoord2     ; Fanuc style or
"E2"        : SetWorkCoord2     ; Fadal style or
"E02"       : SetWorkCoord2     ; Multiple definitions are supported
"G15*H2"    : SetWorkCoord1     ; Okuma style
```

Function code: 7
Group code: 101400
RVP codes

SetWorkCoord3

Description: Maps the third work or fixture offset
Tech Tips: Selects the third work offset. On most CNC controls this is a G56 code. An example using this function keyword includes the following:

```
"G56"       : SetWorkCoord3     ; Fanuc style or
"E3"        : SetWorkCoord3     ; Fadal style or
"E03"       : SetWorkCoord3     ; Multiple definitions are supported
"G15*H3"    : SetWorkCoord1     ; Okuma style
```

Function code: 8
Group code: 101400
RVP codes

SetWorkCoord4

Description: Maps the fourth work or fixture offset
Tech Tips: Selects the fourth work offset. On most CNC controls this is a G57 code. An example using this function keyword includes the following:

```
"G57"       : SetWorkCoord4     ; Fanuc style or
"E4"        : SetWorkCoord4     ; Fadal style or
"E04"       : SetWorkCoord4     ; Multiple definitions are supported
"G15*H4"    : SetWorkCoord1     ; Okuma style
```

Function code: 9

Group code: 101400
RVP codes

SetWorkCoord5

Description: Maps the fifth work or fixture offset
Tech Tips: Selects the fifth work offset. On most CNC controls this is a G58 code. An example using this function keyword includes the following:

```
"G58"       : SetWorkCoord5     ; Fanuc style or
"E5"        : SetWorkCoord5     ; Fadal style or
"E05"       : SetWorkCoord5     ; Multiple definitions are supported
"G15*H5"    : SetWorkCoord1     ; Okuma style
```

Function code: 10
Group code: 101400
RVP codes

SetWorkCoord6

Description: Maps the sixth work or fixture offset
Tech Tips: Selects the sixth work offset. On most CNC controls this is a G59 code. An example using this function keyword includes the following:

```
"G59"       : SetWorkCoord6     ; Fanuc style or
"E6"        : SetWorkCoord6     ; Fadal style or
"E06"       : SetWorkCoord6     ; Multiple definitions are supported
"G15*H6"    : SetWorkCoord1     ; Okuma style
```

Function code: 11
Group code: 101400
RVP codes

SetWorkCoord7 – SetWorkCoord100

Description: Maps the 7^{th} – 100^{th} work or fixture offsets
Tech Tips: Selects the 7^{th} – 100^{th} work or fixture offsets. An example using this function keywords include the following:

```
"G110"      : SetWorkCoord7     ; Haas style
"G111"      : SetWorkCoord8
"G112"      : SetWorkCoord9
"G113"      : SetWorkCoord10
"G114"      : SetWorkCoord11
"G115"      : SetWorkCoord12
"G116"      : SetWorkCoord13
"G117"      : SetWorkCoord14
"G118"      : SetWorkCoord15
"G119"      : SetWorkCoord16
"G120"      : SetWorkCoord17
"G121"      : SetWorkCoord18
"G122"      : SetWorkCoord19
"G123"      : SetWorkCoord20
"G124"      : SetWorkCoord21
"G125"      : SetWorkCoord22
"G126"      : SetWorkCoord23
"G127"      : SetWorkCoord24
"G128"      : SetWorkCoord25
"G129"      : SetWorkCoord26
```

Function code: 11
Group code: 101400
RVP codes

SetWorkCoord

Description: Maps up to 100 work or fixture offsets
Tech Tips: Selects up to 100 work offsets. On most CNC controls this is an E code. An example using this function keyword includes the following:

```
“H”,int              : SetWorkCoord      ; Vickers 2100 style or
“E”,int              : SetWorkCoord      ; Fadal style or
“G54.1”,multi@14     : SetWorkCoord      ; Fanuc G10 style or
"G15*H",int          : SetWorkCoord1     ; Okuma style
```

Function code:
Group code: 101400
RVP codes

SetWorkCoordNext

Description: Maps setting next work coordinates
Tech Tips: Selects setting the next work coordinates. An example using this function keyword includes the following:

```
"G110"               : SetWorkCoordNext  ; Haas style
"G111"               : SetWorkCoordNext
"G112"               : SetWorkCoordNext
"G59.1"              : SetWorkCoordNext  ; Allen Bradley style
"G59.2"              : SetWorkCoordNext
"G59.3"              : SetWorkCoordNext
```

Function code: 6
Group code: 101400
RVP codes

Translate

Description: Maps setting translating work coordinates
Tech Tips: Selects setting a work coordinate translation function. An example using this function keyword includes the following:

```
"TRANS",multi@36     : Translate         ; Siemens 840 style
```

Function code: 6
Group code: 101400
RVP codes

TranslateInc

Description: Maps setting translating work coordinates
Tech Tips: Selects setting a work coordinate translation function. An example using this function keyword includes the following:

```
"ATRANS",multi@36    : TranslateInc      ; Siemens 840 style
```

Function code: 6
Group code: 101400

RVP codes

3.2.5.12 GROUP: ABSOLUTE ZERO POSITION

The absolutezero function group includes only one function keyword it maps the code to set the absolute zero position. The setabsolutezero function keyword is listed below with a short description:

ChangeProgramOrigin ; Maps setting changing the programming origin position
SetAbsoluteZero ; Maps setting the absolute zero position

ChangeProgramOrigin

Description: Maps changing the programming position
Tech Tips: Selects changing the programming origin. On most CNC controls this is a G50 or G92 code. An example using this function keyword includes the following:

NOTE: This is often used on older CNC controls this style of CNC programming requires the programming origin to be changed and reset per tool.

"G50" : ChangeProgramOrigin ; Fanuc style or
"G92" : ChangeProgramOrigin

Function code: 1
Group code: 101500
RVP codes

SetAbsoluteZero

Description: Maps the absolute zero position
Tech Tips: Selects the absolute Zero position. On most CNC controls this is a G52 or G92 code. An example using this function keyword includes the following:

"G50" : SetAbsoluteZero ; Fanuc style or
"G92" : SetAbsoluteZero
"G26" : SetAbsoluteZero ; Bostomatic style or
"G98" : SetAbsoluteZero ; Vickers 2100 style

Function code: 1
Group code: 101500
RVP codes

3.2.5.13 GROUP: THREADING

The threading function group includes only one function keyword it maps the code to set the thread cutting motion. The threadmove function keyword is listed below with a short description:

ThreadCycle ; Maps threading canned cycle
ThreadMove ; Maps threading

ThreadCycle

Description: Maps threading canned cycle
Tech Tips: Selects a threading canned cycle. On most CNC controls this is a G76 code. An example using this function keyword includes the following:

"G76",multi@14 : ThreadCycle ; Fanuc style or
"G85",multi@14 : ThreadCycle ; Emco style or
"G71",multi@17 : ThreadCycle ; Okuma style

Function code: 1
Group code: 101600
RVP codes

ThreadMove

Description: Maps threading
Tech Tips: Selects threading motion. On most CNC controls this is a G32 or G33 code. An example using this function keyword includes the following:

```
"G32",multi@2        :ThreadMove        ; Fanuc style or
"G33",multi@2        :ThreadMove
"G77",multi@2        :ThreadMove        ; Okuma style
```

Function code: 1
Group code: 101600
RVP codes

3.2.5.14 GROUP: SUBPROGRAM

The subprograms function group includes all function keywords that map the different subprogram codes including starting and ending a subprogram. In addition, calling a subprogram and returning from a subprogram. All subprogram function keywords are listed below with a short description:

```
EndMacroModalCall          ; Maps calling the end of a modal macro
MacroCall                  ; Maps calling a macro
MacroModalCall             ; Maps calling a modal macro
ProgramStart               ; Maps the start of the main program
SubProgramCall             ; Maps calling a subprogram
SubProgramCallandRotation  ; Maps calling a subprogram while optionally rotating the sub
SubProgramEnd              ; Maps the end of each subprogram
SubProgramEx               ; Maps calling an extended subprogram
SubProgramRet              ; Maps returning from a subprogram
SubProgramRetEnd           ; Maps returning end point of subroutines and sub programs
SubProgramRetEx            ; Maps returning from an extended subprogram
SubProgramStart            ; Maps the start of each subprograms
```

EndMacroModalCall

Description: Maps the end of a modal macro call
Tech Tips: Selects the end of a modal macro call. On most CNC controls this is a G67 code. An example of using this function keyword includes the following:

```
"G67"                : MacroCall        ; Fanuc style
```

Function code: 4
Group code: 101700
RVP codes

MacroCall

Description: Maps the calling of macros
Tech Tips: Selects the macro call. On most CNC controls this is a G65 code. An example of using this function keyword includes the following:

```
"G65",multi@8        : MacroCall        ; Fanuc style
```

Function code: 4
Group code: 101700
RVP codes

MacroModalCall

Description: Maps the calling of modal macros
Tech Tips: Selects a modal macro call. An example of using this function keyword includes the following:

```
"G66"          : MacroModalCall    ; Fanuc style or
"MCALL"        : MacroModalCall    ; Siemens 840 style
```

Function code: 4
Group code: 101700
RVP codes

ProgramStart

Description: Maps the starting point of the main program
Tech Tips: Selects the main program start point. On most CNC controls this is a O or a %. An example of using this function keyword includes the following:

```
"O"                          : ProgramStart    ; Fanuc style without subs or
"%\13\10O"                   : ProgramStart    ; Fanuc style with subs or
"PGM",multi@16,delimiters    : ProgramStart    ; G&L Style
```

NOTE: Often the main program start is not defined since it tends to conflict with sub program start definitions. The above example tries to define a unique string that includes a % followed by a CR LF characters and then the letter O.

Function code: 1
Group code: 101700
RVP codes

NOTE: ProgramInit is also supported for backwards compatibility.

SubProgramCall

Description: Maps the calling of subroutines and sub programs
Tech Tips: Selects the subroutine and sub program call. On most CNC controls this is a M98 code. An example of using this function keyword includes the following:

```
"M98",multi@6                              : SubProgramCall    ; Fanuc style or
"G94",multi@6                              : SubProgramCall    ; Bostomatic style or
"G20";multi@6                              : SubProgramCall    ; Fagor style or
"SUB:",multi@6                             : SubProgramCall    ; Siemens 840 style or
"CLS",multi@6                              : SubProgramCall    ; Vickers 2100 style or
"CALL",multi@6                             : SubProgramCall    ; Okuma style or
"CLS",multi@6,multigstrictord,delimiters   : SubProgramCall    ; G&L style or
"CALL*LBL",int                             : SubProgramCall    ; Heidenhain style
"L",multi@6,multigstrictord                : SubProgramCall
"M97",multi@6                              : SubProgramCall    ; Haas M97 style
```

NOTE: There are two limitations with Haas M97 style sub program support. First, when both M98 and M97 style sub program calls will be used at the same time, all sub program numbers must be unique. For example, Predator Vritual CNC

does not support calling M97 P1000 and M98 P1000 within the same program. The second limitation is when M97 style subs will be used, the total number of sequence numbers must be less then 4096. This is because every sequence number is potentially the beginning of a sub program and Predator Virtual CNC has a limit of 4096 sub programs per program. If M97 style subs will be used, try to not have sequence numbers on every line. Just have sequence numbers where they are needed. If this cannot be done, then keep the total number of sequence numbers less then 4096. The actual sequence numbers themselves can be greater then 4096. So M97 P6000 is supported. Just keep the total number of sequence numbers less then 4096.

Function code: 4
Group code: 101700
RVP codes

SubProgramCallandRotation

Description: Maps the calling of subroutines and sub programs while optionally rotating the sub
Tech Tips: Very few CNCs support this method of calling subs. An example of using this function keyword includes the following:

“G22”,multi@21 : SubProgramCall ; Mitsubishi style

Function code: 4
Group code: 101700
RVP codes

SubProgramEnd

Description: Maps the ending point of subroutines and sub programs
Tech Tips: Selects the subroutine and sub program ending point. On most CNC controls this is not used. Instead a SubProgramRet is often used.

Function code: 3
Group code: 101700
RVP codes

SubProgramEx

Description: Maps the calling of extended subroutines and sub programs
Tech Tips: Selects an extended or secondary subroutine or sub program call. On most CNC controls only one method of sub program calls is supported and this keyword is not used. An example of using this function keyword includes the following:

“L”,int : SubProgramEx ; Fadal style

Function code: 4
Group code: 101700
RVP codes

SubProgramRet

Description: Maps the returning from subroutines and sub programs
Tech Tips: Selects the subroutine and sub program return. On most CNC controls this is a M99 code. An example of using this function keyword includes the following:

“M99” : SubProgramRet ; Fanuc style or

```
"G93"          : SubProgramRet          ; Bostomatic style or
"G24"          : SubProgramRet          ; Fagor style or
"M17"          : SubProgramRet          ; Siemens 840 style or
"ENS"          : SubProgramRet          ; Vickers 2100 style or
"RTS"          : SubProgramRet          ; Okuma style or
"RSB,"         : SubProgramRet          ; G&L style or
"LBL*0"        : SubProgramRet          ; Heidenhain style
"LBL0"         : SubProgramRet
"G98L0"        : SubProgramRet
"G98*L0"       : SubProgramRet
```

Function code: 5
Group code: 101700
RVP codes

SubProgramRetEnd

Description: Maps the returning end point of subroutines and sub programs
Tech Tips: Selects the returning subroutine and sub program end point. On most CNC controls this is not used. Instead a SubProgramRet is often used.

Function code: 6
Group code: 101700
RVP codes

SubProgramRetEx

Description: Maps the returning from extended subroutines and sub programs
Tech Tips: Selects an extended or secondary subroutine and sub program return. On most CNC controls only one method of sub program calls is supported and this keyword is not used. An example of using this function keyword includes the following:

```
"M17"          : SubProgramRetEx        ; Fadal style
```

Function code: 5
Group code: 101700
RVP codes

SubProgramStart

Description: Maps the starting point of subroutines and sub programs
Tech Tips: Selects the subroutine and sub program starting point. On most CNC controls this is a : or an O. An example of using this function keyword includes the following:

```
"O",int                          : SubProgramStart      ; Fanuc style or
":",int                          : SubProgramStart
"G92"                            : SubProgramStart      ; Bostomatic style or
"G22";multi@5                    : SubProgramStart      ; Fagor style or
"DFS",string                     : SubProgramStart      ; Vickers 2100 style or
"DFS",multi@17,delimiters        : SubProgramStart      ; G&L style or
"LBL",multi@18,multigstrictord   : SubProgramStart      ; Heidenhain style
"G98",multi@13,multigstrictord   : SubProgramStart
"N",int                          : SubProgramStart      ; Haas M97 style or
```

Function code: 2
Group code: 101700
RVP codes

3.2.5.15 GROUP: VARIABLES

The variables function group includes all function keywords that map defining, setting, and getting variables, parameters, global and local variables. All variable function keywords are listed below with a short description:

DefineGlobalVar ; Maps defining a global variable
DefineLocalVar ; Maps defining a local variable
DefineParameter ; Maps defining a parameter
GetActiveTool ; Maps getting the active tool number
GetActiveDCode ; Maps getting the active diameter compensation value
GetActiveX ; Maps getting the active horizontal machine coordinate
GetActiveY ; Maps getting the active vertical machine coordinate
GetActiveZ ; Maps getting the active depth machine coordinate
GetActiveA ; Maps getting the active rotary horizontal machine coordinate
GetActiveB ; Maps getting the active rotary vertical machine coordinate
GetActiveC ; Maps getting the active rotary depth machine coordinate
GetCurrentReg ; Maps getting current register
GetCurrentVar ; Maps getting current variable
GetGlobalVar ; Maps getting a global variable
GetLocalVar ; Maps getting a local variable
GetParameter ; Maps getting a parameter
GetVariable ; Maps getting a variable
SetGlobalVar ; Maps setting a global variable
SetLocalVar ; Maps setting a local variable
SetParameter ; Maps setting a parameter
SetVariable ; Maps setting a variable

DefineGlobalVar

Description: Maps global variables
Tech Tips: Selects the global variables.

Function code: 2
Group code: 101800
RVP codes:

DefineLocalVar

Description: Maps local variables
Tech Tips: Selects the local variables.

Function code: 3
Group code: 101800
RVP codes:

DefineParameter

Description: Maps parameter definitions
Tech Tips: Selects the parameter definition.

Function code: 1
Group code: 101800
RVP codes:

GetActiveTool

Description: Maps getting the active tool number

Tech Tips: Selects how the active or current tool number is retrieved. On most CNC controls this is a system variable. An example of using this function keyword includes the following:

```
"#4120"     : GetActiveTool     ; Fanuc style or
"TN"        : GetActiveTool     ; Fadal style or
 "#3026"    : GetActiveTool     ; Haas style or
"ATC"       : GetActiveTool     ; G&L style
```

Function code: 8
Group code: 101800
RVP codes:

GetActiveDCode

Description: Maps getting the active diameter compensation value
Tech Tips: Selects how the active or current diameter compensation value is retrieved. On most CNC controls this is a system variable. An example of using this function keyword includes the following:

```
"#4107"     : GetActiveDCode    ; Fanuc style or
"ADC"       : GetActiveDCode    ; G&L style
```

Function code: 8
Group code: 101800
RVP codes:

GetActiveX

Description: Maps getting the active horizontal axis
Tech Tips: Selects how the current horizontal or X axis value is retrieved in machine coordinates. On most CNC controls this is a system variable. An example of using this function keyword includes the following:

```
"#5121"     : GetActiveX        ; Fanuc style
```

Function code: 8
Group code: 101800
RVP codes:

GetActiveY

Description: Maps getting the active vertical axis
Tech Tips: Selects how the current horizontal or Y axis value is retrieved in machine coordinates. On most CNC controls this is a system variable. An example of using this function keyword includes the following:

```
"#5122"     : GetActiveY        ; Fanuc style
```

Function code: 8
Group code: 101800
RVP codes:

GetActiveZ

Description: Maps getting the active depth axis
Tech Tips: Selects how the current depth or Z axis value is retrieved in machine coordinates. On most CNC controls this is a system variable. An example of using this function keyword includes the following:

"#5122" : GetActiveZ ; Fanuc style

Function code: 8
Group code: 101800
RVP codes:

GetActiveA

Description: Maps getting the active horizontal rotary axis
Tech Tips: Selects how the current horizontal rotary or A axis value is retrieved in machine coordinates. On most CNC controls this is a system variable. An example of using this function keyword includes the following:

"#5123" : GetActiveA ; Fanuc style

Function code: 8
Group code: 101800
RVP codes:

GetActiveB

Description: Maps getting the active vertical rotary axis
Tech Tips: Selects how the current vertical rotary or B axis value is retrieved in machine coordinates. On most CNC controls this is a system variable. An example of using this function keyword includes the following:

"#5124" : GetActiveB ; Fanuc style

Function code: 8
Group code: 101800
RVP codes:

GetActiveC

Description: Maps getting the active depth rotary axis
Tech Tips: Selects how the current horizontal rotary or C axis value is retrieved in machine coordinates. On most CNC controls this is a system variable. An example of using this function keyword includes the following:

"#5125" : GetActiveC ; Fanuc style

Function code: 8
Group code: 101800
RVP codes:

GetCurrentReg

Description: Maps getting the current register
Tech Tips: Selects how the active or current register is retrieved. On most CNC controls this is not used. An example of using this function keyword includes the following:

"PPX" : GetCurrentReg ; G&L style
"PPZ" : GetCurrentReg
"PPU" : GetCurrentReg
"PPW" : GetCurrentReg
"PPI" : GetCurrentReg
"PPK" : GetCurrentReg

"PPR" : GetCurrentReg

Function code: 8
Group code: 101800
RVP codes:

GetCurrentVar

Description: Maps getting the current variable
Tech Tips: Selects how the active or current variable is retrieved. On most CNC controls this is not used. An example of using this function keyword includes the following:

“PP”,int : GetCurrentVar ; G&L style

Function code: 8
Group code: 101800
RVP codes:

GetGlobalVar

Description: Maps global variable retrieval
Tech Tips: Selects how global variables are retrieved. On most CNC controls this is a number sign. An example of using this function keyword includes the following:

“#”,int : GetGlobalVar ; Fanuc style or
"#V",int : GetGlobalVar ; Fadal style or
"V",int : GetGlobalVar
"P",int : GetGlobalVar ; Fagor style or
“[“,multi@7 : GetGlobalVar ; Vickers 2100 style or
"R",int : GetGlobalVar ; Fadal style or
"Q",int : GetGlobalVar ; Heidenhain style or
"VC",int : GetGlobalVar ; Okuma style or
"P",int : GetGlobalVar ; G&L style

NOTE: Expressions can also be used to to evaluate variable names. For example, Fanuc Macro B supports the following X#[100+2] this would move the tool based on the value stored in variable #102. If support for this programming style is desired, use expressions. An example follows:

“#”,expression : GetGlobalVar ; Fanuc style

Function code: 8
Group code: 101800
RVP codes:

GetLocalVar

Description: Maps local variable retrieval
Tech Tips: Selects how local variables are retrieved. On most CNC controls this is a number sign. An example of using this function keyword includes the following:

“#”,int :GetLocalVar ; Fanuc style or
“V”,int :GetLocalVar ; Fadal style

Function code: 9
Group code: 101800

RVP codes:

GetParameter

Description: Maps parameter retrieval
Tech Tips: Selects how parameters are retrieved.

```
“R”          GetParameter        ; Fadal style
“R”,int      GetParameter
“#R”,int     GetParameter
```

Function code: 7
Group code: 101800
RVP codes:

GetVariable

Description: Maps variable retrieval
Tech Tips: Selects how variables are retrieved. On most CNC controls this is a number sign. An example of using this function keyword includes the following:

```
 “#”,int      :GetVariable        ; Fanuc style
```

Function code: 10
Group code: 101800
RVP codes:

SetGlobalVar

Description: Maps global variable assignment
Tech Tips: Selects how global variables are assigned. On most CNC controls this is an equal sign. An example of using this function keyword includes the following:

```
“=”,multi@7          : SetGlobalVar      ; Fanuc style or
"+",expression       : SetGlobalVar      ; Fadal style or
"=K",expression      : SetGlobalVar      ; Fagor style or
"P01",expression     : SetGlobalVar      ; Heidenhain style
```

Function code: 5
Group code: 101800
RVP codes

SetLocalVar

Description: Maps local variable assignment
Tech Tips: Selects how local variables are assigned. On most CNC controls this is an equal sign. An example of using this function keyword includes the following:

```
“=”,multi@7  : SetLocalVar       ; Fanuc style
```

Function code: 6
Group code: 101800
RVP codes:

SetParameter

Description: Maps parameter assignment

Tech Tips: Selects how parameters are assigned.

"R",multi@31 : SetParameter ; Siemens style
"+",expression : SetParameter ; Fadal style

Function code: 4
Group code: 101800
RVP codes:

SetVariable

Description: Maps variable assignment
Tech Tips: Selects how variables are assigned.

"=",expression : SetVariable ; Fadal style

Function code: 11
Group code: 101800
RVP codes

3.2.5.16 GROUP: DONOTHING

This section is nearly identical to the IGNORE section with one difference. Any G and M codes that are defined within the DONOTHING section will not display a warning message regardless of the ENABLE_WARNING parameter. So some G codes should display a warning and some G codes should be ignored without displaying a warning message. Set the ENABLE_WARNING parameter to ON and add the G codes that should display a warning message in the IGNORE section and add the G codes that should be ignored without displaying a warning message in the DONOTHING section.

DoNothing ; Ignore CNC codes without warning messages

DoNothing

Description: Maps CNC codes that should be ignored
Tech Tips: Select CNC codes that Predator Virtual CNC should ignore without displaying a warning message. This can vary with each CNC control. Examples using this function keyword include the following:

"G4" : DoNothing
"G04" : DoNothing
"G7" : DoNothing
"G07" : DoNothing

Function code: 0
Group code: 100800
RVP codes

NOTE: The ignore group internal code is 109990.

See section **3.2.5.16** for more information on the Ignore function keyword.

3.2.5.17 GROUP: COMMENTS

The comments function group includes all function keywords that map the different comment codes including start, end, and comments at the end of a NC block or line of code. All comments function keywords are listed below with a short description:

CommentEnds ; Maps the end of a comment
CommentFollows ; Maps the remainder of the line as a comment

CommentStarts ; Maps the start of a comment

CommentEnds

Description: Maps the end of a comment
Tech Tips: Selects the end of a comment. On most CNC controls this is an close parenthesis. The reverse post will ignore everything between a CommentStarts and a CommentEnds.

")",null CommentEnds ; Fanuc style

Function code: 2
Group code: 512
RVP codes:

CommentFollows

Description: Maps comments or block skips
Tech Tips: Selects the start of a comment or block skip. The reverse post will ignore the remainder of the block or line. An example of using this function keyword includes the following:

"/",null CommentFollows ; Block skip or
"*",null CommentFollows ; Fadal style or
"(T)",null CommentFollows ; Dynapath style

Function code: 1
Group code: 512
RVP codes:

CommentStarts

Description: Maps the start of a comment
Tech Tips: Selects the start of a comment. On most CNC controls this is an open parenthesis. The reverse post will ignore everything until it sees a CommentEnds.

"(",null CommentStarts ; Fanuc style

Function code: 1
Group code: 512
RVP codes:

3.2.5.20 GROUP: STOCK AND FIXTURE DEFINITION

The stockdef function group includes all function keywords that map the different stock and fixture creation definitions often contained within comments. Standard stock and fixture shapes include boxes, cylinders, holes, and custom shapes via STL files. All stockdef function keywords are listed below with a short description:

CreateStockBox ; Maps stock box creation
CreateStockCyl ; Maps stock cylinder creation
CreateStockCylHole ; Maps stock cylinder hole creation
CreateStockCylFixt ; Maps fixture cylinder creation
CreateStockFixt ; Maps fixture box creation
CreateStockHole ; Maps stock box hole creation
CreateStockSTL ; Maps STL stock creation
CreateStockSTLFixt ; Maps STL fixture creation

SetSTKFileName ; Maps loading a stock library file from within a comment

CreateStockBox

Description: Maps the box stock definition from within a comment
Tech Tips: Selects how one or more boxes of stock can be scanned from within a comment. This function keyword can be changed to match the output of a CAM system or programming convention. An example on using this function keyword includes the following:

"SBOX",multi@9 : CreateStockBox ; Predator style

NOTE: CreateStockBox, CreateStockFixt and CreateStockHole requires a multi@9 similar to the above. An example on using this feature to automatically create two pieces of stock when scanning a CNC program, should include the following:

```
O1234
(SBOX X0 Y0 Z0 L3.5 W5 H1)
(SBOX X5 Y0 Z0 L3.5 W5 H1)
…
M30
```

Function code: 1
Group code: 101900
RVP codes

CreateStockCyl

Description: Maps cylindrical stock definition from within a comment
Tech Tips: Selects how one or more cylinders of stock can be scanned from within a comment. This function keyword can be changed to match the output of a CAM system or programming convention. An example on using this function keyword includes the following:

"SCYL",multi@10 : CreateStockCyl ; Predator style

NOTE: The r_stock_cyl_axis keyword supports three values 1-3. Using the examples provided S1 is an X axis cylinder, S2 is a Y axis cylinder and S3 is a Z axis cylinder. In the above example CreateStockCyl, CreateStockCylFixt and CreateStockCylHole requires a multi@10 similar to the above. An example on using this feature to automatically create two pieces of cylindrical stock when scanning a CNC program, should include the following:

```
O1234
(SCYL S1 X0 Y0 Z0 D3.5 L5)
(SCYL S2 X5 Y0 Z0 D3.5 L5)
…
M30
```

CreateStockCylFixt

Description: Maps cylindrical fixture definition from within a comment
Tech Tips: Selects how one or more fixture cylinders can be scanned from within a comment. This function keyword can be changed to match the output of a CAM system or programming convention. An example on using this function keyword includes the following:

"FCYL",multi@10 : CreateStockCylFixt ; Predator style

NOTE: The r_stock_cyl_axis keyword supports three values 1-3. Using the examples provided S1 is an X axis cylinder, S2 is a Y axis cylinder and S3 is a Z axis cylinder. In the above example CreateStockCyl, CreateStockCylFixt and CreateStockCylHole requires a multi@10 similar to the above. An example on using this feature to automatically create two pieces of cylindrical stock when scanning a CNC program, should include the following:

```
O1234
(FCYL S1 X0 Y0 Z0 D3.5 L5)
(FCYL S2 X5 Y0 Z0 D3.5 L5)
…
M30
```

CreateStockCylHole

Description: Maps cylindrical hole definition from within a comment
Tech Tips: Selects how one or more cylindrical holes can be subtracted from stock or fixtures when scanned from within a comment. This function keyword can be changed to match the output of a CAM system or programming convention. An example on using this function keyword includes the following:

"HCYL",multi@10 : CreateStockCylHole ; Predator style

NOTE: The r_stock_cyl_axis keyword supports three values 1-3. Using the examples provided S1 is an X axis hole, S2 is a Y axis hole and S3 is a Z axis hole. In the above example CreateStockCyl, CreateStockCylFixt and CreateStockCylHole requires a multi@10 similar to the above. An example on using this feature to automatically subtract two holes from a plate stock when scanning a CNC program, should include the following:

```
O1234
(SBOX X0 Y0 Z0 L10 W10 H1)
(FCYL S1 X0 Y0 Z0 D3.5 L5)
(FCYL S2 X5 Y0 Z0 D3.5 L5)
…
M30
```

CreateStockFixt

Description: Maps the fixture definition from within a comment
Tech Tips: Selects how one or more boxes of fixtures can be scanned from within a comment. This function keyword can be changed to match the output of a CAM system or programming convention. An example on using this function keyword includes the following:

"FBOX",multi@9 : CreateStockFixt ; Predator style

NOTE: CreateStockBox, CreateStockFixt and CreateStockHole requires a multi@9 similar to the above. An example on using this feature to automatically create two fixtures when scanning a CNC program, should include the following:

```
O1234
```

(FBOX X0 Y0 Z0 L.5 W5 H1)
(FBOX X5 Y0 Z0 L.5 W5 H1)
…
M30

CreateStockHole

Description: Maps the box hole definition from within a comment
Tech Tips: Selects how one or more boxes are subtracted from a stock or fixture while scanned from within a comment. This function keyword can be changed to match the output of a CAM system or programming convention. An example on using this function keyword includes the following:

“HBOX”,multi@9 : CreateStockHole ; Predator style

NOTE: CreateStockBox, CreateStockFixt and CreateStockHole requires a multi@9 similar to the above. An example on using this feature to automatically create two fixtures when scanning a CNC program, should include the following:

O1234
(HBOX X0 Y0 Z0 L.5 W5 H1)
(HBOX X5 Y0 Z0 L.5 W5 H1)
…
M30

CreateStockSTL

Description: Maps the STL file stock definition from within a comment
Tech Tips: Selects how one or more STL stock files can be scanned from within a comment. This function keyword can be changed to match the output of a CAM system or programming convention. An example on using this function keyword includes the following:

“SSTL”,multi@11 : CreateStockSTL ; Predator style

NOTE: CreateStockSTL and CreateStockSTLFixt requires a multi@11 similar to the above. An example on using this feature to automatically create two pieces of stock when scanning a CNC program, should include the following:

O1234
(SSTL STOCK1.STL)
(SSTL STOCK2.STL)
…
M30

CreateStockSTLFixt

Description: Maps the STL file fixture definition from within a comment
Tech Tips: Selects how one or more STL fixture files can be scanned from within a comment. This function keyword can be changed to match the output of a CAM system or programming convention. An example on using this function keyword includes the following:

“FSTL”,multi@11 : CreateStockSTLFixt ; Predator style

NOTE: CreateStockSTL and CreateStockSTLFixt requires a multi@11 similar to the above. An example on using this feature to automatically create two

pieces of stock when scanning a CNC program, should include the following:

```
O1234
(FSTL FIXTURE1.STL)
(FSTL FIXTURE2.STL)
…
M30
```

SetSTKFileName

Description: Maps loading a stock library file from within a comment
Tech Tips: Selects loading one or more stock and fixture library file or .STK file from within a comment. This function keyword can be changed to match the output of a CAM system or programming convention. An example on using this function keyword includes the following:

```
"STK_FILE=",multi@102        : SetSTKFileName          ; Predator style
```

```
O1234
(STK_FILE=PALLET01.STK)
…
M30
```

3.2.5.21 GROUP: LATHECYCLES

The lathecycles function group includes all function keywords that map the different lathe canned cycle definitions. Standard OD and ID cycles for roughing, semi-finishing and finishing cycles are supported. In addition, tapping, grooving, face drilling are also supported. All lathecycles function keywords are listed below with a short description:

```
EndLabel                ; Maps the end of canned cycles
FaceDrillCycle          ; Maps face drilling cycles
FaceLongTurningCycle    ; Maps face long turning cycles
FinishCycle             ; Maps finish cycles
RoughXCycle             ; Maps X roughing cycles
RoughZCycle             ; Maps Z roughing cycles
StartLabel              ; Maps the start of canned cycles
TurnCycle               ; Maps rough and semi-finish cycles
TurnGrooveCycle         ; Maps grooving cycles
TurnTapCycle            ; Maps tapping cycles
```

EndLabel

Description: Maps the end of canned cycles
Tech Tips: Selects the end of a profile to be used within a canned cycle. An example on using this function keyword includes the following:

```
"[END",int          : EndLabel          ; Vickers 2100 style or
"G80"               : EndLabel          ; Okuma style or
"(ENC)"             : EndLabel          ; Cincinnati style
```

Function code: 12
Group code: 102000
RVP codes:

FaceDrillCycle

Description: Maps face drilling cycles

Tech Tips: Selects the face drilling cycles. An example on using this function keyword includes the following:

"G74",multi@19 : FaceDrillCycle( )

NOTE: The FaceDrillCycle supports drilling at any position while DrillCycle supports drilling down the centerline only.

Function code: 10770
Group code: 102000
Parameters: peck dwell retractfeed manualfeed bottomCW bottomstop oriented_bottomstop
RVP codes:

FaceLongTurningCycle

Description: Maps face long turning cycles
Tech Tips: Selects the face long turning cycles. An example on using this function keyword includes the following:

"G84",multi@19 : FaceLongTurningCycle( ) ; Emco style

Function code: 10770
Group code: 102000
RVP codes:

FinishCycle

Description: Maps lathe finish cycles
Tech Tips: Selects the lathe finish cycles. An example on using this function keyword includes the following:

"M129",multi@10 : FinishCycle()

Function code: 1
Group code: 102000
RVP codes:

RoughXCycle

Description: Maps X axis lathe roughing cycles
Tech Tips: Selects X axis lathe roughing cycles. An example on using this function keyword includes the following:

"M201",multi@10 : RoughXCycle()

Function code: 3
Group code: 102000
RVP codes:

RoughZCycle

Description: Maps Z axis lathe roughing cycles
Tech Tips: Selects Z axis lathe roughing cycles. An example on using this function keyword includes the following:

"M202",multi@10 : RoughZCycle()

Function code: 2

Group code: 102000
RVP codes:

StartLabel

Description: Maps the start of canned cycles
Tech Tips: Selects the start of a profile to be used within a canned cycle. An example on using this function keyword includes the following:

```
"[BEGIN",int        : StartLabel        ; Vickers 2100 style or
"G81",string        : StartLabel        ; Okuma style
```

Function code: 11
Group code: 102000
RVP codes:

TurnCycle

Description: Maps lathe roughing and semi-finishing turn cycles
Tech Tips: Selects the lathe roughing and semi-finishing turn cycles. An example on using this function keyword includes the following:

```
"G70",multi@4       : TurnCycle(finish)                          ; Fanuc style or
"G71",multi@4       : TurnCycle(horizontal box_strategy pre_finish)
"G71",multi@4       : TurnCycle(horizontal box_strategy pocket)
"G72",multi@4       : TurnCycle(vertical box_strategy)
"G72",multi@15      : TurnCycle(finish)                          ; Vickers 2100 style or
"G73",multi@16      : TurnCycle(horizontal box_strategy)
"G85",multi@4       : TurnCycle(horizontal box_strategy)         ; Okuma style or
"G87",multi@4       : TurnCycle(finish)
"(BAR,",multi@4     : TurnCycle(horizontal box_strategy post_finish) ; Cincinnati style
```

NOTE: By combining the turn cycle parameters additional custom turn cycles can be created.

NOTE: The Pocket parameter is used to enable support for complex profiles that require pocketing of one or more areas. Just like a CNC control, Predator Virtual CNC does not compensate the tool motion for the back of the insert. However, any resulting collisions with the back of the insert are shown. Most CNC controls do not support complex profiles within their turning cycles. If this is the case, do not include the pocket parameter within the turncycle definition. With some CNC controls the ability to machine complex pockets is based on the version of the CNC firmware and not just the model number of the CNC control.

Function code: 10500
Group code: 102000
Parameters: finish horizontal vertical box_strategy pre_finish pocket
RVP codes:

TurnGrooveCycle

Description: Maps lathe grooving cycles
Tech Tips: Selects the lathe grooving cycles. An example on using this function keyword includes the following:

```
"G75",multi@20      : TurnGrooveCycle(horizontal)        ; Fanuc style or
"G74",multi@18      : TurnGrooveCycle(vertical)
"G73",multi@18      : TurnGrooveCycle(horizontal)        ; Okuma style
```

NOTE: The TurnGrooveCycle keyword's behavior is influenced by the cycle_param_style parameter. For example setting cycle_param_style=fanuc supports two line grooving cycles while setting cycle_param_style=new_fanuc support one line grooving cycles.

Function code: 10700
Group code: 102000
Parameters: horizontal vertical
RVP codes:

TurnTapCycle

Description: Maps lathe tapping cycles
Tech Tips: Selects the lathe tapping cycles. An example on using this function keyword includes the following:

```
"G77",multi@15      : TurnTapCycle(horizontal)      ; Fanuc style or
"G92",multi@18      : TurnTapCycle(horizontal)
"G94",multi@15      : TurnTapCycle(vertical)
"G90",multi@15      : TurnTapCycle(horizontal)      ; Okuma style
"G94",multi@15      : TurnTapCycle(vertical)
```

NOTE: By combining the tapping cycle parameters additional custom tapping cycles can be created.

Function code: 10600
Group code: 102000
Parameters: horizontal vertical
RVP codes:

3.2.5.22 GROUP: PROGRAMMING

The programming function group includes all function keywords that map the different looping and conditional definitions. IF THENs, DO WHILEs, and GOTOs are supported. All programming function keywords are listed below with a short description:

```
Condition           ; Maps conditional statements
EndLoop             ; Maps the end of loops
Goto                ; Maps goto style jump statements
SetLabel            ; Maps the beginning of labels
StartLoop           ; Maps the beginning of loops
Then                ; Maps conditional statements
```

Condition

Description: Maps conditional statements
Tech Tips: Selects conditional statements. An example on using this function keyword includes the following:

```
"IF",expression     : Condition     ; Fanuc style or
"WHILE",expression  : Condition
"WH",expression     : Condition     ; Haas style or
"IFS",expression    : Condition     ; G&L style
```

Function code: 1
Group code: 102100

RVP codes:

EndLoop

Description: Maps the end of loops
Tech Tips: Selects the end of loops. An example on using this function keyword includes the following:

```
"END",int              : EndLoop          ; Fanuc style
```

Function code: 3
Group code: 102100
RVP codes:

Goto

Description: Maps goto style jumps
Tech Tips: Selects goto style jumps. An example on using this function keyword includes the following:

```
"GOTO",int             : Goto      ; Fanuc style or
"GTO",string,delimiters : Goto     ; G&L style or
"GTO",int              : Goto      ; Vickers 2100 style or
"GOTO*N",int           : Goto      ; Okuma style
```

Function code: 4
Group code: 102100
RVP codes:

SetLabel

Description: Maps the beginning of labels
Tech Tips: Selects the beginning of labels. An example on using this function keyword includes the following:

```
"LBL",string,delimiters        : SetLabel          ; G&L style
```

Function code: 5
Group code: 102100
RVP codes:

StartLoop

Description: Maps the beginning of loops
Tech Tips: Selects the beginning of loops. An example on using this function keyword includes the following:

```
"DO",int                       : StartLoop         ; Fanuc style
```

Function code: 2
Group code: 102100
RVP codes:

Then

Description: Maps conditional statements
Tech Tips: Selects conditional statements. This keyword works with and requires the CONDITION keyword to also be defined. An example on using this function keyword includes the following:

"THEN" : Then ; Fadal style

Function code: 1
Group code: 102100
RVP codes:

3.2.5.23 GROUP: ARITHMETIC FUNCTIONS

The arithmetic function group includes all function keywords that map the basic math functions. Addition, subtraction, multiplication, division, modulus division and exponentiation are supported. All arithmetic function keywords are listed below with a short description:

Add ; Maps additions
Divide ; Maps divisions
Exponent ; Maps exponentiations
Increment ; Maps increments
Integer ; Maps integers
Module ; Maps modulus
Multiply ; Maps multiplications
Subtract ; Maps subtractions

NOTE: By combining the arithmetic parameters after the above functions additional custom arithmetic functions can be supported.

Parameters: single double in pre

Add

Description: Maps additions
Tech Tips: Selects additions of variables and constants. An example on using this function keyword includes the following:

"+",#5 : Add() ; Fanuc style

Function code: 1
Group code: 102200
RVP codes:

Divide

Description: Maps divisions
Tech Tips: Selects divisions of variables and constants. An example on using this function keyword includes the following:

"/",#10 : Divide() ; Fanuc style

Function code: 4
Group code: 102200
RVP codes:

Exponent

Description: Maps exponentiations
Tech Tips: Selects exponentiations of variables and constants. An example on using this function keyword includes the following:

"^",#20 : Exponent() ; Fanuc style
"**",#20 : Exponent() ; Cincinnati style

Function code: 6
Group code: 102200
RVP codes:

Increment

Description: Maps increments
Tech Tips: Selects increments of variables and constants. An example on using this function keyword includes the following:

```
"IC",multi@21          : Increment (single pre)          ; Siemens 840 style
```

Function code: 6
Group code: 102200
RVP codes:

Integer

Description: Maps integers
Tech Tips: Selects the integer value of variables and constants. For example, the integer function returns a 2 when 2.75 is provided as input. An example on using this function keyword includes the following:

```
"INT"                  : Integer (single pre)            ; Fadal style
"FIX"                  : Integer (single pre)            ; Fanuc style
```

NOTE: This function is similar to the Round function.

Function code: 6
Group code: 102200
RVP codes:

Module

Description: Maps modulus
Tech Tips: Selects modulus of variables and constants. An example on using this function keyword includes the following:

```
"MOD",#20              : Module()                        ; Fanuc style
"\",#20                : Module()                        ; Cincinnati 2100 style
```

Function code: 5
Group code: 102200
RVP codes:

Multiply

Description: Maps multiplications
Tech Tips: Selects multiplications of variables and constants. An example on using this function keyword includes the following:

```
"*",#10                : Multiply()                      ; Fanuc style
```

Function code: 3
Group code: 102200
RVP codes:

Subtract

Description: Maps subtractions
Tech Tips: Selects subtractions of variables and constants. An example on using this function keyword includes the following:

```
"-",#5          : Subtract()          ; Fanuc style
```

Function code: 2
Group code: 102200
RVP codes:

3.2.5.24 GROUP: BOOLEAN FUNCTIONS

The boolean function group includes all function keywords that map the boolean and conditional math functions. And, or, equal and not equal along with many others are supported. All boolean function keywords are listed below with a short description:

And ; Maps ands
Equal ; Maps equals
Great ; Maps greater thans
GreatOrEqual ; Maps greater than or equals
Less ; Maps less thans
LessOrEqual ; Maps less thans or equals
NotEqual ; Maps not equals
Or ; Maps ors
Xor ; Maps xors

NOTE: By combining the boolean parameters after the above functions additional custom boolean functions can be supported.

Parameters: single double in pre

And

Description: Maps ands
Tech Tips: Selects a boolean and condition between variables and constants. An example on using this function keyword includes the following:

```
"AND",#30       : And(double in)      ; Fanuc style
```

Function code: 8
Group code: 102220
RVP codes:

Equal

Description: Maps equals
Tech Tips: Selects a boolean equal condition between variables and constants. An example on using this function keyword includes the following:

```
"EQ",#30        : Equal(double in)    ; Fanuc style or
"=",#30         : Equal(double in)    ; Fadal style
```

Function code: 10
Group code: 102220
RVP codes:

Great

Description: Maps greater thans
Tech Tips: Selects a boolean greater than condition between variables and constants. An example on using this function keyword includes the following:

```
"GT",#30          : Great(double in)          ; Fanuc style or
">",#30           : Great(double in)          ; Fadal style
```

Function code: 12
Group code: 102220
RVP codes:

GreatOrEqual

Description: Maps greater thans or equals
Tech Tips: Selects a boolean greater than or equal condition between variables and constants. An example on using this function keyword includes the following:

```
"GE",#30          : GreatOrEqual(double in)   ; Fanuc style or
">=",#30          : GreatOrEqual(double in)   ; Fadal style
```

Function code: 14
Group code: 102220
RVP codes:

Less

Description: Maps less thans
Tech Tips: Selects a boolean less than condition between variables and constants. An example on using this function keyword includes the following:

```
"LT",#30          : Less(double in)           ; Fanuc style or
"<",#30           : Less(double in)           ; Fadal style
```

Function code: 13
Group code: 102220
RVP codes:

LessOrEqual

Description: Maps less thans or equals
Tech Tips: Selects a boolean less than or equal condition between variables and constants. An example on using this function keyword includes the following:

```
"LE",#30          : LessOrEqual(double in)    ; Fanuc style or
"<=",#30          : LessOrEqual(double in)    ; Fadal style
```

Function code: 15
Group code: 102220
RVP codes:

NotEqual

Description: Maps not equals
Tech Tips: Selects a boolean not equal condition between variables and constants. An example on using this function keyword includes the following:

"NE",#30 : NotEqual(double in) ; Fanuc style or
"<>",#30 : NotEqual(double in) ; Fadal style

Function code: 11
Group code: 102220
RVP codes:

Or

Description: Maps ors
Tech Tips: Selects a boolean or condition between variables and constants. An example on using this function keyword includes the following:

"OR",#30 : Or(double in) ; Fanuc style

Function code: 7
Group code: 102220
RVP codes:

Xor

Description: Maps xors
Tech Tips: Selects a boolean xor condition between variables and constants. An example on using this function keyword includes the following:

"XOR",#30 : Xor(double in) ; Fanuc style

Function code: 9
Group code: 102220
RVP codes:

3.2.5.24 GROUP: TRIGONOMETRIC FUNCTIONS

The trigonometric function group includes all function keywords that map the trigonometric and advanced math functions. Sin, cosine, tangent, round, and square root along with many others are supported. All trigonometric function keywords are listed below with a short description:

ABSOLUTE ; Maps absolutes
ARCCOSINE ; Maps arc cosines
ARCSINE ; Maps arc sins
ARCTANGENT ; Maps arc tangents
COSINE ; Maps cosines
EXPONENT ; Maps exponents
LOGARITHM ; Maps logarithms
ROUND ; Maps rounds
SINE ; Maps sines
SQUAREROOT ; Maps square roots
TANGENT ; Maps tangents

NOTE: By combining the trigonometric parameters after the above functions additional custom trigonometric functions can be supported.

Parameters: single double in pre

Absolute

Description: Maps absolutes
Tech Tips: Selects absolute functions between variables and constants. An example on using this function keyword includes the following:

"ABS",#10 : Absolute(single pre) ; Fanuc style

Function code: 21
Group code: 102300
RVP codes:

ArcCosine

Description: Maps arc cosines
Tech Tips: Selects arc cosine functions between variables and constants. An example on using this function keyword includes the following:

"ACOS",#10 : Arccosine(single pre) ; Fanuc style
"ARCCOS",#10 : Arccosine(single pre) ; Cincinnati 2100 style
"ASN",#10 : Arccosine(single pre)

Function code: 23
Group code: 102300
RVP codes:

ArcSine

Description: Maps arc sines
Tech Tips: Selects arc sine functions between variables and constants. An example on using this function keyword includes the following:

"ASIN",#10 : Arcsine(single pre) ; Fanuc style or
"ARCSIN",#10 : Arcsine(single pre) ; Cincinnati 2100 style

Function code: 19
Group code: 102300
RVP codes:

ArcTangent

Description: Maps arc tangents
Tech Tips: Selects arc tangent functions between variables and constants. An example on using this function keyword includes the following:

"ATAN",#10 : Arctangent(single pre) ; Fanuc style or
"ATN",#10 : Arctangent(single pre) ; Fadal style
"ARCTAN",#10 : Arctangent(single pre) ; Cincinnati 2100 style

Function code: 19
Group code: 102300
RVP codes:

Cosine

Description: Maps cosines
Tech Tips: Selects cosine functions between variables and constants. An example on using this function keyword includes the following:

"COS",#10 : Cosine(single pre) ; Fanuc style

Function code: 17
Group code: 102300
RVP codes:

Exponent

Description: Maps exponents
Tech Tips: Selects exponent functions between variables and constants. An example on using this function keyword includes the following:

"EXP",#10 : Exponent(single pre) ; Fanuc style

Function code: 25
Group code: 102300
RVP codes:

Logarithm

Description: Maps logarithms
Tech Tips: Selects logarithm functions between variables and constants. An example on using this function keyword includes the following:

"LN",#10 : Logarithm(single pre) ; Fanuc style

Function code: 24
Group code: 102300
RVP codes:

Round

Description: Maps rounding
Tech Tips: Selects the rounded up value of variables and constants. For example, the round function returns a 3 when 2.75 is provided as input. An example on using this function keyword includes the following:

"ROUND",#10 : Round(single pre) ; Fanuc style or
"RND",#10 : Round(single pre) ; Fadal style

Function code: 22
Group code: 102300
RVP codes:

Sine

Description: Maps sines
Tech Tips: Selects sine functions between variables and constants. An example on using this function keyword includes the following:

"SIN",#10 : Sine(single pre) ; Fanuc style

Function code: 16
Group code: 102300
RVP codes:

SquareRoot

Description: Maps square roots
Tech Tips: Selects square root functions between variables and constants. An example on using this function keyword includes the following:

"SQRT",#10 : Squareroot(single pre) ; Fanuc style or
"SQR",#10 : Squareroot(single pre) ; Fadal style

Function code: 20
Group code: 102300
RVP codes:

Tangent

Description: Maps tangents
Tech Tips: Selects tangent functions between variables and constants. An example on using this function keyword includes the following:

```
"TAN",#10          : Tangent(single pre)          ; Fanuc style
```

Function code: 18
Group code: 102300
RVP codes:

3.2.5.25 GROUP: SYSTEM VARIABLES

The system variables function keywords maps CNC system variables to Predator Virtual CNC variables. Tool numbers, diameter offsets, length offsets, custom variables, X, Y, and Z corrdinates can be accessed with system variables during processing of the CNC program.

CUSTOM ; Maps custom system variables
GETACTIVETOOL ; Maps the current tool number
GETACTIVEX ; Maps the current X axis in machine coordinates
GETACTIVEY ; Maps the current Y axis in machine coordinates
GETACTIVEZ ; Maps the current Z axis in machine coordinates
GETCUTTERCOMPD1 – D50 ; Maps diameter compensation registers 1 - 50
GETCUTTERCOMPH1 – H50 ; Maps length compensation registers 1-50

Custom

Description: Maps custom system variables
Tech Tips: Selects an unlimited number of pre-defined system variables. An example on using this function keyword includes the following:

```
#100=3.1416                ; Fanuc style
#101=2.5
```

NOTE: The above system variables are pre-defined and can be referenced using the variable numbers #100 and #101 within any CNC program. These system variables should match any system variables within the CNC machine. In the above example, PI can be referenced with the #100 system variable.

Function code: 21
Group code: 102300
RVP codes:

GetActiveTool

Description: Maps the current tool number
Tech Tips: Assigns the current tool number to a system variable. An example on using this function keyword includes the following:

```
#3026=GetActiveTool()      ; Fanuc style
TN=GetActiveTool()         ; Fadal style
```

Function code: 21

Group code: 102300
RVP codes:

GetActiveX

Description: Maps the current X axis
Tech Tips: Assigns the current X axis in machine coordinates to a system variable. An example on using this function keyword includes the following:

```
#5021=GetActiveX()          ; Fanuc style
AX=GetActiveX()             ; Fadal style
```

Function code: 21
Group code: 102300
RVP codes:

GetActiveY

Description: Maps the current Y axis
Tech Tips: Assigns the current Y axis in machine coordinates to a system variable. An example on using this function keyword includes the following:

```
#5022=GetActiveY()          ; Fanuc style
AY=GetActiveY()             ; Fadal style
```

Function code: 21
Group code: 102300
RVP codes:

GetActiveZ

Description: Maps the current Z axis
Tech Tips: Assigns the current Z axis in machine coordinates to a system variable. An example on using this function keyword includes the following:

```
#5023=GetActiveZ()          ; Fanuc style
AZ=GetActiveZ()             ; Fadal style
```

Function code: 21
Group code: 102300
RVP codes:

GetCutterCompD1 – D50

Description: Maps diameter compensation registers 1 - 50
Tech Tips: Assigns the diameter cutter compensation values to system variables. An example on using this function keyword includes the following:

```
#2401=GetCutterCompD1       ; Fanuc style
#2402=GetCutterCompD2
#2403=GetCutterCompD3
```

Function code: 21
Group code: 102300
RVP codes:

GetCutterCompH1 – H50

Description: Maps length compensation registers 1 - 50

Tech Tips: Assigns the cutter length compensation values to system variables. An example on using this function keyword includes the following:

```
#2001=GetCutterCompH1      ; Fanuc style
#2002=GetCutterCompH2
#2003=GetCutterCompH3
```

Function code: 21
Group code: 102300
RVP codes:

3.2.6 REGISTER Keywords:

Register keywords begin with an r_register_name and they allow specific G & M codes to be assigned to internal registers within Predator Virtual CNC. Although CNC controls don't use register names, these registers are identical to those on real CNC controls. Registers are maintained for modal values and updated as the NC or CNC program is processed just like real CNC controls. Over 200 registers are maintained by Predator Virtual CNC, refer to the following for specifics on each register.

NOTE: Register keywords can only be used within the pattern function groups.

r_Arc_CCW

Description: Maps the arc ccw direction
Tech Tips: Selects or assigns the arc counter clockwise direction register. An example of using this register keyword includes the following:

```
"+"          : r_arc_ccw          ; Heidenhain style or
"1"          : r_arc_ccw          ; Dynapath style
```

Function code: 0
Group code: 0
RVP codes

r_Arc_CW

Description: Maps the arc cw direction
Tech Tips: Selects or assigns the arc clockwise direction register. An example of using this register keyword includes the following:

```
"-"          : r_arc_cw           ; Heidenhain style or
"0"          : r_arc_cw           ; Dynapath style
```

Function code: 0
Group code: 0
RVP codes

r_Arc_Radius

Description: Maps the arc radius
Tech Tips: Selects or assigns the arc radius register. An example of using this register keyword includes the following:

```
%f,real      : r_arc_radius       ; APTCL style
```

Function code: 0

Group code: 0
RVP codes

r_Arc_Angle

Description: Maps the arc angle
Tech Tips: Selects or assigns the arc angle register value. An example of using this register keyword includes the following:

```
%f,int          : r_arc_angle          ; APTCL style
```

Function code: 0
Group code: 0
RVP codes

r_Arc_Normal_U

Description: Maps the horizontal arc normal
Tech Tips: Selects the current horizontal arc normal or U register position. Often this register keyword is used for processing APTCL instead of a CNC program. An example of using this register keyword includes the following:

```
%f,real         : r_arc_normal_u       ; APTCL style
```

Function code: 63
Group code: 0
RVP codes

r_Arc_Normal_V

Description: Maps the vertical arc normal
Tech Tips: Selects the current vertical arc normal or V register position. Often this register keyword is used for processing APTCL instead of a CNC program. An example of using this register keyword includes the following:

```
%f,real         : r_arc_normal_v       ; APTCL style
```

Function code: 64
Group code: 0
RVP codes

r_Arc_Normal_W

Description: Maps the depth arc normal
Tech Tips: Selects the current depth arc normal or W register position. Often this register keyword is used for processing APTCL instead of a CNC program. An example of using this register keyword includes the following:

```
%f,real         : r_arc_normal_w       ; APTCL style
```

Function code: 65
Group code: 0
RVP codes

r_Axis_U

Description: Maps the U or I axis register position
Tech Tips: Selects the current U or I axis normal vector. In most APTCL formats this is one of the parameters in a GOTO statement. An example of using this register keyword includes the following:

%f,real : r_axis_u ; APTCL style

Function code: 60
Group code: 0
RVP codes

r_ Axis_V

Description: Maps the V or J axis register position
Tech Tips: Selects the current V or J axis normal vector. In most APTCL formats this is one of the parameters in a GOTO statement. An example of using this register keyword includes the following:

%f,real : r_axis_v ; ATPCL style

Function code: 61
Group code: 0
RVP codes

r_ Axis_W

Description: Maps the W or K axis register position
Tech Tips: Selects the current W or K axis normal vector. In most APTCL formats this is one of the parameters in a GOTO statement. An example of using this register keyword includes the following:

%f,real : r_axis_w ; APTCL style

Function code: 62
Group code: 0
RVP codes

r_ChangeOff_4thAxisValue

Description: Maps the 4th axis change offset register
Tech Tips: Selects the current 4th axis or A axis offset register. In most CNC formats this is one of the parameters following a G10 code. An example of using this register keyword includes the following:

“A”,real : r_changeoff_4thAxisValue ; Fanuc style

Function code: 0
Group code: 0
RVP codes

r_ChangeOff_5thAxisValue

Description: Maps the 5th axis change offset register
Tech Tips: Selects the current 5th axis or B axis offset register. In most CNC formats this is one of the parameters following a G10 code. An example of using this register keyword includes the following:

“B”,real : r_changeoff_5thAxisValue ; Fanuc style

Function code: 0
Group code: 0
RVP codes

r_ChangeOff_Func

Description: Maps the change offset register
Tech Tips: Selects the current change offset register or L register. In most CNC formats this is one of the parameters following a G10 code. An example of using this register keyword includes the following:

“L”,int : r_changeoff_func ; Fanuc style

Function code: 0
Group code: 0
RVP codes

r_ChangeOff_Num

Description: Maps the change offset number
Tech Tips: Selects the current change offset number or P register. In most CNC formats this is one of the parameters following a G10 code. An example of using this register keyword includes the following:

“P”,int : r_changeoff_num ; Fanuc style

Function code: 0
Group code: 0
RVP codes

r_ChangeOff_ToolLength

Description: Maps the tool length change offset register
Tech Tips: Selects the current tool length or H axis offset register. In most CNC formats this is one of the parameters following a G10 code. An example of using this register keyword includes the following:

“H”,real : r_changeoff_toollength ; Fanuc style

Function code: 0
Group code: 0
RVP codes

r_ChangeOff_XValue

Description: Maps the X axis change offset register
Tech Tips: Selects the current horizontal or X axis offset register. In most CNC formats this is one of the parameters following a G10 code. An example of using this register keyword includes the following:

“X”,real : r_changeoff_xvalue ; Fanuc style

Function code: 0
Group code: 0
RVP codes

r_ChangeOff_YValue

Description: Maps the Y axis change offset register
Tech Tips: Selects the current vertical or Y axis offset register. In most CNC formats this is one of the parameters following a G10 code. An example of using this register keyword includes the following:

“Y”,real : r_changeoff_yvalue ; Fanuc style

Function code: 0
Group code: 0
RVP codes

r_ChangeOff_ZValue

Description: Maps the Z axis change offset register
Tech Tips: Selects the current depth or Z axis offset register. In most CNC formats this is one of the parameters following a G10 code. An example of using this register keyword includes the following:

```
“Z”,real                : r_changeoff_zvalue          ; Fanuc style
```

Function code: 0
Group code: 0
RVP codes

r_Coord_Mode

Description: Maps the coordinate mode
Tech Tips: Selects the coordinate mode between absolute and incremental. On most CNC controls this is a G98 or G99 code. An example of using this register keyword includes the following:

```
@3:#0
"G90"                   : use_absolute
"G91"                   : use_incremental
@2:#0
"",switch@3,casesens    : r_coord_mode
```

Function code: 220
Group code: 0
RVP codes

r_Cycle_Num

Description: Maps the cycle number
Tech Tips: Selects the cycle number. An example of using this register keyword includes the following:

```
"C",int         : r_cycle_num
```

Function code: 0
Group code: 0
RVP codes

r_Cycle_Param

Description: Maps the cycle parameter
Tech Tips: Selects the cycle parameter. An example of using this register keyword includes the following:

```
"P",int         : r_cycle_param
```

Function code: 0
Group code: 0
RVP codes

r_Diam_Offset

Description: Maps the diameter offset register
Tech Tips: Selects the current diameter offset register. In most CNC formats this is one of the parameters following a G10 code. An example of using this register keyword includes the following:

“D”,real : r_diam_offset ; Fanuc style

Function code: 0
Group code: 0
RVP codes

r_Drill_Pos_A

Description: Maps the A axis drill position
Tech Tips: Selects the current A or primary rotary axis drill position. On most CNC controls this is an A. An example of using this register keyword includes the following:

“A”,real : r_drill_pos_a ; Fanuc style

Function code: 0
Group code: 0
RVP codes

r_Drill_Pos_B

Description: Maps the B axis drill position
Tech Tips: Selects the current B or secondary rotary axis drill position. On most CNC controls this is an B. An example of using this register keyword includes the following:

“B”,real : r_drill_pos_b ; Fanuc style

Function code: 0
Group code: 0
RVP codes

r_Drill_Pos_X

Description: Maps the X axis drill position
Tech Tips: Selects the current X or horizontal axis drill position. On most CNC controls this is an X. An example of using this register keyword includes the following:

“X”,real : r_drill_pos_x ; Fanuc style

Function code: 70
Group code: 0
RVP codes

r_Drill_Pos_Y

Description: Maps the Y axis drill position
Tech Tips: Selects the current Y or vertical axis drill position. On most CNC controls this is an Y. An example of using this register keyword includes the following:

“Y”,real : r_drill_pos_y ; Fanuc style

Function code: 71
Group code: 0
RVP codes

r_Drill_Pos_Z

Description: Maps the Z axis drill position
Tech Tips: Selects the current Z or depth axis drill position. On most CNC controls this is an Z. An example of using this register keyword includes the following:

```
"Z",real        : r_drill_pos_z        ; Fanuc style
```

Function code: 72
Group code: 0
RVP codes

r_Dwell_Int_Time

Description: Maps the dwell time register position
Tech Tips: Selects the current P or dwell time register position as an integer. On most CNC controls this is an P. An example of using this register keyword includes the following:

```
"P",int         : r_dwell_int_time     ; Fanuc style
```

Function code: 0
Group code: 0
RVP codes

r_Dwell_Time

Description: Maps the dwell time register position
Tech Tips: Selects the current P or dwell time register position. On most CNC controls this is an P. An example of using this register keyword includes the following:

```
"P",real        : r_dwell_time         ; Fanuc style or
"R04*",real     : r_dwell_time         ; Siemens style
```

Function code: 245
Group code: 0
RVP codes

r_Expression

Description: Maps a mathematical expression
Tech Tips: Selects the mathematical expression as a string. On most CNC controls this is used with programming statements such as WHILE and IF commands. An example of using this register keyword includes the following:

```
%s,string_reg   : r_expression         ; Fanuc style
```

Function code: 420
Group code: 0
RVP codes

r_Feed_Rate

Description: Maps the feed rate
Tech Tips: Selects the current feed rate or F register position. On most CNC controls this is an F. An example of using this register keyword includes the following:

```
"F",real          : r_feed_rate          ; Fanuc style or
"Feed~",real      : r_feed_rate          ; Anilam style or
"R10*",real       : r_feed_rate          ; Siemens style
```

Function code: 40
Group code: 0
RVP codes

r_GenCycle_Angle

Description: Maps an angle for a generic cycle
Tech Tips: Selects the current angle register for use within a generic cycle. An example of using this register keyword includes the following:

```
"P133=K",real     : r_gencycle_angle     ; Fagor style
```

Function code: 0
Group code: 0
RVP codes

r_GenCycle_Angle_Y

Description: Maps a Y angle for a generic cycle
Tech Tips: Selects the current Y angle value for use within a generic cycle. An example of using this register keyword includes the following:

```
"P137=K",real     : r_gencycle_angle_y   ; Fagor style
```

Function code: 0
Group code: 0
RVP codes

r_GenCycle_Depth

Description: Maps a depth for a generic cycle
Tech Tips: Selects the current depth for use within a generic cycle. An example of using this register keyword includes the following:

```
"Z",real          : r_gencycle_depth     ; Haas style or
"DEPTH",real      : r_gencycle_depth     ; Heidenhain style or
"Q201=",real      : r_gencycle_depth
"P108=K",real     : r_gencycle_depth     ; Fagor style
```

Function code: 0
Group code: 0
RVP codes

r_GenCycle_Feed_Finish

Description: Maps a finishing feed rate for a generic cycle
Tech Tips: Selects the current finishing feed rate for use within a generic cycle. An example of using this register keyword includes the following:

```
"F",real            : r_gencycle_feed_finish      ; Haas style or
"Q206=",real        : r_gencycle_feed_finish      ; Heidenhain style
```

Function code: 0
Group code: 0
RVP codes

r_GenCycle_Feed_Rough

Description: Maps a roughing feed rate for a generic cycle
Tech Tips: Selects the current roughing feed rate for use within a generic cycle. An example of using this register keyword includes the following:

```
"F",real            : r_gencycle_feed_rough       ; Haas style or
"Q207=",real        : r_gencycle_feed_rough       ; Heidenhain style or
"R0",real           : r_gencycle_feed_rough       ; Fadal style or
"P198=K",real       : r_gencycle_feed_rough       ; Fagor style
"P205=K",real       : r_gencycle_feed_rough
```

Function code: 0
Group code: 0
RVP codes

r_GenCycle_Finishing

Description: Maps a finishing value for a generic cycle
Tech Tips: Selects the current finishing value for use within a generic cycle. An example of using this register keyword includes the following:

```
"F",real            : r_gencycle_finishing
```

Function code: 0
Group code: 0
RVP codes

r_GenCycle_Inc_Move

Description: Maps an incremental move for a generic cycle
Tech Tips: Selects the current incremental move value for use within a generic cycle. An example of using this register keyword includes the following:

```
"Q",real            : r_gencycle_inc_move         ; Haas style or
"P148=K",real       : r_gencycle_inc_move         ; Fagor style
```

Function code: 0
Group code: 0
RVP codes

r_GenCycle_N_Holes_X

Description: Maps the number of holes in the X axis for a generic cycle
Tech Tips: Selects the number of holes in the horizontal or X axis for use within a generic cycle. An example of using this register keyword includes the following:

```
"P132=K",real       : r_gencycle_n_holes_x        ; Fagor style
```

Function code: 0
Group code: 0

RVP codes

r_GenCycle_N_Holes_Y

Description: Maps the number of holes in the Y axis for a generic cycle
Tech Tips: Selects the number of holes in the horizontal or Y axis for use within a generic cycle. An example of using this register keyword includes the following:

```
"P136=K",real        : r_gencycle_n_holes_y        ; Fagor style
```

Function code: 0
Group code: 0
RVP codes

r_GenCycle_Plane

Description: Maps a plane for a generic cycle
Tech Tips: Selects an active plane for use within a generic cycle. An example of using this register keyword includes the following:

```
"G17.2"              : r_gencycle_plane
```

Function code: 0
Group code: 0
RVP codes

r_GenCycle_Pocket_R

Description: Maps the radius of the pocket for a generic cycle
Tech Tips: Selects the radius of the pocket for use within a generic cycle. An example of using this register keyword includes the following:

```
"I",real             : r_gencycle_pocket_r         ; Haas style or
"R2",real            : r_gencycle_pocket_r         ; Fadal style or
"P136=K",real        : r_gencycle_pocket_r         ; Fagor style or
"P141=K",real        : r_gencycle_pocket_r
"RADIUS",real        : r_gencycle_pocket_r         ; Heidenhain style
"Q220=",real         : r_gencycle_pocket_r
"Q223=",real         : r_gencycle_pocket_r
```

Function code: 0
Group code: 0
RVP codes

r_GenCycle_Pocket_Alt_R

Description: Maps an alternate radius of the pocket for a generic cycle
Tech Tips: Selects an alternate radius of the pocket for use within a generic cycle. An example of using this register keyword includes the following:

```
"K",real             : r_gencycle_pocket_alt_r     ; Haas style
```

Function code: 0
Group code: 0
RVP codes

r_GenCycle_R_Allow

Description: Maps a radius allowance for a generic cycle

Tech Tips: Selects a radius allowance for use within a generic cycle. An example of using this register keyword includes the following:

"A",real : r_gencycle_r_allow

Function code: 0
Group code: 0
RVP codes

r_GenCycle_Retract

Description: Maps a retract value for a generic cycle
Tech Tips: Selects a retract value for use within a generic cycle. An example of using this register keyword includes the following:

"R",real : r_gencycle_retract

Function code: 0
Group code: 0
RVP codes

r_GenCycle_RetRapid

Description: Maps a retract rapid value for a generic cycle
Tech Tips: Selects a retract rapid value for use within a generic cycle. An example of using this register keyword includes the following:

"R",real : r_gencycle_retrapid

Function code: 0
Group code: 0
RVP codes

r_GenCycle_Roughing

Description: Maps a roughing value for a generic cycle
Tech Tips: Selects the current roughing value for use within a generic cycle. An example of using this register keyword includes the following:

“R”,real : r_gencycle_roughing

Function code: 0
Group code: 0
RVP codes

r_GenCycle_Sense_CW

Description: Maps a cw direction for a generic cycle
Tech Tips: Selects a clock wise direction value for use within a generic cycle. An example of using this register keyword includes the following:

"DR",switch@4 : r_gencycle_sense_cw

Function code: 0
Group code: 0
RVP codes

r_GenCycle_Sense_CCW

Description: Maps a ccw direction for a generic cycle

Tech Tips: Selects a counter clock wise direction value for use within a generic cycle. An example of using this register keyword includes the following:

"DR",switch@4 : r_gencycle_sense_ccw

Function code: 0
Group code: 0
RVP codes

r_GenCycle_Slot_Dir

Description: Maps a slot direction for a generic cycle
Tech Tips: Selects a slot direction value for use within a generic cycle. An example of using this register keyword includes the following:

"D+" : r_gencycle_slot_dir

Function code: 0
Group code: 0
RVP codes

r_GenCycle_Start_Angle

Description: Maps a start angle for a generic cycle
Tech Tips: Selects the current start angle register for use within a generic cycle. An example of using this register keyword includes the following:

"P133=K",real : r_gencycle_start_angle ; Fagor style

Function code: 0
Group code: 0
RVP codes

r_GenCycle_Start_Pos

Description: Maps a start position for a generic cycle
Tech Tips: Selects the current start position register for use within a generic cycle. An example of using this register keyword includes the following:

"X",real : r_gencycle_start_pos

Function code: 0
Group code: 0
RVP codes

r_GenCycle_Tool_Radius

Description: Maps a tool radius for a generic cycle
Tech Tips: Selects the tool radius value for use within a generic cycle. An example of using this register keyword includes the following:

"R1",real : r_gencycle_tool_radius ; Fadal style

Function code: 0
Group code: 0
RVP codes

r_GenCycle_X_Allow

Description: Maps a X axis allowance for a generic cycle

Tech Tips: Selects a horizontal or X axis allowance value for use within a generic cycle. An example of using this register keyword includes the following:

"X",real : r_gencycle_x_allow

Function code: 0
Group code: 0
RVP codes

r_GenCycle_X_Dim

Description: Maps a X axis dimension for a generic cycle
Tech Tips: Selects a horizontal or X axis dimension value for use within a generic cycle. An example of using this register keyword includes the following:

"R2",real : r_gencycle_x_dim ; Fadal style or
"Q218=",real : r_gencycle_x_dim ; Heidenhain style

Function code: 0
Group code: 0
RVP codes

r_GenCycle_X_Pos

Description: Maps a X axis position for a generic cycle
Tech Tips: Selects a horizontal or X axis position value for use within a generic cycle. An example of using this register keyword includes the following:

"P110=K",real : r_gencycle_x_pos ; Fagor style or
"Q216=",real : r_gencycle_x_pos ; Heidenhain style

Function code: 0
Group code: 0
RVP codes

r_GenCycle_Y_Allow

Description: Maps a Y axis allowance for a generic cycle
Tech Tips: Selects a vertical or Y axis allowance value for use within a generic cycle. An example of using this register keyword includes the following:

"Y",real : r_gencycle_y_allow

Function code: 0
Group code: 0
RVP codes

r_GenCycle_Y_Dim

Description: Maps a Y axis dimension for a generic cycle
Tech Tips: Selects a vertical or Y axis dimension value for use within a generic cycle. An example of using this register keyword includes the following:

"R3",real : r_gencycle_y_dim ; Fadal style or
"Q219=",real : r_gencycle_y_dim ; Heidenhain style

Function code: 0
Group code: 0
RVP codes

r_GenCycle_Y_Pos

Description: Maps a Y axis position for a generic cycle
Tech Tips: Selects a vertical or Y axis position value for use within a generic cycle. An example of using this register keyword includes the following:

```
"P111=K",real        : r_gencycle_y_pos        ; Fagor style or
"Q217=",real         : r_gencycle_y_pos        ; Heidenhain style
```

Function code: 0
Group code: 0
RVP codes

r_GenCycle_Z_Allow

Description: Maps a Z axis allowance for a generic cycle
Tech Tips: Selects a depth or Z axis allowance value for use within a generic cycle. An example of using this register keyword includes the following:

```
"Z",real             : r_gencycle_z_allow
```

Function code: 0
Group code: 0
RVP codes

r_GenCycle_Z_Init

Description: Maps an initial Z axis value for a generic cycle
Tech Tips: Selects an initial depth or Z axis value for use within a generic cycle. An example of using this register keyword includes the following:

```
"P106=K",real        : r_gencycle_z_init       ; Fagor style or
"Q200=",real         : r_gencycle_z_init       ; Heidenhain style
```

Function code: 0
Group code: 0
RVP codes

r_GenCycle_Z_Peck

Description: Maps an peck value for a generic cycle
Tech Tips: Selects an peck depth or Z axis peck value for use within a generic cycle. An example of using this register keyword includes the following:

```
"P109=K",real        : r_gencycle_z_peck       ; Fagor style or
"Q202=",real         : r_gencycle_z_peck       ; Heidenhain style
```

Function code: 0
Group code: 0
RVP codes

r_GenCycle_Z_Pos

Description: Maps a Z axis position for a generic cycle
Tech Tips: Selects a depth or Z axis position value for use within a generic cycle. An example of using this register keyword includes the following:

```
"P106=K",real        : r_gencycle_z_pos        ; Fagor style or
"Q203=",real         : r_gencycle_z_pos        ; Heidenhain style
```

Function code: 0
Group code: 0
RVP codes

r_GenCycle_Z_Retract

Description: Maps a Z axis retract value for a generic cycle
Tech Tips: Selects a depth or Z axis retract value for use within a generic cycle. An example of using this register keyword includes the following:

"Q204=",real : r_gencycle_z_retract ; Heidenhain style

Function code: 0
Group code: 0
RVP codes

r_Global_Variable_Num

Description: Maps a global variable number
Tech Tips: Selects a global variable number. An example of using this register keyword includes the following:

"#",int : r_global_variable_num

Function code: 341
Group code: 0
RVP codes

r_Global_Variable_String

Description: Maps a global variable string
Tech Tips: Selects a global variable string. An example of using this register keyword includes the following:

"DECLARE=" : r_global_variable_string

Function code: 413
Group code: 0
RVP codes

r_Goto_Line

Description: Maps a line number to go or jump to
Tech Tips: Selects a line number or sequence number to go or jump to. An example of using this register keyword includes the following:

"N",int : r_goto_line

Function code: 0
Group code: 0
RVP codes

r_Groove_Amount

Description: Maps the groove amount
Tech Tips: Selects the groove amount. On most CNC controls this is a R parameter following a G87, G88 or G89 code. An example of using this register keyword includes the following:

"R",real : r_groove_amount ; Fanuc style

Function code: 0
Group code: 0
RVP codes

r_Groove_Feed

Description: Maps the grooving feed rate
Tech Tips: Selects the grooving cycle's feed rate. On most CNC controls this is a F paramter following a G87, G88 or G89 code. An example of using this register keyword includes the following:

"F",real : r_groove_feed ; Fanuc style

Function code: 0
Group code: 0
RVP codes

r_Groove_IncX

Description: Maps an incremental groove length in the X axis
Tech Tips: Selects an incremental groove length in the vertical or X axis. On most CNC controls this is a P parameter following a G87, G88 or G89 code. An example of using this register keyword includes the following:

"P",real : r_groove_incx ; Fanuc style

Function code: 0
Group code: 0
RVP codes

r_Groove_IncZ

Description: Maps an incremental groove length in the Z axis
Tech Tips: Selects an incremental groove length in the horizontal or Z axis. On most CNC controls this is a Q parameter following a G87, G88 or G89 code. An example of using this register keyword includes the following:

"Q",real : r_groove_incz ; Fanuc style

Function code: 0
Group code: 0
RVP codes

r_Groove_Radius

Description: Maps an groove radius value
Tech Tips: Selects a value for the groove radius. An example of using this register keyword includes the following:

"GR=",real : r_groove_radius

Function code: 0
Group code: 0
RVP codes

r_Groove_U

Description: Maps the U value for the groove cycle

Tech Tips: Selects the U value for the groove cycle. On most CNC controls this is a U parameter following a G87, G88 or G89 code. An example of using this register keyword includes the following:

"U",real : r_groove_u ; Fanuc style

Function code: 0
Group code: 0
RVP codes

r_Groove_V

Description: Maps the V value for the groove cycle
Tech Tips: Selects the V value for the groove cycle. On most CNC controls this is a W parameter following a G87, G88 or G89 code. An example of using this register keyword includes the following:

"W",real : r_groove_v ; Fanuc style

Function code: 0
Group code: 0
RVP codes

r_Groove_X

Description: Maps the X value for the groove cycle
Tech Tips: Selects the X value for the groove cycle. On most CNC controls this is a X parameter following a G87, G88 or G89 code. An example of using this register keyword includes the following:

"X",real : r_groove_x ; Fanuc style

Function code: 0
Group code: 0
RVP codes

r_Groove_Z

Description: Maps the Z value for the groove cycle
Tech Tips: Selects the Z value for the groove cycle. On most CNC controls this is a Z parameter following a G87, G88 or G89 code. An example of using this register keyword includes the following:

"Z",real : r_groove_z ; Fanuc style

Function code: 0
Group code: 0
RVP codes

r_Hole_Center_X

Description: Maps the X value for the center of a hole pattern cycle
Tech Tips: Selects the X value for the center of a hole pattern cycle. An example of using this register keyword includes the following:

"X",real : r_hole_center_x ; Vickers 2100 style

Function code: 0
Group code: 0

RVP codes

r_Hole_Center_Y

Description: Maps the Y value for the center of a hole pattern cycle
Tech Tips: Selects the Y value for the center of a hole pattern cycle. An example of using this register keyword includes the following:

"Y",real : r_hole_center_y ; Vickers 2100 style

Function code: 0
Group code: 0
RVP codes

r_Hole_Diam_Circ

Description: Maps the diameter value for the hole pattern cycle
Tech Tips: Selects the diameter value for the hole pattern cycle. An example of using this register keyword includes the following:

"D",real : r_hole_diam_circ ; Vickers 2100 style

Function code: 0
Group code: 0
RVP codes

r_Hole_First_X

Description: Maps the first hole center X value of a hole pattern cycle
Tech Tips: Selects the first hole center X value for a hole pattern cycle. An example of using this register keyword includes the following:

"I",real : r_hole_first_x ; Vickers 2100 style

Function code: 0
Group code: 0
RVP codes

r_Hole_First_Y

Description: Maps the first hole center Y value of a hole pattern cycle
Tech Tips: Selects the first hole center Y value for a hole pattern cycle. An example of using this register keyword includes the following:

"J",real : r_hole_first_y ; Vickers 2100 style

Function code: 0
Group code: 0
RVP codes

r_Hole_Int_Ang

Description: Maps the initial angle of the first hole in a hole pattern cycle
Tech Tips: Selects the initial angle of first hole in a hole pattern cycle. An example of using this register keyword includes the following:

"O",real : r_hole_int_ang ; Vickers 2100 style

Function code: 0
Group code: 0

RVP codes

r_Hole_Num

Description: Maps the number of holes in a hole pattern cycle
Tech Tips: Selects the number of holes in a hole pattern cycle. An example of using this register keyword includes the following:

"K",int : r_hole_num ; Vickers 2100 style

Function code: 0
Group code: 0
RVP codes

r_Hole_Retract

Description: Maps the retract plane in a hole pattern cycle
Tech Tips: Selects the retract plane in a hole pattern cycle. An example of using this register keyword includes the following:

"W",int : r_hole_retract ; Vickers 2100 style

Function code: 0
Group code: 0
RVP codes

r_Inches

Description: Maps inches
Tech Tips: Selects inches. An example of using this register keyword includes the following:

"IN" : r_inches ; Heidenhain style

Function code: 0
Group code: 0
RVP codes

r_Int_Ignore

Description: Maps an integer value to ignore
Tech Tips: Selects an integer value to ignore. An example of using this register keyword includes the following:

"N",int : r_int_ignore

Function code: 0
Group code: 0
RVP codes

r_IntVar_Local_Value

Description: Maps a local integer variable
Tech Tips: Selects a local integer variable. An example of using this register keyword includes the following:

"LV",int : r_intvar_local_value

Function code: 348
Group code: 0

RVP codes

r_IntVar_Global_Value

Description: Maps a global integer variable
Tech Tips: Selects a global integer variable. An example of using this register keyword includes the following:

"GV",int : r_intvar_global_value

Function code: 347
Group code: 0
RVP codes

r_IntVar_Param_Value

Description: Maps a parameter integer variable
Tech Tips: Selects a parameter integer variable. An example of using this register keyword includes the following:

"PV",int : r_intvar_param_value

Function code: 346
Group code: 0
RVP codes

r_IntVar_Value

Description: Maps an integer variable
Tech Tips: Selects an integer variable. An example of using this register keyword includes the following:

"IV",int : r_intvar_value

Function code: 344
Group code: 0
RVP codes

r_Last_Pos_X

Description: Maps the previous X register position
Tech Tips: Selects the last X or horizontal register position. On most CNC controls this is an X. An example of using this register keyword includes the following:

“X”,real : r_last_pos_x

Function code: 22
Group code: 0
RVP codes

r_Last_Pos_Y

Description: Maps the previous Y register position
Tech Tips: Selects the last Y or vertical register position. On most CNC controls this is an Y. An example of using this register keyword includes the following:

“Y”,real : r_last_pos_y

Function code: 23

Group code: 0
RVP codes

r_Last_Pos_Z

Description: Maps the previous Z register position
Tech Tips: Selects the last Z or depth register position. On most CNC controls this is an Z. An example of using this register keyword includes the following:

“Z”,real : r_last_pos_z

Function code: 24
Group code: 0
RVP codes

r_Length_Offset

Description: Maps the length offset register

Tech Tips: Selects the current length offset register. In most CNC formats this is one of the parameters following a G10 code. An example of using this register keyword includes the following:

“H”,real : r_diam_offset ; Fanuc style

Function code: 0
Group code: 0
RVP codes

r_Local_Variable_Num

Description: Maps a local variable number
Tech Tips: Selects a local variable number. An example of using this register keyword includes the following:

"#",int : r_local_variable_num

Function code: 342
Group code: 0
RVP codes

r_Local_Variable_String

Description: Maps a local variable string
Tech Tips: Selects a local variable string. An example of using this register keyword includes the following:

"DECLARE=” : r_local_variable_string

Function code: 414
Group code: 0
RVP codes

r_Millimeters

Description: Maps millimeters
Tech Tips: Selects millimeters. An example of using this register keyword includes the following:

"MM” : r_millimeters ; Heidenhain style

Function code: 0
Group code: 0
RVP codes

r_Min_Peck

Description: Maps the minimum amount to peck drill
Tech Tips: Selects the minimum amount to peck drill. An example of using this register keyword includes the following:

“MP=”,real : r_min_peck

Function code: 75
Group code: 0
RVP codes

r_Misc_MCH_FileName

Description: Maps the machine or .MCH file name value
Tech Tips: Selects the machine or .MCH file name value from within CNC comments. An example of using this register keyword includes the following:

%s,string_reg : r_misc_mch_filename ; Predator style

Function code: 0
Group code: 0
RVP codes

r_Num_Diam_Offset

Description: Maps the diameter offset number
Tech Tips: Selects the diameter offset number from within CNC comments. An example of using this register keyword includes the following:

%d,int_reg : r_num_diam_offset ; Predator style

Function code: 0
Group code: 0
RVP codes

r_Num_Length_Offset

Description: Maps the length offset number
Tech Tips: Selects the length offset number from within CNC comments. An example of using this register keyword includes the following:

%d,int_reg : r_num_length_offset ; Predator style

Function code: 0
Group code: 0
RVP codes

r_Orig_X

Description: Maps the origin in the X axis
Tech Tips: Selects the X axis origin. An example of using this register keyword includes the following:

“X”,real : r_orig_x ; Seimens style

Function code: 0
Group code: 0
RVP codes

r_Orig_Y

Description: Maps the origin in the Y axis
Tech Tips: Selects the Y axis origin. An example of using this register keyword includes the following:

“Y”,real : r_orig_y ; Seimens style

Function code: 0
Group code: 0
RVP codes

r_Orig_Z

Description: Maps the origin in the Z axis
Tech Tips: Selects the Z axis origin. An example of using this register keyword includes the following:

“Z”,real : r_orig_z ; Seimens style

Function code: 0
Group code: 0
RVP codes

r_Orig_4th_axis

Description: Maps the origin in the primary rotary axis
Tech Tips: Selects the primary rotary axis origin. An example of using this register keyword includes the following:

“B”,real : r_orig_4th_axis ; Seimens style

Function code: 0
Group code: 0
RVP codes

r_Orig_5th_axis

Description: Maps the origin in the secondary rotary axis
Tech Tips: Selects the secondary rotary axis origin. An example of using this register keyword includes the following:

“A”,real : r_orig_5th_axis ; Seimens style

Function code: 0
Group code: 0
RVP codes

r_Param_Variable_Num

Description: Maps a parameter variable number
Tech Tips: Selects a parameter variable number. An example of using this register keyword includes the following:

"PN”,int : r_param_variable_num

Function code: 340
Group code: 0
RVP codes

r_Param_Variable_String

Description: Maps a parameter variable string
Tech Tips: Selects a parameter variable string. An example of using this register keyword includes the following:

"PS",string_reg : r_param_variable_string

Function code: 412
Group code: 0
RVP codes

r_Params

Description: Maps parameters
Tech Tips: Selects a parameter. An example of using this register keyword includes the following:

"P",int : r_params

Function code: 0
Group code: 0
RVP codes

r_Peck_Dec

Description: Maps the peck drill decrement register
Tech Tips: Selects the current peck drill decrement register. An example of using this register keyword includes the following:

“QD=”,real : r_peck_dec

Function code: 75
Group code: 0
RVP codes

r_Peck_Inc

Description: Maps the peck drill increment register
Tech Tips: Selects the current Q or peck drill increment register position. On most CNC controls this is an Q. An example of using this register keyword includes the following:

“Q”,real : r_peck_inc ; Fanuc style or
“Peck~”,real : r_peck_inc ; Anilam style

Function code: 75
Group code: 0
RVP codes

r_Peck_Retract

Description: Maps the peck drill retract register position

Tech Tips: Selects the current R or peck drill retract register position. On most CNC controls this is an R. An example of using this register keyword includes the following:

```
"R",real              : r_peck_retract        ; Fanuc style or
"ReturnHgt~",real     : r_peck_retract        ; Anilam style or
"R03*",real           : r_peck_retract        ; Seimens style
```

Function code: 74
Group code: 0
RVP codes

r_Peck_Repeats

Description: Maps the canned cycle number of repeats
Tech Tips: Selects the current canned cycle number of repeats. An example of using this register keyword includes the following:

```
"L",real              : r_peck_repeats        ; Haas style
```

Function code: 241
Group code: 0
RVP codes

r_Peck_Z

Description: Maps the Z peck drill register position
Tech Tips: Selects the current Z or peck drill register position. On most CNC controls this is an Z. An example of using this register keyword includes the following:

```
"Z",real              : r_peck_z              ; Fanuc style or
"ZDepth~",real        : r_peck_z              ; Anilam style or
"R02*",real           : r_peck_z              ; Seimens style
```

Function code: 73
Group code: 0
RVP codes

r_Polar_Angle

Description: Maps the polar angle
Tech Tips: Selects the polar angle for polar interpolation. An example of using this register keyword includes the following:

```
"PA",real             : r_polar_angle         ; Heidenhain style
```

Function code: 0
Group code: 0
RVP codes

r_Polar_Center_X

Description: Maps the horizontal center axis for polar interpolation
Tech Tips: Selects the horizontal center axis for polar interpolation. An example of using this register keyword includes the following:

```
"X",real              : r_polar_center_x      ; Seimens style
```

Function code: 0
Group code: 0
RVP codes

r_Polar_Center_Y

Description: Maps the vertical center axis for polar interpolation
Tech Tips: Selects the vertical center axis for polar interpolation. An example of using this register keyword includes the following:

"Y",real : r_polar_center_y ; Seimens style

Function code: 0
Group code: 0
RVP codes

r_Polar_Center_Z

Description: Maps the depth center axis for polar interpolation
Tech Tips: Selects the depth center axis for polar interpolation. An example of using this register keyword includes the following:

"Z,real : r_polar_center_z ; Seimens style

Function code: 0
Group code: 0
RVP codes

r_Polar_IncAngle

Description: Maps an incremental polar angle
Tech Tips: Selects an incremental polar angle for polar interpolation. An example of using this register keyword includes the following:

“IPA”,real : r_polar_angle ; Heidenhain style

Function code: 0
Group code: 0
RVP codes

r_Polar_Radius

Description: Maps the polar radius
Tech Tips: Selects the polar radius for polar interpolation. An example of using this register keyword includes the following:

“PR”,real : r_polar_radius ; Heidenhain style

Function code: 0
Group code: 0
RVP codes

r_Pos_4thAxis

Description: Maps the primary rotary axis register position
Tech Tips: Selects the primary rotary axis register position. On most CNC controls this is an A axis. An example of using this register keyword includes the following:

“A”,real : r_pos_4thaxis

Function code: 8
Group code: 0
RVP codes

r_Pos_5thAxis

Description: Maps the secondary rotary axis register position
Tech Tips: Selects the secondary rotary axis register position. On most CNC controls this is an B or C axis. An example of using this register keyword includes the following:

"B",real : r_pos_5thaxis

Function code: 8
Group code: 0
RVP codes

r_Pos_X

Description: Maps the X register position
Tech Tips: Selects the current X or horizontal register position. On most CNC controls this is an X. An example of using this register keyword includes the following:

"X",real : r_pos_x

Function code: 8
Group code: 0
RVP codes

r_Pos_Y

Description: Maps the Y register position
Tech Tips: Selects the current Y or vertical register position. On most CNC controls this is an Y. An example of using this register keyword includes the following:

"Y",real : r_pos_y

Function code: 9
Group code: 0
RVP codes

r_Pos_Z

Description: Maps the Z register position
Tech Tips: Selects the current Z or depth register position. On most CNC controls this is an Z. An example of using this register keyword includes the following:

"Z",real : r_pos_z

Function code: 10
Group code: 0
RVP codes

r_Pos_I

Description: Maps the I register position

Tech Tips: Selects the current I or horizontal circular center register position. On most CNC controls this is an I. An example of using this register keyword includes the following:

"I",real : r_pos_i

Function code: 4
Group code: 0
RVP codes

r_Pos_J

Description: Maps the J register position
Tech Tips: Selects the current J or vertical circular center register position. On most CNC controls this is an J. An example of using this register keyword includes the following:

"I",real : r_pos_i

Function code: 5
Group code: 0
RVP codes

r_Pos_K

Description: Maps the K register position
Tech Tips: Selects the current K or depth circular center register position. On most CNC controls this is an K. An example of using this register keyword includes the following:

"K",real : r_pos_k

Function code: 6
Group code: 0
RVP codes

r_Pos_U

Description: Maps the incremental vertical lathe register position
Tech Tips: Selects the incremental vertical lathe register position. On most CNC controls this is an U. An example of using this register keyword includes the following:

"U",real : r_pos_u

Function code: 0
Group code: 0
RVP codes

r_Pos_V

Description: Maps the incremental horizontal lathe register position
Tech Tips: Selects the incremental horizontal lathe register position. On most CNC controls this is an W. An example of using this register keyword includes the following:

"W",real : r_pos_v

Function code: 0

Group code: 0
RVP codes

r_Prog_Name

Description: Maps the program name
Tech Tips: Selects the current program name. An example of using this register keyword includes the following:

%s,string_reg : r_prog_name ; Heidenhain style

Function code: 0
Group code: 0
RVP codes

r_Real_Ignore

Description: Maps a real number to ignore
Tech Tips: Selects a real number to ignore. An example of using this register keyword includes the following:

“U”,real : r_real_ignore

Function code: 0
Group code: 0
RVP codes

r_RealVar_Param_Value

Description: Maps a parameter real value
Tech Tips: Selects a parameter real value. An example of using this register keyword includes the following:

"RP”,real_reg : r_realvar_param_value

Function code: 140
Group code: 0
RVP codes

r_RealVar_Global_Value

Description: Maps a global real value
Tech Tips: Selects a global real value. An example of using this register keyword includes the following:

"RG”,real_reg : r_realvar_global_value

Function code: 141
Group code: 0
RVP codes

r_RealVar_Local_Value

Description: Maps a local real value
Tech Tips: Selects a local real value. An example of using this register keyword includes the following:

"RL”,real_reg : r_realvar_local_value

Function code: 142

Group code: 0
RVP codes

r_RealVar_Value

Description: Maps a real value
Tech Tips: Selects a real value. An example of using this register keyword includes the following:

"RV",real_reg : r_realvar_value

Function code: 144
Group code: 0
RVP codes

r_Reference_Pos_4th

Description: Maps the primary rotary reference register position
Tech Tips: Selects the current A, B or C secondary horizontal reference register position. An example of using this register keyword includes the following:

“A”,real : r_reference_pos_4th

NOTE: For backwards compatibility r_pos_4thaxis can also be used.

Function code: 104
Group code: 0
RVP codes

r_Reference_Pos_5th

Description: Maps the secondary rotary reference register position
Tech Tips: Selects the current A, B or C secondary horizontal reference register position. An example of using this register keyword includes the following:

“B”,real : r_reference_pos_5th

NOTE: For backwards compatibility r_pos_4thaxis can also be used.

Function code: 104
Group code: 0
RVP codes

r_Reference_Pos_U

Description: Maps the U reference register position
Tech Tips: Selects the current U or secondary horizontal reference register position. An example of using this register keyword includes the following:

“U”,real : r_reference_pos_u

Function code: 104
Group code: 0
RVP codes

r_Reference_Pos_V

Description: Maps the V reference register position
Tech Tips: Selects the current V or secondary vertical reference register position. An example of using this register keyword includes the following:

"V",real : r_reference_pos_v

Function code: 105
Group code: 0
RVP codes

r_Reference_Pos_W

Description: Maps the W reference register position
Tech Tips: Selects the current W or secondary depth reference register position. An example of using this register keyword includes the following:

"W",real : r_reference_pos_w

Function code: 106
Group code: 0
RVP codes

r_Reference_Pos_X

Description: Maps the X reference register position
Tech Tips: Selects the current X or horizontal reference register position. On most CNC controls this is a X parameter following a G28 or G29. An example of using this register keyword includes the following:

"X",real : r_reference_pos_x ; Fanuc style

Function code: 101
Group code: 0
RVP codes

r_Reference_Pos_Y

Description: Maps the Y reference register position
Tech Tips: Selects the current Y or vertical reference register position. On most CNC controls this is a Y parameter following a G28 or G29. An example of using this register keyword includes the following:

"Y",real : r_reference_pos_y ; Fanuc style

Function code: 102
Group code: 0
RVP codes

r_Reference_Pos_Z

Description: Maps the Z reference register position
Tech Tips: Selects the current Z or depth reference register position. On most CNC controls this is a Z parameter following a G28 or G29. An example of using this register keyword includes the following:

"Z",real : r_reference_pos_z ; Fanuc style

Function code: 103
Group code: 0
RVP codes

r_Retract_Mode

Description: Maps the retract mode
Tech Tips: Selects the drill cycle's retract mode between the initial plane or retract plane. On most CNC controls this is a G98 or G99 code. An example of using this register keyword includes the following:

```
"",switch@4,casesens   : r_retract_mode          ; Fanuc style
```

Function code: 244
Group code: 0
RVP codes

r_Rot_Angle

Description: Maps the rotation angle
Tech Tips: Selects the rotation angle for rotation mode. An example of using this register keyword includes the following:

```
"R",real               : r_rot_angle             ; Fanuc style or
"ROT+",real            : r_rot_angle             ; Heidenhain style
```

Function code: 0
Group code: 0
RVP codes

r_Rot_Angle_X

Description: Maps the rotation angle in the X axis
Tech Tips: Selects the rotation value for rotation in the X axis. An example of using this register keyword includes the following:

```
"X",real               : r_rot_angle_x           ; Seimens style
```

Function code: 0
Group code: 0
RVP codes

r_Rot_Angle_Y

Description: Maps the rotation angle in the Y axis
Tech Tips: Selects the rotation value for rotation in the Y axis. An example of using this register keyword includes the following:

```
"Y",real               : r_rot_angle_Y           ; Seimens style
```

Function code: 0
Group code: 0
RVP codes

r_Rot_Angle_Z

Description: Maps the rotation angle in the Z axis
Tech Tips: Selects the rotation value for rotation in the Z axis. An example of using this register keyword includes the following:

```
"Z",real               : r_rot_angle_z           ; Seimens style
```

Function code: 0
Group code: 0

RVP codes

r_Rot_Center_X

Description: Maps the X axis for the center of rotation
Tech Tips: Selects the horizontal or X axis value for the center point of rotation for rotation mode. An example of using this register keyword includes the following:

```
"X",real        : r_rot_center_x        ; Fanuc style
```

Function code: 0
Group code: 0
RVP codes

r_Rot_Center_Y

Description: Maps the Y axis for the center of rotation
Tech Tips: Selects the vertical or Y axis value for the center point of rotation for rotation mode. An example of using this register keyword includes the following:

```
"Y",real        : r_rot_center_y        ; Fanuc style
```

Function code: 0
Group code: 0
RVP codes

r_Scal_Factor

Description: Maps the scale factor
Tech Tips: Selects the scale factor for scale mode. An example of using this register keyword includes the following:

```
"P",real        : r_scal_factor         ; Fanuc style or
"SCL",real      : r_scal_factor         ; Heidenhain style
```

Function code: 0
Group code: 0
RVP codes

r_Scale_Factor_X

Description: Maps the scale factor in the X axis
Tech Tips: Selects the scale factor for scale mode in the X axis. An example of using this register keyword includes the following:

```
"X",real        : r_scale_factor_x      ; Seimens style
```

Function code: 0
Group code: 0
RVP codes

r_Scale_Factor_Y

Description: Maps the scale factor in the Y axis
Tech Tips: Selects the scale factor for scale mode in the Y axis. An example of using this register keyword includes the following:

```
"Y",real        : r_scale_factor_y      ; Seimens style
```

Function code: 0
Group code: 0
RVP codes

r_Scale_Factor_Z

Description: Maps the scale factor in the Z axis
Tech Tips: Selects the scale factor for scale mode in the Z axis. An example of using this register keyword includes the following:

```
"Z",real            : r_scale_factor_z          ; Seimens style
```

Function code: 0
Group code: 0
RVP codes

r_Scal_Center_X

Description: Maps the X axis for the center for scaling
Tech Tips: Selects the horizontal or X axis value for the center point of scaling for scale mode. An example of using this register keyword includes the following:

```
"X",real            : r_scal_center_x           ; Fanuc style
```

Function code: 0
Group code: 0
RVP codes

r_Scal_Center_Y

Description: Maps the Y axis for the center for scaling
Tech Tips: Selects the vertical or Y axis value for the center point of scaling for scale mode. An example of using this register keyword includes the following:

```
"Y",real            : r_scal_center_y           ; Fanuc style
```

Function code: 0
Group code: 0
RVP codes

r_Scal_Center_Z

Description: Maps the Z axis for the center for scaling
Tech Tips: Selects the depth or Z axis value for the center point of scaling for scale mode. An example of using this register keyword includes the following:

```
"Z",real            : r_scal_center_z           ; Fanuc style
```

Function code: 0
Group code: 0
RVP codes

r_Spindle_Speed

Description: Maps spindle speed
Tech Tips: Selects the spindle speed. An example of using this register keyword includes the following:

"S",real : r_spindle_speed ; Fanuc style

Function code: 0
Group code: 0
RVP codes

r_Stock_Box

internal function code: 0
internal functionality group code:0
description

RVP codes

r_Stock_Box_Type

internal function code: 0
internal functionality group code:0
description

RVP codes

r_Stock_CenterX

Description: Maps the X axis for the center of cylindrical stock and fixtures
Tech Tips: Selects the horizontal or X axis position for the center point of cylindrical stock and fixture shapes defined within CNC comments. An example of using this register keyword includes the following:

"X",real : r_stock_centerx ; Predator style

Function code: 0
Group code: 0
RVP codes

r_Stock_CenterY

Description: Maps the Y axis for the center of cylindrical stock and fixtures
Tech Tips: Selects the vertical or Y axis position for the center point of cylindrical stock and fixture shapes defined within CNC comments. An example of using this register keyword includes the following:

"Y",real : r_stock_centery ; Predator style

Function code: 0
Group code: 0
RVP codes

r_Stock_CenterZ

Description: Maps the Z axis for the center of cylindrical stock and fixtures
Tech Tips: Selects the depth or Z axis position for the center point of cylindrical stock and fixture shapes defined within CNC comments. An example of using this register keyword includes the following:

"Z",real : r_stock_centerz ; Predator style

Function code: 0
Group code: 0
RVP codes

r_Stock_Cyl

internal function code: 0
internal functionality group code:0
description

RVP codes

r_Stock_Cyl_Axis

Description: Maps the axis direction for cylindrical stock and fixtures
Tech Tips: Selects the X, Y or Z axis direction for cylindrical stock and fixture shapes defined within CNC comments. An example of using this register keyword includes the following:

“S”,int : r_stock_cyl_axis ; Predator style

NOTE: Using the above example, an S1 defines a X axis cylinder, S2 for a Y axis cylinder and S3 for a Z axis cylinder.

Function code: 0
Group code: 0
RVP codes

r_Stock_Cyl_Type

internal function code: 0
internal functionality group code:0
description

RVP codes

r_Stock_Diameter

Description: Maps the diameter for cylindrical stock and fixtures
Tech Tips: Selects the diameter for cylindrical stock and fixture shapes defined within CNC comments. An example of using this register keyword includes the following:

“D”,real : r_stock_diameter ; Predator style

Function code: 0
Group code: 0
RVP codes

r_Stock_Length

Description: Maps the length for cylindrical stock and fixtures
Tech Tips: Selects the length for cylindrical stock and fixture shapes defined within CNC comments. An example of using this register keyword includes the following:

“L”,real : r_stock_length ; Predator style

Function code: 0
Group code: 0
RVP codes

r_Stock_LengthX

Description: Maps the X axis length for box stock and fixtures
Tech Tips: Selects the horizontal or X axis value for box or rectangular stock and fixture shapes defined within CNC comments. An example of using this register keyword includes the following:

“X”,real : r_stock_lengthx ; Predator style

Function code: 0
Group code: 0
RVP codes

r_Stock_LengthY

Description: Maps the Y axis length for box stock and fixtures
Tech Tips: Selects the vertical or Y axis value for box or rectangular stock and fixture shapes defined within CNC comments. An example of using this register keyword includes the following:

“Y”,real : r_stock_lengthy ; Predator style

Function code: 0
Group code: 0
RVP codes

r_Stock_LengthZ

Description: Maps the Z axis length for box stock and fixtures
Tech Tips: Selects the horizontal or Z axis value for box or rectangular stock and fixture shapes defined within CNC comments. An example of using this register keyword includes the following:

“Z”,real : r_stock_lengthz ; Predator style

Function code: 0
Group code: 0
RVP codes

r_Stock_MinX

Description: Maps the X axis position for box stock and fixtures
Tech Tips: Selects the horizontal or X axis position for box or rectangular stock and fixture shapes defined within CNC comments. An example of using this register keyword includes the following:

“X”,real : r_stock_minx ; Predator style

Function code: 0
Group code: 0
RVP codes

r_Stock_MinY

Description: Maps the Y axis position for box stock and fixtures
Tech Tips: Selects the vertical or Y axis position for box or rectangular stock and fixture shapes defined within CNC comments. An example of using this register keyword includes the following:

“Y”,real : r_stock_miny ; Predator style

Function code: 0
Group code: 0
RVP codes

r_Stock_MinZ

Description: Maps the Z axis position for box stock and fixtures
Tech Tips: Selects the depth or Z axis position for box or rectangular stock and fixture shapes defined within CNC comments. An example of using this register keyword includes the following:

“Z”,real : r_stock_minz ; Predator style

Function code: 0
Group code: 0
RVP codes

r_Stock_Type

internal function code: 0
internal functionality group code:0
description

RVP codes

r_Stock_STL

internal function code: 0
internal functionality group code:0
description

RVP codes

r_Stock_STL_Filename

Description: Maps the .STL file name for complex stock and fixtures
Tech Tips: Selects the .STL file name for complex stock and fixtures, such as castings, from within CNC comments. An example of using this register keyword includes the following:

%s,string_reg : r_stock_stl_filename ; Predator style

Function code: 0
Group code: 0
RVP codes

r_Stock_STL_Type

internal function code: 0
internal functionality group code:0
description

RVP codes

r_Stock_STK_Filename

Description: Maps stock and fixture library or .STK file names
Tech Tips: Selects the stock and fixture library or .STK file names for complex assemblies of multiple stock and fixture components from within CNC

comments. An example of using this register keyword includes the following:

%s,string_reg : r_stock_STK_filename ; Predator style

Function code: 0
Group code: 0
RVP codes

r_String_Ignore

Description: Maps a string to ignore
Tech Tips: Selects a alphanumeric string to ignore. An example of using this register keyword includes the following:

“PARTNO=”,string_reg : r_string_ignore

Function code: 0
Group code: 0
RVP codes

r_StringVar_Local_Value

Description: Maps a local string value
Tech Tips: Selects a local string value. An example of using this register keyword includes the following:

"SL”,string_reg : r_stringvar_local_value

Function code: 419
Group code: 0
RVP codes

r_StringVar_Global_Value

Description: Maps a global string value
Tech Tips: Selects a global string value. An example of using this register keyword includes the following:

"SG”,string_reg : r_stringvar_global_value

Function code: 418
Group code: 0
RVP codes

r_StringVar_Param_Value

Description: Maps a parameter string value
Tech Tips: Selects a parameter string value. An example of using this register keyword includes the following:

"SP”,string_reg : r_stringvar_param_value

Function code: 417
Group code: 0
RVP codes

r_StringVar_Value

Description: Maps a variable string value

Tech Tips: Selects a variable string value. An example of using this register keyword includes the following:

"SV",string_reg : r_stringvar_value

Function code: 416
Group code: 0
RVP codes

r_Subprogram_EndLineRet

internal function code: 0
internal functionality group code:0
description

RVP codes

r_SubProgram_EndrsTimes

internal function code: 0
internal functionality group code:0
description

RVP codes

r_SubProgram_GotoLine

Description: Maps a line number to go or jump to within a sub program call
Tech Tips: Selects a line number or sequence number to go or jump to inside a sub program when it is called. An example of using this register keyword includes the following:

"N",int : r_subprogram_gotoline

Function code: 0
Group code: 0
RVP codes

r_SubProgram_FirstLine

Description: Maps a starting line number to go or jump to within a sub program call
Tech Tips: Selects a starting line number or sequence number to go or jump to inside a sub program when it is called. An example of using this register keyword includes the following:

"P",int : r_subprogram_firstline

Function code: 0
Group code: 0
RVP codes

r_SubProgram_LastLine

Description: Maps an ending line number to go or jump to within a sub program call
Tech Tips: Selects an ending line number or sequence number to go or jump to inside a sub program when it is called. An example of using this register keyword includes the following:

"Q",int : r_subprogram_lastline

Function code: 0
Group code: 0
RVP codes

r_SubProgram_Number

Description: Maps the subroutine or sub program number
Tech Tips: Selects the current subroutine or sub program number. On most CNC controls this is an O number. An example of using this register keyword includes the following:

```
"P",real        : r_subprogram_start        ; Fanuc style or
"O",real        : r_subprogram_start        ; Okuma style
```

Function code: 324
Group code: 0
RVP codes

r_SubProgram_Start

Description: Maps the subroutine or sub program start number
Tech Tips: Selects the current subroutine or sub program start number. On most CNC controls this is an N number. An example of using this register keyword includes the following:

```
 "N",int        : r_subprogram_start        ; Fagor style
```

Function code: 320
Group code: 0
RVP codes

r_SubProgram_StrStart

Description: Maps the subroutine or sub program start string
Tech Tips: Selects the current subroutine or sub program start string. An example of using this register keyword includes the following:

```
%s, string_reg      : r_subprogram_strstart
```

Function code: 410
Group code: 0
RVP codes

r_SubProgram_String

Description: Maps the subroutine or sub program name string
Tech Tips: Selects the current subroutine or sub program name string. An example of using this register keyword includes the following:

```
%s, int_string      : r_subprogram_string       ; Vickers 2100 style
```

Function code: 411
Group code: 0
RVP codes

r_SubProgram_Times

Description: Maps the number of subroutine or sub program repetitions

Tech Tips: Selects the current number of subroutine or sub program repetitions. On most CNC controls this is a L number. An example of using this register keyword includes the following:

```
"L",int          : r_subprogram_times    ; Fanuc style or
%d, int_reg      : r_subprogram_times    ; Vickers 2100 style or
"Q",int          : r_subprogram_times    ; Okuma style
```

Function code: 325
Group code: 0
RVP codes

r_Tapper_Radius

Description: Maps the tapper radius
Tech Tips: Selects the current tapper radius. An example of using this register keyword includes the following:

```
"R",real         : r_tapper_radius       ; G & L style
"I",real         : r_tapper_radius
```

Function code: 0
Group code: 0
RVP codes

r_Thread_Angle

Description: Maps the thread angle
Tech Tips: Selects the thread angle. An example of using this register keyword includes the following:

```
"A",real         : r_thread_angle        ; Okuma style
```

Function code: 0
Group code: 0
RVP codes

r_Thread_Allowance

Description: Maps the thread allowance
Tech Tips: Selects the thread allowance. On most CNC controls this is an R. An example of using this register keyword includes the following:

```
"R",real         : r_thread_allowance    ; Fanuc style or
"U",real         : r_thread_allowance    ; Okuma style
"K",real         : r_thread_allowance
```

Function code: 0
Group code: 0
RVP codes

r_Thread_Feed

Description: Maps the thread feed rate
Tech Tips: Selects the current thread feed rate or F register position. On most CNC controls this is an F. An example of using this register keyword includes the following:

```
"F",real         : r_thread_feed         ; Fanuc style or
```

```
"K",real          : r_thread_feed          ; Okuma style or
"E",real          : r_thread_feed          ; Allen Bradley style or
"R",real          : r_thread_feed          ; Cincinnati style or
"TPIorLead~",real : r_thread_feed          ; Anilam style
```

Function code: 112
Group code: 0
RVP codes

r_Thread_First_Depth

Description: Maps the first depth of cut for threading
Tech Tips: Selects the depth of cut for the first move during threading. On most CNC controls this is an Q. An example of using this register keyword includes the following:

```
"Q",real          : r_thread_first_depth   ; Fanuc style or
"D",real          : r_thread_first_depth   ; Okuma style
```

Function code: 0
Group code: 0
RVP codes

r_Thread_MinLen

Description: Maps the minimum length for threading
Tech Tips: Selects the minimum length for threading. An example of using this register keyword includes the following:

```
"L",real          : r_thread_minlen
```

Function code: 0
Group code: 0
RVP codes

r_Thread_NTimes

Description: Maps the number of times for threading
Tech Tips: Selects the number of times for threading. An example of using this register keyword includes the following:

```
"N",int           : r_thread_ntimes
```

Function code: 0
Group code: 0
RVP codes

r_Thread_Pos_X

Description: Maps the X thread register position
Tech Tips: Selects the current X or vertical thread register position. On most CNC lathe controls this is an X. An example of using this register keyword includes the following:

```
"X",real          : r_thread_pos_x         ; Fanuc style
```

Function code: 110
Group code: 0
RVP codes

r_Thread_Pos_U

Description: Maps the incremental vertical thread register position
Tech Tips: Selects the current U or vertical thread register position. On most CNC lathe controls this is an U. An example of using this register keyword includes the following:

```
"U",real        : r_thread_pos_u        ; Haas style
```

Function code: 111
Group code: 0
RVP codes

r_Thread_Pos_V

Description: Maps the incremental horizontal thread register position
Tech Tips: Selects the current W or horizontal thread register position. On most CNC lathe controls this is an W. An example of using this register keyword includes the following:

```
"W",real        : r_thread_pos_v        ; Haas style
```

Function code: 111
Group code: 0
RVP codes

r_Thread_Pos_Z

Description: Maps the Z thread register position
Tech Tips: Selects the current Z or horizontal thread register position. On most CNC lathe controls this is an Z. An example of using this register keyword includes the following:

```
"Z",real        : r_thread_pos_z        ; Fanuc style
```

Function code: 111
Group code: 0
RVP codes

r_Thread_Total_Depth

Description: Maps the total depth of cut for threading
Tech Tips: Selects the total depth of cut for threading. On most CNC controls this is an P. An example of using this register keyword includes the following:

```
"P",real        : r_thread_total_depth  ; Fanuc style or
"H",real        : r_thread_total_depth  ; Okuma style or
"K",real        : r_thread_total_depth  ; Allen Bradley style
```

Function code: 0
Group code: 0
RVP codes

r_Thread_Taper_Inc

Description: Maps the taper increment for threading
Tech Tips: Selects the taper increment during threading. An example of using this register keyword includes the following:

"I",real : r_thread_taper_inc ; Allen Bradley style

Function code: 0
Group code: 0
RVP codes

r_Tool_Alpha

Description: Maps the tool's alpha value
Tech Tips: Selects the current tool alpha value. Often this register keyword is used for processing APTCL instead of a CNC program. An example of using this register keyword includes the following:

%f,real : r_tool_alpha

Function code: 35
Group code: 0
RVP codes

r_Tool_Angle

Description: Maps the tool angle
Tech Tips: Selects the current tool angle. With the Predator style tool comments this is an A. An example of using this register keyword includes the following:

"A",real : r_tool_angle ; Predator style

Function code: 0
Group code: 0
RVP codes

r_Tool_Ang_Orient

Description: Maps the tool insert orientation
Tech Tips: Selects the current tool insert orientation within the tool holder. With the Predator style tool comments this is an O. An example of using this register keyword includes the following:

"O",real : r_tool_ang_orient ; Predator style

Function code: 0
Group code: 0
RVP codes

r_Tool_Angle_Top

Description: Maps the tool's top tool angle
Tech Tips: Selects the current tool's top tool angle. An example of using this register keyword includes the following:

"T",real : r_tool_angle_top

Function code: 0
Group code: 0
RVP codes

r_Tool_Aux

Description: Maps the tool auxiliary

Tech Tips: Selects the current tool auxiliary value. Often this register keyword is used for processing APTCL instead of a CNC program. An example of using this register keyword includes the following:

%f,real : r_tool_aux

NOTE: This register is now obsolete.

Function code: 36
Group code: 0
RVP codes

r_Tool_Ax_Crad_Offset

Description: Maps the tool's corner radius offset
Tech Tips: Selects the current tool's corner radius offset. An example of using this register keyword includes the following:

"O",real : r_tool_ax_crad_offset

Function code: 0
Group code: 0
RVP codes

r_Tool_BC_Crad_Offset

Description: Maps the tool's corner radius offset
Tech Tips: Selects the current tool's corner radius offset. An example of using this register keyword includes the following:

"O",real : r_tool_bc_crad_offset

Function code: 0
Group code: 0
RVP codes

r_Tool_Corner_Rad

Description: Maps the tool corner radius
Tech Tips: Selects the current tool corner radius. With the Predator style tool comments this is an C. An example of using this register keyword includes the following:

"C",real : r_tool_corner_rad

Function code: 34
Group code: 0
RVP codes

r_Tool_Curr_Alpha

Description: Maps the tool's current alpha
Tech Tips: Selects the current tool's alpha value. Often this register keyword is used for processing APTCL instead of a CNC program. An example of using this register keyword includes the following:

%f,real : r_tool_curr_alpha

Function code: 0

Group code: 0
RVP codes

r_Tool_Curr_Aux

Description: Maps the tool's current auxiliary
Tech Tips: Selects the current tool's auxiliary value. Often this register keyword is used for processing APTCL instead of a CNC program. An example of using this register keyword includes the following:

%f,real : r_tool_curr_aux

Function code: 0
Group code: 0
RVP codes

r_Tool_Curr_Corner_R

Description: Maps the tool's current corner radius
Tech Tips: Selects the current tool's corner radius value. Often this register keyword is used for processing APTCL instead of a CNC program. An example of using this register keyword includes the following:

%f,real : r_tool_curr_corner_r

Function code: 0
Group code: 0
RVP codes

r_Tool_Curr_Diam

Description: Maps the tool's current diameter
Tech Tips: Selects the current tool's diameter value. Often this register keyword is used for processing APTCL instead of a CNC program. An example of using this register keyword includes the following:

%f,real : r_tool_curr_diam

Function code: 0
Group code: 0
RVP codes

r_Tool_Curr_Height

Description: Maps the tool's current height
Tech Tips: Selects the current tool's height value. Often this register keyword is used for processing APTCL instead of a CNC program. An example of using this register keyword includes the following:

%f,real : r_tool_curr_height

Function code: 0
Group code: 0
RVP codes

r_Tool_Cut_Dir

Description: Maps the tool's cut direction
Tech Tips: Selects the current tool's cut direction. An example of using this register keyword includes the following:

%f,real : r_tool_cut_dir

Function code: 0
Group code: 0
RVP codes

r_Tool_Diameter

Description: Maps the tool diameter
Tech Tips: Selects the current tool diameter. With the Predator style tool comments this is an D. An example of using this register keyword includes the following:

"D",real : r_tool_diameter ; Predator style

Function code: 33
Group code: 0
RVP codes

r_Tool_Height

Description: Maps the current tool height
Tech Tips: Selects the current tool flute height. With the Predator style tool comments this is an H. An example of using this register keyword includes the following:

"H",real : r_tool_height ; Predator style

Function code: 37
Group code: 0
RVP codes

r_Tool_Holder_Angle

Description: Maps the current tool holder angle
Tech Tips: Selects the current tool holder angle. With the Predator style tool comments this is an HA. An example of using this register keyword includes the following:

"HA",real : r_tool_holder_angle ; Predator style

Function code: 0
Group code: 0
RVP codes

r_Tool_Holder_D

Description: Maps the current tool holder diameter
Tech Tips: Selects the current tool holder diameter. With the Predator style tool comments this is an HD. An example of using this register keyword includes the following:

"HD",real : r_tool_holder_d ; Predator style

Function code: 0
Group code: 0
RVP codes

r_Tool_Holder_Tip

Description: Maps the current tool holder tip
Tech Tips: Selects the current tool holder tip. With the Predator style tool comments this is an HP. An example of using this register keyword includes the following:

“HP”,real : r_tool_holder_tip ; Predator style

Function code: 0
Group code: 0
RVP codes

r_Tool_Holder_Type

Description: Maps the current tool holder type
Tech Tips: Selects the current tool holder type. With the Predator style tool comments this is an HT. An example of using this register keyword includes the following:

“HT”,int : r_tool_holder_type ; Predator style

NOTE: HT1 = Flat style, FT2 = Ball style, FT3 = Bull nose style, and FT4 = chamfer style.

Function code: 0
Group code: 0
RVP codes

r_Tool_Holder_H

Description: Maps the current tool holder height
Tech Tips: Selects the current tool holder height. With the Predator style tool comments this is an HH. An example of using this register keyword includes the following:

“HH”,real : r_tool_holder_h ; Predator style

Function code: 0
Group code: 0
RVP codes

r_Tool_Home_Num

Description: Maps the current tool home number
Tech Tips: Selects the current tool home number. With the Predator style tool comments this is an M. An example of using this register keyword includes the following:

“M”,real : r_tool_home_num ; Predator style

Function code: 0
Group code: 0
RVP codes

r_Tool_Ins_Circle

Description: Maps the current tool insert’s inscribed circle
Tech Tips: Selects the current tool insert’s inscribed circle. With the Predator style tool comments this is an I. An example of using this register keyword includes the following:

"I",real : r_tool_ins_circle ; Predator style

Function code: 0
Group code: 0
RVP codes

r_Tool_Name

Description: Maps the current tool name
Tech Tips: Selects the current tool name. An example of using this register keyword includes the following:

"NAME:",string : r_tool_name ; Seimens style

Function code: 0
Group code: 0
RVP codes

r_Tool_Number

Description: Maps the current tool number
Tech Tips: Selects the current tool number. With the Predator style tool comments this is an T. An example of using this register keyword includes the following:

"T",real : r_tool_number ; Predator style

Function code: 0
Group code: 0
RVP codes

r_Tool_NTG_Filename

Description: Maps the current custom tool or NTG filename
Tech Tips: Selects the current custom tool or NTG style filename. An example of using this register keyword includes the following:

"NTG=",string_reg : r_tool_ntg_filename

Function code: 0
Group code: 0
RVP codes

r_Tool_Orient

Description: Maps the current tool orientation
Tech Tips: Selects the current tool orientation. With the Predator style tool comments this is an E. An example of using this register keyword includes the following:

"E",real : r_tool_orient ; Predator style

Function code: 0
Group code: 0
RVP codes

r_Tool_Orient_X

internal function code: 0

internal functionality group code:0
description

RVP codes

r_Tool_Orient_Y

internal function code: 0
internal functionality group code:0
description

RVP codes

r_Tool_Orient_Z

internal function code: 0
internal functionality group code:0
description

RVP codes

r_Tool_Ref

Description: Maps the current tool reference name
Tech Tips: Selects the current tool reference or tool library name. An example of using this register keyword includes the following:

“REF:”,string : r_tool_ref ; Predator style

Function code: 0
Group code: 0
RVP codes

r_Tool_Shank_D

Description: Maps the current tool shank diameter
Tech Tips: Selects the current tool shank diameter. With the Predator style tool comments this is an SD. An example of using this register keyword includes the following:

“SD”,real : r_tool_shank_d ; Predator style

Function code: 0
Group code: 0
RVP codes

r_Tool_Shank_H

Description: Maps the current tool shank height
Tech Tips: Selects the current tool shank height. With the Predator style tool comments this is an SH. An example of using this register keyword includes the following:

“SH”,real : r_tool_shank_h ; Predator style

Function code: 0
Group code: 0
RVP codes

r_Tool_TLB_Filename

Description: Maps the current tool library or TLB filename
Tech Tips: Selects the current toolkit, tool library or TLB filename. An example of using this register keyword includes the following:

```
%s,string_reg          : r_tool_tlb_filename          ; Predator style
```

Function code: 0
Group code: 0
RVP codes

r_Tool_Turret_Number

Description: Maps the current tool turret number
Tech Tips: Selects the current tool turret number. With the Predator style tool comments this is an N. An example of using this register keyword includes the following:

```
"N",real               : r_tool_turret_number         ; Predator style
```

Function code: 0
Group code: 0
RVP codes

r_Tool_Turret_Pos

Description: Maps the current tool turret position
Tech Tips: Selects the current tool turret position. With the Predator style tool comments this is an P. An example of using this register keyword includes the following:

```
"P",real               : r_tool_turret_pos            ; Predator style
```

Function code: 0
Group code: 0
RVP codes

r_Tool_Type

Description: Maps the current tool type
Tech Tips: Selects the current tool type or shape. With the Predator style tool comments this is an S. An example of using this register keyword includes the following:

```
"S",int                : r_tool_type                  ; Predator style
```

NOTE: For milling tool types S1 = Flat style, S2 = Ball style, S3 = Bull nose style, S4 = Drill style, S5 = Radius style, S6 = Sphere style, S7 = Chamfer style, S8 = Dovetail, S9 = Taper-ball, and S10 = Taperbull nose style.

NOTE: For turning tool types S1 = Diamond style, S2 = Triangle style, S3 = Square style, S4 = Pentagon style, S5 = Hexagon style, S6 = Radius Groove style, S7 = Groove style, S8 = Button style, S9 = Thread style, and S10 = Drill style.

Function code: 0
Group code: 0
RVP codes

r_Tool_Width

Description: Maps the current tool width
Tech Tips: Selects the current tool width. With the Predator style tool comments this is an W. An example of using this register keyword includes the following:

```
"W",real        : r_tool_width        ; Predator style
```

Function code: 0
Group code: 0
RVP codes

r_Tool_X_Prog_Point

Description: Maps the current tool insert programming point in X
Tech Tips: Selects the current tool insert programming point in the X or vertical axis. With the Predator style tool comments this is an X. An example of using this register keyword includes the following:

```
"X",real        : r_tool_x_prog_point        ; Predator style - Turning
```

NOTE: For turning type tools both the only the r_Tool_Z_Prog_Point and the r_Tool_X_Prog_Point registers are needed.

Function code: 0
Group code: 0
RVP codes

r_Tool_Z_Prog_Point

Description: Maps the current tool programming point in Z
Tech Tips: Selects the current tool programming point or offset in the Z or horizontal axis. With the Predator style tool comments this is an O for milling and a Z for turning. An example of using this register keyword includes the following:

```
"O",real        : r_tool_z_prog_point        ; Predator style - Milling
"Z",real        : r_tool_z_prog_point        ; Predator style - Turning
```

NOTE: For milling type tools only the r_Tool_Z_Prog_Point register is needed.

Function code: 0
Group code: 0
RVP codes

r_Turn_Cycle_Depth

Description: Maps the depth of cut for turn roughing and semi finishing cycles
Tech Tips: Selects the depth of cut for turn roughing and semi finishing cycles. On most CNC controls this is an D. An example of using this register keyword includes the following:

```
"D",real        : r_turn_cycle_depth        ; Fanuc style
```

Function code: 0
Group code: 0
RVP codes

r_Turn_Cycle_Excess

Description: Maps the left over material for turning cycles
Tech Tips: Selects the left over material turning cycles. An example of using this register keyword includes the following:

“D”,real : r_turn_cycle_depth ; Fanuc style

Function code: 0
Group code: 0
RVP codes

r_Turn_Cycle_End_Label

Description: Maps the end of a profile for turn roughing and semi finishing cycles
Tech Tips: Selects the end of a profile for turn roughing and semi finishing cycles. An example of using this register keyword includes the following:

“END=”,string_reg : r_turn_cycle_end_label

Function code: 0
Group code: 0
RVP codes

r_Turn_Cycle_First_Block

Description: Maps the first block for turn roughing and semi finishing cycles
Tech Tips: Selects the first block or the beginning of the profile for turn roughing and semi finishing cycles. On most CNC controls this is a P. An example of using this register keyword includes the following:

“P”,real : r_turn_cycle_first_block ; Fanuc style

Function code: 0
Group code: 0
RVP codes

r_Turn_Cycle_Last_Block

Description: Maps the last block for turn roughing and semi finishing cycles
Tech Tips: Selects the last block or the end of the profile for turn roughing and semi finishing cycles. On most CNC controls this is a Q. An example of using this register keyword includes the following:

“Q”,real : r_turn_cycle_last_block ; Fanuc style

Function code: 0
Group code: 0
RVP codes

r_Turn_Cycle_Ret_Ang

Description: Maps the retract angle for turn roughing and semi finishing cycles
Tech Tips: Selects the retract angle for turn roughing and semi finishing cycles. On most CNC controls this is an A. An example of using this register keyword includes the following:

“A”,real : r_turn_cycle_ret_ang ; Fanuc style

Function code: 0

Group code: 0
RVP codes

r_Turn_Cycle_Retract

Description: Maps the retract value for turn roughing and semi finishing cycles
Tech Tips: Selects the retract value for turn roughing and semi finishing cycles. On most CNC controls this is a R. An example of using this register keyword includes the following:

```
"R",real        : r_turn_cycle_retract      ; Fanuc style or
"E",real        : r_turn_cycle_retract      ; Vickers 2100 style
```

Function code: 0
Group code: 0
RVP codes

r_Turn_Cycle_Start_Label

Description: Maps the beginning of a profile for turn roughing and semi finishing cycles
Tech Tips: Selects the beginning of a profile for turn roughing and semi finishing cycles. An example of using this register keyword includes the following:

```
"START=",string_reg     : r_turn_cycle_start_label
```

Function code: 0
Group code: 0
RVP codes

r_Turn_Cycle_X_Allow

Description: Maps the allowance in X to leave for turn roughing and semi finishing
Tech Tips: Selects the allowance in X or vertical axis to leave for turn roughing and semi finishing cycles. On most CNC controls this is a U. An example of using this register keyword includes the following:

```
"U",real        : r_tool_cycle_x_allow      ; Fanuc style or
"I",real        : r_tool_cycle_x_allow      ; Cincinnati style
```

Function code: 0
Group code: 0
RVP codes

r_Turn_Cycle_Z_Allow

Description: Maps the allowance in Z to leave for turn roughing and semi finishing
Tech Tips: Selects the allowance in Z or horizontal axis to leave for turn roughing and semi finishing cycles. On most CNC controls this is a W. An example of using this register keyword includes the following:

```
"W",real        : r_tool_cycle_z_allow      ; Fanuc style or
"K",real        : r_tool_cycle_z_allow      ; Cincinnati style
```

Function code: 0
Group code: 0
RVP codes

r_Units

Description: Maps units

Tech Tips: Selects units. An example of using this register keyword includes the following:

""",switch@2,casesens : r_units ; Heidenhain style

Function code: 0
Group code: 0
RVP codes

r_Var_Access_Mode

internal function code: 345
internal functionality group code:0
description variable access mode (parameter, global or local)

RVP codes

r_Variable_Num

Description: Maps a variable number
Tech Tips: Selects a variable number value. An example of using this register keyword includes the following:

"VN",real_reg : r_variable_num

Function code: 343
Group code: 0
RVP codes

r_Variable_String

Description: Maps a variable string
Tech Tips: Selects a variable string value. An example of using this register keyword includes the following:

"VS",string_reg : r_variable_string

Function code: 415
Group code: 0
RVP codes

3.2.7 FORMAT keywords

The following functions and registers are used only in the multigroup format definition.

concat

internal function code: 0
internal functionality group code:0
Scope: sub-group in **multi**
Description: Allows more than one line for a pattern definition.

RVP codes:

data_separator

internal function code: 0
internal functionality group code:0
Scope: sub-group in **multi**

Description: used to separate register patterns from function patterns in multigroup formats.

RVP codes:

code_num

internal function code: 1
internal functionality group code:0
Scope: sub-group in **multi**
Description: Stores
Issues an error message
RVP codes:

code_and_subcode

internal function code: 0
internal functionality group code:0
description
RVP codes

subcode_num

internal function code: 0
internal functionality group code:0
description
RVP codes

separator

internal function code: 200
internal functionality group code:0
Scope: sub-group in **multi**
Description: used to separate patterns in multigroup formats.
RVP codes:

label

internal function code: 200
internal functionality group code:0
Scope: sub-group in **multi**
Description: used to define pattern delimiters in multigroup formats.
RVP codes:

3.2.8 FUNCTION PARAMETERS Keywords:

Some special functions allow the use of parameters. Refer to its definition for more information.

peck

function(s): **_DrillCycle**
internal format code: 2
description: enables pecking

bottomstop

function(s): **_DrillCycle**
internal format code: 4
description manual stop on bottom

bottomCW

function(s): **_DrillCycle**
internal format code: 8

description CW rotation at bottom

oriented_bottomCW
function(s): **_DrillCycle**
internal format code: 8
description CW rotation at bottom

manualfeed
function(s): **_DrillCycle**
internal format code: 8
description manual feed

retractfeed
function(s): **_DrillCycle**
internal format code: 16
description feed retract (default: rapid)

dwell
function(s): **_DrillCycle**
internal format code: 18
description dwell

3.5 Parameters:

This section allows for a wide range of CNC warnings, rules and control limits to be specified. Many parameters are either On or Off. Others parameters require a number or require specific keywords. A portion of a Fanuc 18.RPM's parameters section follows:

```
[parameters]
Check_Spindle_Speed=On          ; On requires M3 or M4 before tool motion
Check_Helical_Move=Off          ; Off allows helical interpolation
SubProg_Pos=Bottom              ; Bottom requires all subs to be below the main program
SubProg_Nest=4                  ; Sets the maximum number of nested subs
SubProg_Max=16                  ; Sets the maximum number of subs
```

Each parameter is followed by an equal sign and an appropriate value. The first two are Check parameters and require an On or Off. The third parameter, SubProg_Pos, accepts None, Top, Bottom or Both. The last two SubProg parameters require a number. Details on each parameter follow:

AB_AXIS_SENSE_MODAL
Description: Toggles the modality of A and B axis rotary motion
Tech Tips: The ON option enables modal A and B axis rotary direction. The OFF option disables modal A and B axis rotary direction. Examples using this parameter include the following:

```
AB_AXIS_SENSE_MODAL=ON      ; Enables modal rotary motion or
AB_AXIS_SENSE_MODAL=OFF     ; Disables modal rotary motion
```

Default value: ON
Parameter: string

A_AXIS_DEFAULT_SENSE
Description: Toggles default direction for A axis rotary motion
Tech Tips: The CW option enables clockwise rotary motion for the A axis. The CCW option enables counter clockwise rotary motion. Examples using this parameter include the following:

A_AXIS_DEFAULT_SENSE=CW ; Clockwise
A_AXIS_DEFAULT_SENSE=CCW ; Counter clockwise

Default value: CW
Parameter: string

ADVANCED_LINE_NUMBERS

Description: Toggles advanced line number support
Tech Tips: The ON option enables support for duplicate line numbers within multiple CNC programs, while allowing sub program calls to those appropriate line numbers. The OFF option does not support duplicate line numbers across multiple CNC programs.

NOTE: The OFF option is faster and uses less memory.

ADVANCED_LINE_NUMBERS=ON ; Supports duplicates or
ADVANCED_LINE_NUMBERS=OFF ; Simpler and faster

Default value: OFF
Parameter: string

ARC_CHECK

Description: Toggles depth and radius checking for circular interpolation
Tech Tips: The ON option enables checking each arc radius and depth to determine spiral and helical motion. The OFF option disables checking each arc radius and depth. Examples using this parameter include the following:

ARC_CHECK=ON ; Standard CNCs or
ARC_CHECK=OFF ; Older CNCs

Default value: ON
Parameter: string

ARC_MODALITY

Description: Toggles modality for circular interpolation
Tech Tips: The MODAL option allows circular interpolation, typically G2 and G3 to be modal. Most CNC controls are designed in this fashion. The NON_MODAL option requires circular interpolation codes to be present on each block. Examples using this parameter include the following:

ARC_MODALITY=MODAL ; Standard CNCs or
ARC_MODALITY=NON_MODAL ; Older CNCs

Default value: MODAL
Parameter: string

ARC_TOL_INCH

Description: Sets the tolerance for circular interpolation in inches
Tech Tips: Lower this tolerance for high precision applications and raise the tolerance for less precise work. The lower the tolerance the slower circular interpolation moves will process and the higher the tolerance the processing speed will increase. For most applications the default value of 0.005 represents a perfect balance. This tolerance affects the accuracy,

speed, and display of the simulation. It is also used to for helical interpolation. Examples using this parameter include the following:

```
ARC_TOL_INCH=0.005      ; Standard applications or
ARC_TOL_INCH=0.001      ; Higher precision applications
```

Default value: 0.005
Parameter: float

ARC_TOL_MM

Description: Sets the tolerance for circular interpolation in millimeters
Tech Tips: Lower this tolerance for high precision applications and raise the tolerance for less precise work. The lower the tolerance the slower circular interpolation moves will process and the higher the tolerance the processing speed will increase. For most applications the default value of 0.05 represents a perfect balance. This tolerance affects the accuracy, speed, and display of the simulation. It is also used for helical interpolation. Examples using this parameter include the following:

```
ARC_TOL_MM=0.05      ; Standard applications or
ARC_TOL_MM=0.01      ; Higher precision applications
```

Default value: 0.05
Parameter: float

ARCCENTER_ABSOLUTE

Description: Toggles absolute or incremental coordinates for circular interpolation
Tech Tips: The ON option specifies absolute coordinates for circular motion. The OFF option specifies incremental coordinates for circular motion.

```
ARCCENTER_ABSOLUTE=ON      ; Absolute I, J and K values or
ARCCENTER_ABSOLUTE=OFF     ; Incremental I, J and K values
```

Default value: OFF
Parameter: string

AUTO_SPINDLE_COMP

Description: Toggles automatic spindle compensation for 4 and 5 axis machines
Tech Tips: The ON option enables automatic spindle compensation. The OFF option disables automatic spindle compensation.

```
AUTO_SPINDLE_COMP=ON      ; Enables spindle compensation or
AUTO_SPINDLE_COMP=OFF     ; Disables spindle compensation
```

Default value: OFF
Parameter: string

AXIS_MODALITY

Description: Toggles modality for axis orientation
Tech Tips: The MODAL option enables modal axis orientation. This option is designed to be used when processing ATPCL files and not G code. The NON_MODAL option requires the axis orientation to be specified within each APT record. Examples using this parameter include the following:

```
AXIS_MODALITY=MODAL                ; Standard APTCL or
```

AXIS_MODALITY=NON_MODAL ; Non-standard APTCL

Default value: MODAL
Parameter: string

BOLT_PATTERN_MODE

Description: Sets bolt patterns to radius or diameter
Tech Tips

BOLT_PATTERN_MODE = RADIUS
BOLT_PATTERN_MODE = DIAMETER

Default value: RADIUS
Parameter: string

BOLT_PATTERN_STYLE

Description: Sets bolt hole pattern style
Tech Tips

BOLT_PATTERN_STYLE=MODELESS
BOLT_PATTERN_STYLE=MODAL

Default value: MODAL
Parameter: string

B_AXIS_DEFAULT_SENSE

Description: Toggles default direction for B axis rotary motion
Tech Tips: The CW option enables clockwise rotary motion for the B axis. The CCW option enables counter clockwise rotary motion. Examples using this parameter include the following:

B_AXIS_DEFAULT_SENSE=CW ; Clockwise
B_AXIS_DEFAULT_SENSE=CCW ; Counter clockwise

Default value: CW
Parameter: string

CANNED_CYCLES_MODE

Description: Toggles canned cycle modality
Tech Tips: The MODAL option specifies that canned cycles are modal until they are canceled. The NON_MODAL option requires that the canned cycle is not modal and must be repeated within each block. Examples using this parameter include the following:

CANNED_CYCLES_MODE=MODAL
CANNED_CYCLES_MODE=NON_MODAL

Default value: MODAL
Parameter: string

CHECK_COOLANT

Description: Toggles checking of machine coolant while machining
Tech Tips: The ON option will trigger an error when the coolant is not enabled and if any material is removed. The OFF option will allow the use of coolant independently of the material removal process.

CHECK_COOLANT=ON ; Requires coolant prior to any machining
CHECK_COOLANT=OFF ; Disables any coolant requirements

Default value: OFF
Parameter: string

CHECK_DIAM_OFFSET

Description: Toggles checking of diameter offset use
Tech Tips: The ON option will trigger an error when diameter offsets are not used or defined. The OFF option will allow the use of diameter offsets as needed.

CHECK_DIAM_OFFSET=ON ; Requires using diameter offsets
CHECK_DIAM_OFFSET=OFF ; Disables any diameter offset requirements

Default value: OFF
Parameter: string

CHECK_HELICAL_MOVE

Description: Toggles checking of start and end points for helical motion
Tech Tips: The ON option will trigger an error when the start and end points exceed the ARC_TOL_INCH or ARC_TOL_MM parameters. The OFF option will allow circular motion that is actually helical motion.

CHECK_HELICAL_MOVE=ON
CHECK_HELICAL_MOVE=OFF

Default value: OFF
Parameter: string

CHECK_LENGTH_OFFSET

Description: Toggles checking of length offset use
Tech Tips: The ON option will trigger an error when length offsets are not used or defined. The OFF option will allow the use of length offsets as needed.

CHECK_LENGTH_OFFSET=ON ; Requires using length offsets
CHECK_LENGTH_OFFSET=OFF ; Disables any length offset requirements

Default value: OFF
Parameter: string

CHECK_SPINDLE

Description: Toggles checking spindle rotation before tool motion
Tech Tips: The ON option requires spindle rotation prior to any tool motion. The OFF option allows tool motion to occur prior to enabling any spindle rotation.

CHECK_SPINDLE=ON ; Standard for milling applications or
CHECK_SPINDLE=OFF ; Standard for laser, flame and waterjet applications

Default value: OFF
Parameter: string

CHECK_SPINDLE_SPEED

Description: Toggles checking spindle speed before tool motion

Tech Tips: The ON option requires spindle speed greater then or less then zero prior to any tool motion. The OFF option allows tool motion to occur prior to enabling any spindle speed.

CHECK_SPINDLE_SPEED=ON ; Milling applications or
CHECK_SPINDLE_SPEED=OFF ; Laser, flame and waterjet applications

Parameter Format: string
Default value: OFF

CHECK_SPIRAL_MOVE

Description: Toggles checking start and end points for spiral motion
Tech Tips: The ON option will trigger an error when the start and end points exceed the ARC_TOL_INCH or ARC_TOL_MM parameters. The OFF option will allow circular motion that is actually spiral motion.

CHECK_SPIRAL_MOVE=ON
CHECK_SPIRAL_MOVE=OFF

Default value: OFF
Parameter: string

CHECK_TOOL_LOAD

Description: Toggles checking for loading a tool prior to performing a tool change
Tech Tips: The ON option enables checking for loading a tool prior to performing a tool change. The OFF option will allow the use of loading a tool independently of the tool change process.

CHECK_TOOL_LOAD=ON ; Requires loading a tool prior to a tool change
CHECK_TOOL_LOAD=OFF ; Disables any tool load requirements

Default value: OFF
Parameter: string

CHECK_TRAVELS

Description: Toggles checking for tool motion that exceeds the machine's travel limits
Tech Tips: The ON option enables checking for tool motion that exceeds the machine's travel limits. The OFF option ignores the machine's travel limits and does not trigger an error if they are exceeded.

CHECK_TRAVELS=ON ; Enables travel limit checking
CHECK_TRAVELS=OFF ; Disables any travel limit checking

Default value: OFF
Parameter: string

CHECK_WORK_OFFSET

Description: Toggles checking of work offset use
Tech Tips: The ON option will trigger an error when work offsets are not used or defined. The OFF option will allow the use of work offsets as needed.

CHECK_WORK_OFFSET=ON ; Requires using work offsets
CHECK_WORK_OFFSET=OFF ; Disables any work offset requirements

Default value: OFF

Parameter: string

CIRCULAR_PARAMS

Description: Toggles use of arc center in circular moves
Tech Tips: The CENTER option enables the use of center point for arc moves. The END option enables the use of end point for arc moves. Examples using this parameter include the following:

CIRCULAR_PARAMS=CENTER
CIRCULAR_PARAMS=END

Default value: END
Parameter: string

CONTOUR_SUPPORT

Description: Toggles support for Heidenhain style contours
Tech Tips: The ON option enabled support for Heidenhain style contours. The OFF option should be used for all non-Heidenhain CNC controls or when faster processing of Heidenhain files is desired. Examples using this parameter include the following.

CONTOUR_SUPPORT=ON ; Heidenhain style or
CONTOUR_SUPPORT=OFF ; Non-Heidenhain styles

Default value: OFF
Parameter: string

CONTROL_TYPE

Description: Sets the type of CNC control
Tech Tips: The MILL option requires mill only parameters and function keywords and will trigger an error if any lathe parameters or function keywords are used. The LATHE option requires lathe only parameters and function keywords and will trigger an error if any mill parameters or function keywords are used. The MILL-TURN option will allow all parameters and function keywords.

CONTROL_TYPE=MILL ; Mill only or
CONTROL_TYPE=LATHE ; Lathe only or
CONTROL_TYPE=MILL-TURN ; Mill/Turn

Default value: MILL
Parameter: string

NOTE: Currently the MILL-TURN option is reserved for future use and has not yet been implemented.

CUTTERCOMP_CONNECT

Description: Specifies two different methods of connecting external corners
Tech Tips: When this parameter is set to STANDARD the tool motion for connecting external corners will be by extending the linear motion. The ROUND option connects the external corners with a circular tool motion.

CUTTERCOMP_CONNECT=STANDARD ; Fanuc style compensation
CUTTERCOMP_CONNECT=ROUND ; Yasnac style compensation

Default value: STANDARD

Parameter: string

CUTTERCOMP_ENTRY_MODE

Description: Specifies which entry mode will be used in cutter compensation
Tech Tips:

CUTTERCOMP_ENTRY_MODE=CLASSIC
CUTTERCOMP_ENTRY_MODE=TANGENTIAL
CUTTERCOMP_ENTRY_MODE=PERPENDICULAR

Default value: CLASSIC
Parameter: string

CUTTERCOMP_ERR_LEVEL

Description: Toggles diameter compensation checking after compensating the tool motion
Tech Tips: Predator Virtual CNC will verify the length of the compensated move is greater then the size of the tool. Shoud it fail this check Predator Virtual CNC can issue either a warning or an error. The HIGH option will generate errors if the tool is too large for the compensated motion. The LOW option will generate a warning if the tool is too large for the compensated motion.

CUTTERCOMP_ERR_LEVEL=HIGH
CUTTERCOMP_ERR_LEVEL=LOW

Default value: LOW
Parameter: string

NOTE: This verification check is similar to the WARN_CUTCOMP_DIAMTOOLARGE parameter. If both of these parameters are enabled then Predator Virtual CNC will check before and after the tool compensation.

CUTTERCOMP_MODE

Description: Specifies three different methods of applying cutter compensation
Tech Tips: When this parameter is set to RADIUS each tool's radius value will automatically be applied as the cutter compensation value to the appropriate tool motion. The RADIUS option eliminates the requirement of specifying the compensation value for each tool within the Job. The DIAMETER method is identical to the RADIUS with the tool's diameter value applied as the cutter compensation value. The REGISTER method is the default method of specifying each tool's the cutter compensation values within the Job.

CUTTERCOMP_MODE=REGISTER ; Tool wear style compensation
CUTTERCOMP_MODE=RADIUS ; Part edge style compensation
CUTTERCOMP_MODE=DIAMETER

Default value: REGISTER
Parameter: string

CUTTERCOMP_STYLE

Description: Specifies two different methods of cutter compensation
Tech Tips: When this parameter is set to MODAL then cutter compensation will stay in effect until canceled. The MODELESS option requires that cutter compensation be stated on every motion line.

CUTTERCOMP_STYLE=MODAL ; Fanuc style compensation
CUTTERCOMP_STYLE=MODELESS ; Heidenhain style compensation

Default value: MODAL
Parameter: string

CUTTERCOMP_TOLERANCE

Description: Specifies the tolerance be used when calculating cutter compensation
Tech Tips: None

CUTTERCOMP_TOLERANCE=0.005 Default or
CUTTERCOMP_TOLERANCE=0.001 Tighter tolerance

Default value: 0.005
Parameter: string

CUTTERCOMP_USE_OFFSET

Description: Toggles an option where the diameter compensation value is added to the tool diameter or tool radius for a total diameter compensation
Tech Tips: When this parameter is enabled the total diameter compensation value is based on the value specified within the JOB + the value specified with the CUTTERCOMP_MODE. For example, if CUTTERCOMP_MODE is set to RADIUS and the CUTTERCOMP_USE_OFFSET is ON then the total diameter compensation value is the tool's radius plus the D value. For example, the tool's radius is 0.5 and the D value is 0.1 then the total amount of compensation, 0.6, would be applied to the tool motion.

CUTTERCOMP_USE_OFFSET=ON
CUTTERCOMP_USE_OFFSET=OFF ; Default

Default value: OFF
Parameter: string

CYCLE_DEPTH_ALWAYS_NEGATIVE

Description: Forces cycle depth to be always negative
Tech Tips: When set to ON, It will convert a depth positive value into a negative one

CYCLE_DEPTH_ALWAYS_NEGATIVE=ON
CYCLE_DEPTH_ALWAYS_NEGATIVE=OFF

Default value: OFF
Parameter: string

CYCLE_FEED_RATE

Description: Toggles support to honor modality when feed rates are defined within a cycle after the cycle completes
Tech Tips: The ON option means any feed rate specified within a canned cycle will continue to be in effect after the cycle completes. The OFF option means that any feed rate specified within the canned cycle only stays ineffect within the canned cycle. After the canned cycle completes the previous feed rate is re-established. Examples using this parameter include the following:

CYCLE_FEED_RATE=ON ; Standard CNCs

CYCLE_FEED_RATE=OFF ; Non standard CNCs

Default value: ON
Parameter: string

CYCLE_HIRETRACT

Description: Option enabling additional clearance when retracting within a canned cycle
Tech Tips: None. Examples using this parameter include the following:

CYCLE_HIRETRACT=0.01 ; Adds 0.01 to the retract plane or
CYCLE_HIRETRACT=0 ; Default

Default value: 0.0
Parameter: string

CYCLE_INC_DEPTH_FROM_R_PLANE

Description: Toggles the depth of the hole from the retract plane
Tech Tips: The ON option enables calculating the depth of the hole as an incremental amount from the rapid plane. The OFF option calculates the depth of the normally. Examples using this parameter include the following:

CYCLE_INC_DEPTH_FROM_R_PLANE=ON ; Non-Standard CNCs or
CYCLE_INC_DEPTH_FROM_R_PLANE=OFF ; Standard CNCs

Default value: OFF
Parameter: string

CYCLE_PARAM_STYLE

Description: Specifies which CNC vendor style canned cycle behavior will be used
Tech Tips: Cycles vary greatly with each builder and it is important that the correct vendor style be specified when advanced cycles are used. This parameter is used by Predator Virtual CNC to address CNC logic and tool motion that would often be to difficult to describe generically.

CYCLE_PARAM_STYLE=CINCINNATI
CYCLE_PARAM_STYLE=FAGOR
CYCLE_PARAM_STYLE=FADAL
CYCLE_PARAM_STYLE=FANUC
CYCLE_PARAM_STYLE=HANKOOK
CYCLE_PARAM_STYLE=HAAS
CYCLE_PARAM_STYLE=HEIDENHAIN
CYCLE_PARAM_STYLE=MAHO
CYCLE_PARAM_STYLE=NEW_FANUC
CYCLE_PARAM_STYLE=OKUMA
CYCLE_PARAM_STYLE=SIEMENS
CYCLE_PARAM_STYLE=UNIGRAPHICS

Default value: FANUC
Parameter: string

NOTE: The FANUC option provides compatibility with the original two line Fanuc canned cycles. The NEW_FANUC option provides compatibility with the newer one line Fanuc canned cycles.

CYCLE_RETURN

Description: Toggles the depth to return from a cycle
Tech Tips: The INIT option enables returning to the initial point. The REFP option enables returning to the reference plane. Examples using this parameter include the following:

```
CYCLE_RETURN=INIT          ; Initial depth or
CYCLE_RETURN=REFP          ; Reference Plane
```

Default value: INIT
Parameter: string

CYCLE_RETURN_FEED

Description: Specifies the modality when returning from a canned cycle
Tech Tips: The MODEL option restores the feed rate prior to the canned cycle. The FEED option honors the last feed rate specified within the canned cycle to be used for subsequent tool motion. The RAPID option automatically reverts the modality of the CNC control to rapid when returning from a canned cycle. Examples using this parameter include the following:

```
CYCLE_RETURN_FEED=MODAL    ; Returns to the previous feed
CYCLE_RETURN_FEED=FEED     ; Returns using the current feed rate
CYCLE_RETURN_FEED=RAPID    ; Returns in a Rapid state
```

Default value: MODAL
Parameter: string

CYCLE_RETRACT

Description: Sets the amount to retract in a cycle
Tech Tips: This value will override any programmed retract amount within the CNC program. Examples using this parameter include the following:

```
CYCLE_RETRACT=0.01         ; Standard retract depth or
CYCLE_RETRACT=0.05
CYCLE_RETRACT=PROCESS
CYCLE_RETRACT=IGNORE
```

Default value: 0.01
Parameter: float

DECIMAL_POINT

Description: Enables the conversion of numbers without decimal points
Tech Tips: If the number is not zero it will divide all real numbers by the specified value to convert the numbers without decimal points. A value of 0 will not convert values and assumes that decimal points are used within the CNC program. Examples using this parameter include the following:

```
DECIMAL_POINT=0            ; Assume decimal points or
DECIMAL_POINT=10000        ; Convert values without decimal points
```

Default value: 0
Parameter: number

DECIMAL_SEPARATOR

Description: Specify a decimal seperator

Tech Tips: Specifies the appropriate character to separate the integer portion of a number from its fractional portion. This is usually a period or a comma.

DECIMAL_SEPARATOR =”.”
DECIMAL_SEPARATOR =”,”

Default value: “.”
Parameter: string

NOTE: If multiple decimal sepatators need to be processed at the same time enable the SECONDARY_DECIMAL_SEPERATOR with the appropriate value. Typically, the DECIMAL_SEPERATOR is set to a period and the SECONDARY_DECIMAL_SEPERATOR is set to a comma.

DRILL_CYCLE_RAPID_AMOUNT

Description: Percent used to compute the rappid length of the rapid move before the next peck.
Tech Tips: This is used only with drill cycles that peck.

DRILL_CYCLE_RAPID_AMOUNT=0.05 ; Rapid to 0.050 above previous peck
DRILL_CYCLE_RAPID_AMOUNT=0.01 ; Rapid to 0.010 above previous peck
DRILL_CYCLE_RAPID_AMOUNT=0 ; 100% Feed

Default value: 0
Parameter: string

DWELL_UNITS

Description: Specifies which units will be used for Dwell.
Tech Tips: This is specially useful for cycle time computations

DWELL_UNITS=SECONDS
DWELL_UNITS=MILISECONDS

Default value: SECONDS
Parameter: string

ENABLE_WARNINGS

Description: Toggles the display of warning messages
Tech Tips: The ON option will stop processing when the first warning message is triggered. The OFF option will not stop and ignores any warning messages. Examples using this parameter include the following:

ENABLE_WARNINGS=ON ; Display warning messages or
ENABLE_WARNINGS=OFF ; Ignore warning messages

Default value: OFF
Parameter: string

ERROR_MODE

Description: Toggles the display of detailed error messages
Tech Tips: The DETAILED option will display extended and detailed error messages. The NORMAL option will display a normal level of error message detail. Examples using this parameter include the following:

ERROR_MODE=DETAILED ; Display extended detailed error messages

ERROR_MODE=NORMAL ; Display default error messages

Default value: NORMAL
Parameter: string

EXPRESSION_CLOSE

Description: Defines a string to specify the end of a nested math operation
Tech Tips: Although different strings can be defined most CNC controls default to a] character to end a nested math operation. Examples using this parameter include the following:

EXPRESSION_CLOSE=')‘ ; Fadal style
EXPRESSION_CLOSE=']‘ ; Fanuc style

Default value:]
Parameter: string

EXPRESSION_OPEN

Description: Defines a string to specify the beginning of a nested math operation
Tech Tips: Although different strings can be defined most CNC controls default to a [character to begin a nested math operation. Examples using this parameter include the following:

EXPRESSION_OPEN='(‘ ; Fadal style
EXPRESSION_OPEN='[‘ ; Fanuc style

Default value: [
Parameter: string

FIRST_CODES

Description: One or more G & M codes to assume enabled upon startup
Tech Tips: List one or more G & M codes, without spaces, to match power up or startup conditions of the CNC control. Examples using this parameter include the following:

FIRST_CODES= ; No startup G & M codes or
FIRST_CODES=G4080 ; Cancel cutter comp and canned cycles on startup

Default value: <empty>
Parameter: string

FEED_RATE_UNITS

Description: Sets feed rate units
Tech Tips None

FEED_RATE _UNITS=METERS
FEED_RATE _UNITS=MILIMETERS

Default value: MILIMETERS
Parameter: string

FOURTH_AXIS_AS_CIRCULAR

Description: Toggles support for 3D arcs on 4 and 5 axis Heidenhains
Tech Tips The ON option enables 3D arcs on 4 and 5 axis Heidenhain style machines. This parameter works with the MoveCircular command for the

arc definition. The OFF option disables support for 3D arcs. Refer to the following examples:

FOURTH_AXIS_AS_CIRCULAR=ON ; Enables Heidenhain style 3D arcs
FOURTH_AXIS_AS_CIRCULAR=OFF

Default value: OFF
Parameter: string

IGNORE_UNTIL

Description: Option to ignore processing until after a specific string
Tech Tips This option allows header information to be ignored by Predator Virtual CNC. Refer to the following examples:

IGNORE_UNTIL=% ; Ignore until after the first %
IGNORE_UNTIL=<HEADER> ; Ignore until after the <HEADER> string

Default value: Null
Parameter: string

IJK_MODALITY

Description: Enables or disables IJK modality
Tech Tips: Option MODAL will treat I,J and K registers as modal. Option NON_MODAL will treat them as non modal

IJK_MODALITY=MODAL
IJK_MODALITY=NON_MODAL

Default value: MODAL
Parameter: string

INVERSE_TIME_FEED

Description: Toggles support for inverse time feed rates
Tech Tips: The ON option will enable support for inverse time feed rates. The OFF option will disable support for inverse time feed rates. Examples using this parameter include the following:

INVERSE_TIME_FEED=ON ; Enable Inverse time feed rates
INVERSE_TIME_FEED=OFF ; Disable Inverse time feed rates

Default value: OFF
Parameter: string

JUMP_MODE

Description: Specifies the mode to be used for goto instructions
Tech Tips: When set to NUMBER, it will go jump to the specified line. When set to LABEL, it will jump to the corresponding label

JUMP_MODE=LABEL
JUMP_MODE=NUMBER

Default value: NUMBER
Parameter: string

LABEL_REP

Description: Toggle support for Heidenhain style Labels within the Main program
Tech Tips: None

```
LABEL_REP=ON                ; Enable for Heidenhains
LATHE_REP=OFF
```

Default value: OFF
Parameter: string

LABEL_FEED_FACTOR

Description: Specifies a feed rate factor to be used for lathe motion
Tech Tips: None

```
LABEL_FEED_FACTOR=1000      ; Default in MM
LATHE_FEED_FACTOR=12        ; Default in Inch
```

Default value: 1000 for Metric and 12 for Inch
Parameter: string

LATHE_FEED_MODE

Description: Specifies whether lathe feed rates will be specified per minute or per revolution
Tech Tips: None

```
LATHE_FEED_MODE=REVOLUTION
LATHE_FEED_MODE=MINUTE
```

Default value: MINUTE
Parameter: string

LATHE_GROOVE_INCX

Description: Specifies a default stepover in the X axis for lathe grooving when no stepover is provided within the cycle.
Tech Tips: This value should match the CNC's default parameter.

```
LATHE_GROOVE_INCX=.2
```

Default value: .1
Parameter: number

LATHE_GROOVE_INCZ

Description: Specifies a default stepover in the Z axis for lathe grooving when no stepover is provided within the cycle.
Tech Tips: This value should match the CNC's default parameter.

```
LATHE_GROOVE_INCZ=.2
```

Default value: .1
Parameter: number

LATHE_GROOVE_INCREMENTS

Description: Specifies a default stepovers for lathe grooving when no stepover is provided within the cycle.
Tech Tips: This value should match the CNC's default behavior.

LATHE_GROOVE_INCREMENTS=USE_DEFAULTS
LATHE_GROOVE_INCREMENTS=???

Default value: USE_DEFAULTS
Parameter: number

LATHE_INSERT_MOVES

Description: Toggle lathe insert moves
Tech Tips: The ENABLED turns on support for lathe insert moves within lathe canned cycles. The DISABLED option turns off support for lathe insert moves within lathe canned cycles. Examples using this parameter include the following:

LATHE_INSERT_MOVES=DISABLED
LATHE_INSERT_MOVES=ENABLED

Default value: DISABLED
Parameter: string

LATHE_PROG_MODE

Description: Toggle the programming method for the X axis
Tech Tips: The DIAMETER option divides all X axis values by two prior to processing. The RADIUS option uses all X axis values as specified. Examples using this parameter include the following:

LATHE_PROG_MODE=DIAMETER ; Diameter style programming or
LATHE_PROG_MODE=RADIUS ; Radius style programming

Default value: RADIUS
Parameter: string

LCYCLE_FACTOR

Description: Enables lathe canned cycle values to match the machine builder's definitions for lathe canned cycles
Tech Tips: If the number is not zero it will divide all real numbers by the specified value to convert the numbers without decimal points. A value of 0 will not convert values and assumes that decimal points are used within the CNC program. Examples using this parameter include the following:

LCYCLE_FACTOR=10000 ; Machine Builder A
LCYCLE_FACTOR=1000 ; Machine Builder B

NOTE: Although this option is maintained for backwards compatibility, we are recommending that the reverse post be setup so REAL2, REAL3, REAL4 or REAL5 is used within the lathe canned cycle pattern definition.

Default value: 1
Parameter: real

LOADTOOL_FROM_SPINDLE

Description: Allows multiple spindles to have duplicate tool numbers
Tech Tips: The LOADTOOL_FROM_SPINDLE option can be enabled to support different tool numbers within a single setup provide they are each assigned to a different spindles. For example T1 on Spindle1 can be a 1" end mill

while T1 on Spindle2 can be a 2” end mill. Examples using this parameter include the following:

```
LOADTOOL_FROM_SPINDLE=ON      ; Enable duplicate tool numbers or
LOADTOOL_FROM_SPINDLE=OFF     ; Requires unique tool numbers
```

Default value: OFF
Parameter: string

LOFFSET_AXIS

Description: Set the axis used for length offset
Tech Tips: The X option specifies the X axis. The Y option specifies the Y axis. The Z option specified the Z axis. The PROG option specifies that no axis is defined. Examples using this parameter include the following:

```
LOFFSET_AXIS=X          ; X Axis or
LOFFSET_AXIS=Y          ; Y Axis or
LOFFSET_AXIS=Z          ; Z Axis or
LOFFSET_AXIS=PROG       ; no axis specified
```

Default value: PROG
Parameter: string

LOFFSET_SIGN

Description: Toggles the length offset sign
Tech Tips: The ON option allows the length offset to be negative or positive. The OFF option ignores the sign. Examples using this parameter include the following:

```
LOFFSET_SIGN=ON
LOFFSET_SIGN=OFF
```

Default value: OFF
Parameter: string

LOOKAHEAD_BLOCKS

Description: Sets the number of blocks to look ahead for cutter compensation
Tech Tips: This is the number of non-motion blocks Predator will look ahead to find the next motion block to determine cutter compensation. This number should not match the look ahead capabilities of the CNC. Just like a CNC, Predator Virtual CNC looks ahead 1 or 2 motion blocks automatically when calculating cutter compensation. Examples using this parameter include the following:

```
LOOKAHEAD_BLOCKS=500      ; Defines the number of blocks to look ahead
LOOKAHEAD_BLOCKS=1000     ; Default number
```

Default value: 1000
Parameter: string

MACRO_ARG_LIST

Description: Sets the variables used for macro calls
Tech Tips: Assigns the variables to specific letter addresses. The letters are used to pass appropriate values with a “G66” macro call. The numbers are

variables used within the macro. Examples using this parameter include the following:

```
MACRO_ARG_TYPE=TYPE2          ; Custom style Macro Calls
MACRO_ARG_MAX=5
MACRO_ARG_LIST=A,5,B,4,C,3,D,2,E,1
```

Default value: void
Parameter: string

NOTE: This parameter is only required when the MACRO_ARG_TYPE=TYPE2 and is not required when TYPE1 style macro calls are used.

MACRO_ARG_MAX

Description: Sets the maximum number of arguments for a macro call
Tech Tips: Selects the maximum number of arguments allowed by the CNC control. Examples using this parameter include the following:

```
MACRO_ARG_MAX=14              ; Fanuc style
```

Default value: 16
Parameter: int

MACRO_ARG_TYPE

Description: Sets the type of macro call supported by the CNC
Tech Tips: The TYPE1 option enables a predefined “FANUC” standard variable assignment with macro calls. The TYPE2 option allows configuration of the assignment of letter addresses to specific variables. Examples using this parameter include the following:

```
MACRO_ARG_TYPE=TYPE1          ; Fanuc style or
MACRO_ARG_TYPE=TYPE2          ; Custom style
```

Default value: TYPE1
Parameter: string

MACRO_MAX

Description: Specifies the maximum number of embedded macros
Tech Tips: This parameter only gets used when MACRO_SUPPORT=ON. Examples using this parameter include the following:

```
MACRO_MAX=16          ; Predator Default or
MACRO_MAX=8           ; Custom number to save memory
```

Default value: 16
Parameter: string

MACRO_MAX_PARAM

Description: Specified the maximum number of parameters used with an embedded macro
Tech Tips: This parameter only gets used when MACRO_SUPPORT=ON. Examples using this parameter include the following:

```
MACRO_MAX_PARAM=32            ; Predator Default
MACRO_MAX_PARAM=4             ; Custom number to save memory
```

Default value: 32
Parameter: string

MACRO_MODAL_STYLE

Description: Toggles calling a macro modal call with the initial code
Tech Tips: The INCLUDE_DEF option will call a macro call on the same line that it is specified. The EXCLUDE_DEF will call a macro call on the following line. Typically a macro modal call is a G66 and a G67 disables this function. Examples using this parameter include the following:

```
MACRO_MODAL_STYLE=INCLUDE_DEF        ; Fadal style or
MACRO_MODAL_STYLE=EXCLUDE_DEF        ; Fanuc style
```

Default value: EXCLUDE_DEF
Parameter: string

MACRO_SUPPORT

Description: Toggles an option to embed macros within the reverse post
Tech Tips: The ON option enables embedding a macro withint the reverse post. On a Fanuc control this is just like having a 9000 series macro assigned to a particular M or G code. With this option the actual 9000 series macro can be downloaded from the CNC and pasted into the reverse post for processing by Predator Virtual CNC. The OFF option disables embedded macros and will process standard G code faster. Examples using this parameter include the following:

```
MACRO_SUPPORT=ON          ; Enable custom Fanuc 9000 series macros or
MACRO_SUPPORT=OFF         ; Disable support for faster processing
```

Default value: OFF
Parameter: string

NOTE: Macro support allows you to add additional G and M code support to Predator Virtual CNC by writing a G code style macro.

MANUAL_TOOL_LENGTH

Description: Toggles support for T offsets in addition to standard length offsets
Tech Tips: The ON option will ignore any T offsets specified within the Length Offset section of the Machine Tab. The OFF option will honor any T offsets specified within the Legnth Offset section of the Machine tab. T offsets are different than standard legnth offsets or H values. T offsets are used to allow posted G code to be output coordinates at the spindle face or any other position vs. the tip of the tool.

NOTE: This option should be set to OFF for all 5 axis simultanous machine simulation.

```
MANUAL_TOOL_LENGTH=ON     ; Ignores upper and lower case differences
MANUAL_TOOL_LENGTH=OFF    ; Required for 5 Axis simultanous machine sim.
```

Default value: ON
Parameter: string

MAX_LOOPS

Description: Sets the maximum number of loops allowed

Tech Tips: A common programming mistake is the infinite loop. By default Predator Virtual CNC expects the CNC program to progress in a linear fashion. If Predator Virtual CNC needs to allow you CNC program to execute a series of loops then define the maximum with this parameter. With a Fanuc CNC loops are commonly done with GOTOs and DO WHILEs. Examples using this parameter include the following:

```
MAX_LOOPS=0          ; Default or
MAX_LOOPS=100        ; Custom style
```

Default value: 0
Parameter: int

MCH_FILE

Description: Sets an optional second machine to be loaded and unloaded
Tech Tips: Assigns a second machine or .MCH file to be loaded and unloaded when activated with specific G codes. This parameter works with the LoadMachine and RestoreMachine keywords. Examples using this parameter include the following:

```
MCH_FILE=5axis_KGK_WRAH.MCH    ; Second machine filename
```

Default value: Null
Parameter: string

NOTE: This parameter can be used to load a second machine with an optional attachment. For example, a right angle head can be loaded via G or M codes. If this is the case the only difference between the primary and the secondary machine would be the right angle head. Another example would be to switch between a roughing head and a finishing head.

NC_IGNORE_CASE

Description: Toggles checking of upper and lower case characters
Tech Tips: The ON option will assume that upper and lower case characters are the same and can be used interchangeably. For example, G1 and g1 could both be used as needed for linear motion. The OFF option treats upper and lower case characters independently. For example, G1 would be used for linear motion while g1 would cause an error.

```
NC_IGNORE_CASE=ON      ; Ignores upper and lower case differences
NC_IGNORE_CASE=OFF     ; Upper and lower case letters are different
```

Default value: OFF
Parameter: string

NC_FORMAT

Description: Sets the file format of the CNC program
Tech Tips: The ASCII option enables the CNC program to be in ASCII. The ASCII72$ option enables the CNC program to be 72 characters wide with a $ at the end of each line. Examples using this parameter include the following:

```
NC_FORMAT=ASCII        ; Standard CNC style or
NC_FORMAT=ASCII72$     ; Older CNC style
```

Default value: ASCII
Parameter: string

NEED_SETSPINDLE

Description: Toggles option that requires a spindle definition per tool
Tech Tips: The ON option requires that a spindle be defined for each tool within the Job setup. This option is commonly enabled when the machine has multiple spindles. The OFF option is for single spindle machines or when the first spindle is defined by default for each tool. Examples using this parameter include the following:

```
NEED_SETSPINDLE=ON          ; Enabled for multiple spindle machines or
NEED_SETSPINDLE=OFF         ; Single spindle machines
```

Default value: OFF
Parameter: string

NULL_VALUES

Description: Toggles support for nulls
Tech Tips: The ON option enables support for null values. The OFF option disables support for null values. Examples using this parameter include the following:

```
NULL_VALUES=ON
NULL_VALUES=OFF
```

Default value: OFF
Parameter: string

OFFSET_INDEX_START

Description: Sets the first index used for Set work coordinates
Tech Tips: If set to 0, G10P1 will load the offset specified for G55. If set to 1, G10P1 will load the offset specified for G54

```
OFFSET_INDEX_START=0
OFFSET_INDEX_START=1
```

Default value: 0
Parameter: int

PARAMSBEFORE_CANNEDCYCLE

Description: Specifies whether canned cycle parameters will be specified before or after the cycle code
Tech Tips:

```
PARAMSBEFORE_CANNEDCYCLE=ON          ; Dynapath style
PARAMSBEFORE_CANNEDCYCLE=OFF
```

Default value: OFF
Parameter: string

POCKET_CYCLE_MODE

Description: Enables or disables the modality of pocket cycles
Tech Tips

```
POCKET_CYCLE_MODE=MODAL
POCKET_CYCLE_MODE=NON_MODAL
```

Default value: NON_MODAL
Parameter: string

POLAR_MOVE_MODE

Description: Enables or disables the modality of polar moves
Tech Tips:

POLAR_MOVE_MODE=MODAL
POLAR_MOVE_MODE=NON MODAL

Default value: NON MODAL
Parameter: string

RAPID_CANCELCYCLE

Description: Toggles support for rapid motion canceling a drill cycle
Tech Tips: The ON option enables canceling a drilling cycle when rapid motion occurs. The OFF option allows rapids to occur within a drilling cycle. Examples using this parameter include the following:

RAPID_CANCELCYCLE=ON
RAPID_CANCELCYCLE=OFF

Default value: OFF
Parameter: string

RAPID_MOVE_MODE

Description: Enables or disables rapid moves modality
Tech Tips

RAPID_MOVE_MODE=MODAL
RAPID_MOVE_MODE=NON_MODAL

Default value: MODAL
Parameter: string

REFERENCE_MOVE

Description: Toggles the final position of a reference move
Tech Tips: The TOOL_CHANGE option will return the tool to the tool change position for the specified axes during a reference move. Reference moves are typically defined as a G28. The MACHINE_HOME option will return the tool to the machine's home position during a reference move. The INTERMEDIATE option will only position the tool to the intermediate point specified within the line containing the reference move. Examples using this parameter include the following:

REFERENCE_MOVE=INTERMEDIATE
REFERENCE_MOVE=TOOL_CHANGE
REFERENCE_MOVE=MACHINE_HOME

NOTE: The most popular usage of reference moves is G28G91Z0 it is important that either the TOOL_CHANGE or MACHINE_HOME parameter, position and pattern be defined for proper simulation.

Default value: MACHINE_HOME
Parameter: string

REFERENCE_POINT

Description: Toggle the reference point style
Tech Tips: The ALWAYS_ABSOLUTE option will process a reference point's values using absolute positioning. The INC_AND_ABSOLUTE option will process a reference point's values using incremental or absolute positioning. Examples using this parameter include the following:

REFERENCE_POINT=ALWAYS_ABSOLUTE
REFERENCE_POINT=ALWAYS_INCREMENTAL
REFERENCE_POINT=INC_AND_ABSOLUTE

Default value: INC_AND_ABSOLUTE
Parameter: string

REFERENCE_POS_X

Description: Defines the default reference position in X
Tech Tips: This lathe only parameter defines the default reference position in the X axis. Examples using this parameter include the following:

REFERENCE_POS_X=20.0 ; Specific per Lathe

Default value: 1000000.0
Parameter: string

REFERENCE_POS_Z

Description: Defines the default reference position in Z
Tech Tips: This lathe only parameter defines the default reference position in the Z axis. Examples using this parameter include the following:

REFERENCE_POS_Z=20.0 ; Specific per Lathe

Default value: 1000000.0
Parameter: string

ROTATION_MOVE_MODE

Description: Enables or disables rotation moves modality
Tech Tips:

ROTATION_MOVE_MODE=MODAL
ROTATION_MOVE_MODE=NON MODAL

Default value: NON MODAL
Parameter: string

ROTARY_STYLE

Description: Enables or disables advanced 4 and 5 axis tool compensation
Tech Tips: The NORMAL option is the standard method CNC controls have been performing 4 and 5 axis tool compensation for the last 25 years. The COMPENSATED style is a new more intelligent method of performing 4 and 5 axis tool compensation that has only been available in the last few years.

ROTARY_STYLE=NORMAL ; Cincinnati style
ROTARY_STYLE=COMPENSATED ; Fidia style

NOTE: Enabling or disabling this functionality positions the tool very differently. We recommend creating a few test programs to insure Predator Virtual CNC appropriately duplicates the machine's tool positioning. For example:

N10 T1 M6
N20 G90 G0 X0 Y0 Z100 A0 B0 (RESET TOOL)
N30 Z0 (REFERENCE POINT)
N40 Z50
N50 A45 B45
N60 Z0 (RETURN TO REFERENCE POINT?)

NOTE: At N30 the tool should go to the reference point with the tool positioned straight. At N60 the tool will be positioned at an angle and will either return to the same reference point or it will be off center. If the tool returns to the same reference point enable the rotary tool compensation feature. If the tool is positioned off center then disable rotary tool compensation.

Default value: NORMAL
Parameter: string

RVP_FORMAT

Description: Sets the RVP processing format
Tech Tips: The ASCII option creates ASCII files for processing. The BINARY option creates binary files for faster processing. Examples using this parameter include the following:

RVP_FORMAT=ASCII
RVP_FORMAT=BINARY

Default value: BINARY
Parameter: string

NOTE: Currently this parameter is reserved for future use and has not yet been implemented.

SECONDARY_DECIMAL_SEPARATOR

Description: Specify a secondary decimal seperator
Tech Tips: Specifies the appropriate character to separate the integer portion of a number from its fractional portion. This is usually a period or a comma. Examples using this parameter include the following:

SECONDARY_DECIMAL_SEPARATOR =”.”
SECONDARY_DECIMAL_SEPARATOR =”,”

Default value: “.”
Parameter: string

NOTE: If multiple decimal sepatators need to be processed at the same time enable the SECONDARY_DECIMAL_SEPERATOR with the appropriate value. Typically, the DECIMAL_SEPERATOR is set to a period and the SECONDARY_DECIMAL_SEPERATOR is set to a comma. Examples using this parameter include the following:

SPINDLE_SPEED_UNITS

Description: Sets spindle speed units
Tech Tips

SPINDLE_SPEED_UNITS=METERS
SPINDLE_SPEED_UNITS=MILIMETERS

Default value: MILIMETERS
Parameter: string

STOCK_COMMENTS

Description: Toggles Predator Virtual CNC's ability to scan the CNC program for stock and fixture shapes within comments.
Tech Tips: The ON option enables scanning for stock and fixture shapes. The OFF option disables scanning and requires the stock and fixture shapes to be defined within Predator Virtual CNC. Examples using this parameter include the following:

STOCK_COMMENTS=ON ; Recommended for most applications
STOCK_COMMENTS=OFF

Default value: OFF
Parameter: string

STRING_LABELS

Description: Toggles string style sub program calls.
Tech Tips: The ON option enables alphanumeric labels to be used for sub program calls and lathe canned cycles definitions. The OFF option supports only numeric sub program numbers and lathe canned cycle definitions. Examples using this parameter include the following:

STRING_LABELS= ON ; Siemens or Okuma style
STRING_LABELS= OFF ; Fanuc style

Default value: OFF
Parameter: string

SUBPROG_AND_LINENUM

Description: Toggles storing the sequence number when calling a sub.
Tech Tips: Some Siemens CNC controls have the ability to call specific line numbers within the main or any sub program. This option has to be enabled to support this type of functionality. Examples using this parameter include the following:

SUBPROG_AND_LINENUM=ON ; Siemens style
SUBPROG_AND_LINENUM=OFF ; Default

Default value: OFF
Parameter: string

SUBPROG_CALL

Description: Sets the method for calling subroutines and sub programs
Tech Tips: The PROG_NUM option calls a sub by the program number. The PROG_LINE option calls a sub by the line or block number: The

PROG_STRING option calls a sub by a program name or string. Examples using this parameter include the following:

```
SUBPROG_CALL=PROG_NUM       ; Fanuc style or
SUBPROG_CALL=PROG_LINE      ; Fagor 800 style or
SUBPROG_CALL=PROG_STRING    ; Vickers 2100 style
```

Default value: PROG_NUM
Parameter: string

SUBPROG_DEF_CALL_CODE

Description: Toggles an option to specify whether defining subroutines and sub programs are the same as calling subroutines and sub programs

Tech Tips: The DIFF option specifies that defining and calling subroutines and sub programs are different. The EQUAL option specifies that defining and calling subroutines are the same. Examples using this parameter include the following:

```
SUBPROG_DEF_CALL_CODE=DIFF
SUBPROG_DEF_CALL_CODE=EQUAL
```

Default value: DIFF
Parameter: string

NOTE: Currently this parameter is reserved for future use and has not yet been implemented.

SUBPROG_DEFEXTENSION

Description: Sets a default extension for external sub programs

Tech Tips: Predator Virtual CNC requires that all external sub programs be located in the same directory as the main program. By setting this parameter Predator Virtual CNC will append this extension to the name of the sub program called within the main program.

```
SUBPROG_DEFEXTENSION=SUB    ; or any appropriate extension: NC, NCC,
                            ; NCF, MIN, H, T, TAP, CAM, CNC, etc.
```

Parameter: string

SUBPROG_EXTERNAL

Description: Toggles an option to specify whether subroutines and sub programs are in external files or contained within the main file

Tech Tips: The ON option processes subroutines and sub programs in separate external files. Predator Virtual CNC requires that all external sub programs be located in the same directory as the main program. The OFF option requires that the subroutines and sub programs reside within the main CNC program or file. Examples using this parameter include the following:

```
SUBPROG_EXTERNAL=ON         ; Subs are separate from the main
SUBPROG_EXTERNAL=OFF        ; Subs are included with the main
```

NOTE: Predator Virtual CNC does supported nested external sub program calls.

Default value: OFF
Parameter: string

SUBPROG_MAX

Description: Sets the maximum number of subroutines and sub programs allowed by the CNC control.

Tech Tips: Should this maximum number be exceeded, Predator Virtual CNC will trigger appropriate warning messages. Examples using this parameter include the following:

```
SUBPROG_MAX=16          ; Fanuc 0 or
SUBPROG_MAX=32          ; Vickers 2100
```

Default value: 128
Parameter: int

SUBPROG_MAX_REP

Description: Sets the maximum number of subroutines and sub programs repetitions allowed by the CNC control.

Tech Tips: Should this maximum number be exceeded, Predator Virtual CNC will trigger appropriate warning messages. Examples using this parameter include the following:

```
SUBPROG_MAX_REP=999         ; Fanuc 0 or
SUBPROG_MAX_REP=100         ; Vickers 2100
```

Default value: 16
Parameter: int

SUBPROG_NEST

Description: Sets the maximum number of nested subroutine and sub program calls allowed by the CNC control.

Tech Tips: Should this maximum number be exceeded, Predator Virtual CNC will trigger appropriate warning messages. In addition, Predator Virtual CNC has a simulation limit of up to 512 nested levels of subs. Examples using this parameter include the following:

```
SUBPROG_NEST=4          ; Most CNCs
SUBPROG_NEST=8
```

Default value: 16
Parameter: int

SUBPROG_NUMBER_DIGITS

Description: Sets the number of digits to identify number based subroutines and sub programs.

Tech Tips: Should this number be exceeded, Predator Virtual CNC will trigger appropriate warning messages. Often this parameter is used in combination with SUBPROG_NTIMES_DIGITS and SBPROG_PROGNUMBER_POS to extract the appropriate information. Examples using this parameter include the following:

```
SUBPROG_NUMBER_DIGITS=4     ; Fanuc style or
SUBPROG_NUMBER_DIGITS=2     ; Fadal style
```

Default value: 4
Parameter: int

SUBPROG_NTIMES_DIGITS

Description: Sets the number of digits for a subroutine or sub program to repeat.
Tech Tips: Should this number be exceeded, Predator Virtual CNC will trigger appropriate warning messages. Often this parameter is used in combination with SUBPROG_NUMBER_DIGITS and SBPROG_PROGNUMBER_POS to extract the appropriate information. Examples using this parameter include the following:

```
SUBPROG_NTIMES_DIGITS=4        ; Fanuc style or
SUBPROG_NTIMES_DIGITS=2        ; Fadal style
```

Default value: 0
Parameter: int

SUBPROG_POS

Description: Sets a CNC programming rule or CNC control requirement to where subroutines and sub programs are located in relationship to the main CNC program.
Tech Tips: The NONE option specifies that Predator Virtual CNC should not search through the main CNC program looking for subroutines and sub programs. The TOP option requires all subroutines and sub programs to be located at the beginning of the file before the main program. The BOTTOM option requires all subroutines and sub programs to be located at the end of the file after the main program. The BOTH option allows the subroutines and sub programs to be located at the start or the end of the file.

```
SUBPROG_POS=NONE       ; No sub program support in the CNC
SUBPROG_POS=TOP        ; Sub Programs can only be above the main
SUBPROG_POS=BOTTOM     ; Sub Programs can only be below the main
SUBPROG_POS=BOTH       ; Sub Programs can be at the top or the bottom
```

Default value: NONE
Parameter: string

SUBPROG_PROGNUMBER_POS

Description: Sets the position of the subroutine or sub program number within a number that also contains the number of repetitions.
Tech Tips: The FIRST option specifies that the portion of the subroutine or sub program number that identifies the sub is at the beginning. The LAST option specifies that the number is at the end.

```
SUBPROG_PROGNUMBER_POS=FIRST       ; Fadal style or
SUBPROG_PROGNUMBER_POS=LAST        ; Fanuc style
```

Default value: FIRST
Parameter: string

SUBPROGEX_CALL

Description: Sets the method for calling extended subroutines and sub programs
Tech Tips: The PROG_NUM option calls a sub by the program number. The PROG_LINE option calls a sub by the line or block number: The PROG_STRING option calls a sub by a program name or string. Examples using this parameter include the following:

```
SUBPROGEX_CALL=PROG_NUM            ; Fanuc style or
```

SUBPROGEX_CALL=PROG_LINE ; Fagor 800 style or
SUBPROGEX_CALL=PROG_STRING ; Vickers 2100 style

Default value: PROG_NUM
Parameter: string

SUBPROGEX_POS

Description: Sets a CNC programming rule or CNC control requirement to where extended subroutines and sub programs are located in relationship to the main CNC program.

Tech Tips: The NONE option specifies that Predator Virtual CNC should not search through the main CNC program looking for subroutines and sub programs. The TOP option requires all subroutines and sub programs to be located at the beginning of the file before the main program. The BOTTOM option requires all subroutines and sub programs to be located at the end of the file after the main program. The BOTH option allows the subroutines and sub programs to be located at the start or the end of the file.

SUBPROG_POS=NONE ; No sub program support in the CNC
SUBPROG_POS=TOP ; Sub Programs can only be above the main
SUBPROG_POS=BOTTOM ; Sub Programs can only be below the main
SUBPROG_POS=BOTH ; Sub Programs can be at the top or the bottom

Default value: NONE
Parameter: string

SUBPROGEX_NUMBER_DIGITS

Description: Sets the number of digits to identify number based extended subroutines and sub programs.

Tech Tips: Should this number be exceeded, Predator Virtual CNC will trigger appropriate warning messages. Often this parameter is used in combination with SUBPROGEX_NTIMES_DIGITS and SBPROGEX_PROGNUMBER_POS to extract the appropriate information. Examples using this parameter include the following:

SUBPROGEX_NUMBER_DIGITS=4 ; Fanuc style or
SUBPROGEX_NUMBER_DIGITS=2 ; Fadal style

Default value: 4
Parameter: int

SUBPROGEX_NTIMES_DIGITS

Description: Sets the number of digits for a extended subroutine or sub program to repeat.

Tech Tips: Should this number be exceeded, Predator Virtual CNC will trigger appropriate warning messages. Often this parameter is used in combination with SUBPROGEX_NUMBER_DIGITS and SBPROGEX_PROGNUMBER_POS to extract the appropriate information. Examples using this parameter include the following:

SUBPROGEX_NTIMES_DIGITS=4 ; Fanuc style or
SUBPROGEX_NTIMES_DIGITS=2 ; Fadal style

Default value: 0
Parameter: int

SUBPROGEX_PROGNUMBER_POS

Description: Sets the position of the extended subroutine or sub program number within a number that also contains the number of repetitions.

Tech Tips: The FIRST option specifies that the portion of the extended subroutine or sub program number that identifies the sub is at the beginning. The LAST option specifies that the number is at the end.

```
SUBPROGEX_PROGNUMBER_POS=FIRST    ; Fadal style or
SUBPROGEX_PROGNUMBER_POS=LAST     ; Fanuc style
```

Default value: FIRST
Parameter: string

THREAD_CUTTING_FACTOR

Description: Enables threading cycle values to match the machine builder's definitions for threading

Tech Tips: If the number is not zero it will divide all real numbers by the specified value to convert the numbers without decimal points. A value of 0 or 1 will not convert values and assumes that decimal points are used within the CNC program. For example on a Fanuc control the P and Q values are often integers. Examples using this parameter include the following:

```
THREAD_CUTTING_FACTOR=10000  ; Machine Builder A
THREAD_CUTTING_FACTOR=1000   ; Machine Builder B
```

Default value: 1000
Parameter: real

NOTE: This parameter existed prior to the REAL2, REAL3, REAL4 and REAL5 modifiers. This parameter is still supported. The recommended method of handling this is to modify the thread pattern with the appropriate REALx modifier.

TOOL_COMMENTS

Description: Toggles Predator Virtual CNC's ability to scan the CNC program for tool shapes within comments.

Tech Tips: The ON option enables scanning for tool shapes. The OFF option disables scanning and requires the tool shapes to be defined within Predator Virtual CNC. Examples using this parameter include the following:

```
TOOL_COMMENTS=ON          ; Recommended for most applications
TOOL_COMMENTS=OFF
```

Default value: OFF
Parameter: string

TRAVEL_AS_ERROR

Description: Toggles travel limit error reporting.

Tech Tips: The ON option reports exceeding the travel limits as an error. The OFF option reports exceeding the travel limits as a warning. Examples using this parameter include the following:

```
TRAVEL_AS_ERROR=ON        ; Recommended for most applications
TRAVEL_AS_ERROR=OFF
```

Default value: ON
Parameter: string

UNIT_CHANGE

Description: Toggles a CNC programming rule or CNC control requirement to where changing units within a CNC program is allowed.
Tech Tips: The ON option enables changing units within the CNC program. The OFF option disables changing units within the file CNC program and will trigger a warning message if it does occur. Examples using this parameter include the following:

```
UNIT_CHANGE=ON
UNIT_CHANGE=OFF          ; Recommended for most applications
```

Default value: OFF
Parameter: string

USE_DEFAULT_TOOL

Description: Toggles a CNC programming rule or CNC control requirement that forces a default tool to be loaded prior to tool motion.
Tech Tips: The ON option enables a default tool be loaded and disables checking for tool selection prior to any tool motion. The OFF option disables loading a default tool and enables checking for tool selection prior to any tool motion. Examples using this parameter include the following:

```
USE_DEFAULT_TOOL=ON          ; Recommended for routers, lasers, etc.
USE_DEFAULT_TOOL=OFF         ; Recommended for mills and lathes
```

Default value: ON
Parameter: string

VARIABLE_FACTOR

Description: Multiplies all variables by a given factor
Tech Tips None. Examples using this parameter include the following:

```
VARIABLE_FACTOR=1.0
VARIABLE_FACTOR=100.0
```

Default value: 1.0
Parameter: real

VARIABLE_MAX

Description: Sets the maximum number of variable supported by the CNC control.
Tech Tips The maximum number of variables can be from 1 to 512. Examples using this parameter include the following:

```
VARIABLE_MAX=32              ; Fanuc style
VARIABLE_MAX=16
```

Default value: 64
Parameter: int

VARIABLE_SUPPORT

Description: Toggles a CNC programming rule or CNC control requirement to support variables.

Tech Tips: The ON option enables support for variables. The OFF option disables support for variables. Examples using this parameter include the following:

```
VARIABLE_SUPPORT=ON          ; Recommended for most applications
VARIABLE_SUPPORT=OFF
```

Default value: OFF
Parameter: string

WARNING_AS_ERROR

Description: Toggles showing warnings as error or not
Tech Tips: The ON option will force the parser to treat warnings as errors. The OFF option will leave the warnings as warnings. Examples using this parameter include the following:

```
WARNING_AS_ERROR=ON
WARNING_AS_ERROR=OFF
```

Default value: OFF
Parameter: string

WARNING_CUTCOMP_DIAMTOOLARGE

Description: Toggles a warning when the cutter compensation diameter is too large
Tech Tips: The ON option enables the warning. The OFF option disables the warning. Examples using this parameter include the following:

```
WARNING_CUTCOMP_DIAMTOOLARGE=ON
WARNING_CUTCOMP_DIAMTOOLARGE=OFF
```

Default value: OFF
Parameter: string

NOTE: This warning is issed when the length of the original move before compensation is shorter than the compensation amount. This verification check is similar to the WARN_CUTCOMP_DIAMTOOLARGE parameter. If both of these parameters are enabled then Predator Virtual CNC will check before and after the tool compensation.

WORK_OFFSET

Description: Sets the style of work offset values
Tech Tips: The work offset values are either specified from the machine origin or the programming origin. The MACHINE_DATUM option calculates work offset values from the machine origin. The PROGRAMMING option calculates work offset values from the programming origin. Examples using this parameter include the following:

```
WORK_OFFSET=MACHINE_DATUM        ; Fanuc style
WORK_OFFSET=PROGRAMMING          ; Anilam and Bridgeport style
```

Default value: MACHINE_DATUM
Parameter: string

WRITE_COMMENTS

Description: Toggles an option of writing comments within the RVP file

Tech Tips: The ON option is used by OEMs to include comments within the RVP file for further processing. The OFF options disables writing comments within the RVP file. Examples using this parameter include the following:

```
WRITE_COMMENTS=YES          ; Enables option
WRITE_COMMENTS=NO           ; Disables option
```

Default value: YES
Parameter: string

3.4. Error messages:

Sometimes error messages are displayed when the RPM or RPL file is loaded. Fatal error messages will prevent the CNC program to be processed and must be fixed by editing the RPM or RPL file.

3.4.1 Fatal Errors:

The following fatal errors will prevent the CNC program to be processed.

BADCOMMAND:
Error: "Unknown command"
Tech Tips: Review commands for typos

BADFORMAT:
Error: "Unknown format"
Tech Tips: Review pattern formats for typos

BADMODIF:
Error: "Unknown modifier"
Tech Tips: Review pattern modifiers for typos

BADCYCLE:
Error: "Unknown cycle"
Tech Tips: Review cycles for typos

BADCOMMANDINGROUP:
Error: "Command not defined for this group"
Tech Tips: Some commands are only supported within a specific group. Review documentation for addition details.

BADDELIMITER:
Error: "Missing delimiter"
Tech Tips: Review pattern for a required delimiter

BADMAINGROUP:
Error: "Bad group identifier"
Tech Tips: Review patterns for a missing or undefined group

BEGINBRACKETNOTFOUND:
Error: Begin bracket '[' not found
Tech Tips: Review sections for missing begin brackets

Default
Error: "Unknown error 0x%x"
Tech Tips: Review RPM or RPL

ENDBRACKETNOTFOUND:
Error: "End bracket ']' not found"
Tech Tips: Review sections for missing end brackets

GCF_FERROR:
Error: "File Not Found"
Tech Tips: Review include file section for missing include files

GCF_TOOMANYINCLUDES:
Error: "Too Many Include Files in GCF"
Tech Tips: Minimize the number of include files to 32 or less

INTERNALTABLEOVERFLOW:
Error: "Internal tables overflow"
Tech Tips: Review sections for missing end of section definitions

NOENDQUOTES:
Error: "No end quotes found"
Tech Tips: Review patterns that have a begin quote with a missing end quote

NOBEGINQUOTES:
Error: "No begin quotes found"
Tech Tips: Review patterns that have an end quote with a missing begin quote

NOENDPARENTHESIS:
Error: “Missing closing parenthesis”
Tech Tips: Review RPM or RPL for a beginning parenthesis without an ending parenthesis

3.4.2 Non Fatal Errors:

The following non fatal errors will not prevent the CNC program to be processed.

GCF_EOF:
Error: "End of File"
Tech Tips: n/a

PASTENDMARK:
Error: "Data found after [END]"
Tech Tips: Comment the lines after the [END] with the semi colon or delete all lines after the [END]

4. RP* File Example

```
;|------------------------------------------------
;"FAMILY:                FANUC$"
;"NAME:                  GENERIC $"
;"Author:                Jim Abbassian$"
;"Creation Date: Oct-9-99$"
;"Revisions:             A$"
;"$"
;"NOTES:Generic Reverse Post$"
;"$"
;|------------------------------------------------
;
[PATTERNS]                                          ;This must be the first section
@1sorted:#0                                         ;pattern 1  formats the numbers
%d,int_reg                  : code_num              ;before separator
"."                         : code_and_subcode      ;separator . or ,
%d,int_reg                  : subcode_num           ;after separator
                                                    ;
@3:#0                                               ;pattern 3 is for abs/inc
                                                    ;only used in canned cycle
"G90"                       : use_absolute
"G91"                       : use_incremental
                                                    ;
@4:#0                                               ;pattern 4 is for retract planes
                                                    ;only used in canned cycle
"G98"                       : retract_to_init
"G99"                       : retract_to_R
                                                    ;
@2:#0                                               ;pattern 2 is for canned cycles
"",switch@4,casesens        : r_retract_mode        ;calls pattern 4 for retract plane
                                                    ;only allow G98 or G99
"\0"                        : separator             ;(\0=null) allows for no spaces
"",switch@3,casesens        : r_coord_mode          ;calls pattern 3 for abs/inc
                                                    ;only allow G90 or G91
"\0"                        : separator
"X",real                    : r_drill_pos_X         ;horizontal value for drill position
"\0"                        : separator
"Y",real                    : r_drill_pos_Y         ;vertical value for drill position
"\0"                        : separator
"Z",real                    : r_peck_Z              ;standard
;"Z",real                   : r_drill_pos_Z         ;optional drill depth value
"\0"                        : separator
"Q",real                    : r_peck_inc            ;optionally change "Q" to "Z"
"\0"                        : separator
"R",real                    : r_peck_retract        ;optionally change "R" to "Z"
"\0"                        : separator
"L",int                     : r_peck_repeats
"\0"                        : separator
"P",int                     : r_dwell_time
                                                    ;
@5:#0                                               ;reference point return G28/G29
"X",real                    : r_reference_pos_X
"\0"                        : separator
"Y",real                    : r_reference_pos_Y
```

```
"\0"                    : separator
"Z",real                : r_reference_pos_Z
                                                ;
@6:#0                                           ;pattern 6 is for subroutine calls
"P",int                 : r_subprogram_number
"\0"                    : separator
"L",int                 : r_subprogram_times
                                                ;
@7:#0
%s,string_reg           : r_expression
                                                ;
@8:#0
"P",int                 : r_subprogram_number
"\0"                    : separator
"L",int                 : r_subprogram_times
"\0"                    : separator
%s,string_reg           : r_subprogram_params
                                                ;
@9:#0                                           ;scan stock values from G-code
"X",real                : r_stock_minX
"\0"                    : separator
"Y",real                : r_stock_minY
"\0"                    : separator
"Z",real                : r_stock_minZ
"\0"                    : separator
"L",real                : r_stock_lengthX
"\0"                    : separator
"W",real                : r_stock_lengthY
"\0"                    : separator
"H",real                : r_stock_lengthZ
                                                ;
@10:#0                                          ;scan stock values from G-code
"S",int                 : r_stock_cyl_axis      ; S1=X axis, S2=Y axis or S3=Z axis
"\0"                    : separator
"X",real                : r_stock_centerX
"\0"                    : separator
"Y",real                : r_stock_centerY
"\0"                    : separator
"Z",real                : r_stock_centerZ
"\0"                    : separator
"D",real                : r_stock_diameter
"\0"                    : separator
"L",real                : r_stock_length
                                                ;
@11:#0                                          ;scan stock or fix from G-code
%s,string_reg           : r_stock_STL_filename
                                                ;
@12:#0                                          ;scan tool values from G-code
"T",int                 : r_tool_number
"\0"                    : separator
"S",int                 : r_tool_type           ; S1=Flat S2=Ball S3=Bull
                                                ; S4=Drill S5=Radius S6=Sphere
                                                ; S7=Chamfer S8=Dovetail
                                                ; S9=Taperball S10=Taperbull
"\0"                    : separator
"D",real                : r_tool_diameter
```

```
"\0"                    : separator
"C",real                : r_tool_corner_rad
"\0"                    : separator
"A",real                : r_tool_angle
"\0"                    : separator
"H",real                : r_tool_height
                                                ;
@13:#0                                          ;G10 style work offsets
"L",int                 : r_changeoff_func
"\0"                    : separator
"P",int                 : r_changeoff_num
"\0"                    : separator
"X",real                : r_changeoff_Xvalue
"\0"                    : separator
"Y",real                : r_changeoff_Yvalue
"\0"                    : separator
"Z",real                : r_changeoff_Zvalue
                                                ;
@14:#0                                          ;G10 style extended work offsets
"P",int                 : r_work_coord
                                                ;
                                                ;Function Groups start with $
$DATASEPARATOR: #0                              ;allows for no spaces in file
\0,null                 : data_separator        ;null character
;"=",null               : data_separator
;" ",null               : data_separator        ;Space character
;\9,null                : data_separator        ;TAB character used for
;TAB sequential machines
                                                ;
$MOVETO:#10                                     ;G0 - G3 moves
"X",real                : MoveX
"Y",real                : MoveY
"Z",real                : MoveZ
;"X",expression          : MoveX                 ; Optional method
;"Y",expression          : MoveY
;"Z",expression          : MoveZ
"I",real                : MoveI
"J",real                : MoveJ
"K",real                : MoveK
"R",real                : MoveR
"B",real                : Move4thAxis
"C",real                : Move5thAxis
                                                ;
$MOVETYPE:#100
"G00"                   : MoveRapid
"G0"                    : MoveRapid
"G01"                   : MoveLinear
"G1"                    : MoveLinear
"G02"                   : MoveCircularCW
"G2"                    : MoveCircularCW
"G03"                   : MoveCircularCCW
"G3"                    : MoveCircularCCW
"G90"                   : AbsoluteCoord
"G91"                   : IncrementalCoord
"G17"                   : SelectPlaneXY
"G18"                   : SelectPlaneZX
```

```
"G19"                   : SelectPlaneYZ
"G20"                   : Inches
"G21"                   : Millimeters
"G70"                   : Inches
"G71"                   : Millimeters
                                                ;
$TOOLCHANGE:#20
"T",int                 : SelectTool
"M06"                   : LoadTool
"M6"                    : LoadTool
;"T",int                : SelectAndLoadTool     ; Optional method
"MTOOL",multi@12        : CreateMillTool
                                                ;
$MISCELLANEOUS:#11
"M00"                   : Pause0
"M0"                    : Pause0
"M01"                   : Pause1
"M1"                    : Pause1
                                                ; Note ProgramEnd causes
; scanning to stop
"M02"                   : ProgramEnd            ;
"M2"                    : ProgramEnd
"M03"                   : SpindleCW
"M3"                    : SpindleCW
"M04"                   : SpindleCCW
"M4"                    : SpindleCCW
"M05"                   : Spindlestop
"M5"                    : Spindlestop
"M07"                   : CoolantON
"M7"                    : CoolantON
"M08"                   : CoolantON
"M8"                    : CoolantON
"M09"                   : CoolantOFF
"M9"                    : CoolantOFF
"M30"                   : ProgramEnd
"S",int                 : SpindleSpeed
                                                ;
$FEEDRATE:#110
"F",real                : SetFeed
                                                ;
$CANNEDCYCLES:#20
"G80"                   : CancelCannedCycle
"G73",multi@2           : DrillCycle (peck)
"G74",multi@2           : DrillCycle (bottomCW retractfeed)
"G76",multi@2           : DrillCycle (oriented_bottomstop)
"G81",multi@2           : DrillCycle ()
"G82",multi@2           : DrillCycle (dwell)
"G83",multi@2           : DrillCycle (peck dwell)
"G84",multi@2           : DrillCycle (bottomCW dwell retractfeed)
"G85",multi@2           : DrillCycle (retractfeed)
"G86",multi@2           : DrillCycle (bottomstop)
"G87",multi@2           : DrillCycle (bottomstop manualfeed)
"G88",multi@2           : DrillCycle (dwell bottomstop)
"G89",multi@2           : DrillCycle (dwell retractfeed)
"G98"                   : ReturnToInitial
"G99"                   : ReturnToReference
```

```
                                                                ;
$UNSUPPORTED: #1
"G9"                            : Error                         ;exact check stop
"G09"                           : Error
"G11"                           : Error                         ;data entry cancel
"G15"                           : Error                         ;Polar Coodinates cancel
"G16"                           : Error                         ;Polar Coodinates
"G30"                           : Error                         ;return alternate points
"G31"                           : Error                         ;skip cutting
"G33"                           : Error                         ;thread cutting
"G37"                           : Error                         ;auto tool length measurement
"G50"                           : Error                         ;scaling off
"G51"                           : Error                         ;scaling on
"G52"                           : Error                         ;local coordinate system
"G53"                           : Error                         ;machine coordinate sys. setting
"G60"                           : Error                         ;single direction positioning
"G61"                           : Error                         ;exact stop check mode
"G62"                           : Error                         ;auto. corner override mode
"G63"                           : Error                         ;tapping mode
"G64"                           : Error                         ;cutting mode
"G65"                           : Error                         ;user macro simple call
"G66"                           : Error                         ;user macro modal call
"G67"                           : Error                         ;user macro cancel
"G68"                           : Error                         ;coordinate system rotation on
"G69"                           : Error                         ;coordinate system rotation off
"G101"                          : Error                         ;PC Data setting
"G311"                          : Error                         ;multi-step skip function1
"G312"                          : Error                         ;multi-step skip function2
"G313"                          : Error                         ;multi-step skip function3
"G511"                          : Error                         ;mirror
"G661"                          : Error                         ;macro modal call B
                                                                ;
$IGNORED: #1
"G4"                            : Ignore                        ;dwell
"G04"                           : Ignore
"G7"                            : Ignore                        ;feedrate control
"G07"                           : Ignore
"G22"                           : Ignore                        ;stored stroke limit on
"G23"                           : Ignore                        ;stored stroke limit off
"G27"                           : Ignore                        ;reference return point check
"G37"                           : Ignore                        ;tool compensation z
"G93"                           : Ignore                        ;inverse time feed
"G94"                           : Ignore                        ;per minute feed
"G95"                           : Ignore                        ;per revolution feed
"G96"                           : Ignore                        ;constant surface speed
"G97"                           : Ignore                        ;constant surface speed cancel
"M10"                           : Ignore                        ;clamp axis on
"M11"                           : Ignore                        ;clamp axis off
"M13"                           : Ignore                        ;spindle on coolant on
"M14"                           : Ignore                        ;spindle on CCW coolant on
"M19"                           : Ignore                        ;spindle orientation
"M25"                           : Ignore                        ;tool clamp on
"M26"                           : Ignore                        ;tool clamp off
"M27"                           : Ignore                        ;clutch neutral on
"M28"                           : Ignore                        ;clutch neutral off
"M48"                           : Ignore                        ;feedrate spindle enable
```

```
"M49"                   : Ignore                ;feedrate spindle disable
"M58"                   : Ignore                ;disarm spindle probe
"M59"                   : Ignore                ;arm spindle probe
"M60"                   : Ignore                ;chip conveyor on
"M61"                   : Ignore                ;chip conveyor off
;"M98"                  : Ignore                ;call subroutine
;"M99"                  : Ignore                ;back to main from subroutine
"N",int                 : Ignore
"O",int                 : Ignore
                                                ;
$REFPOINTRETURN:#20
"G28",multi@5           : ReturnToRefP
"G29",multi@5           : ReturnFromRefP
                                                ;
$CUTTERCOMP:#20
"D",int                 : SetCutterCompD
"H",int                 : SetLengthCompH
"G40"                   : CutterCompCancel
"G41"                   : CutterCompLeft
"G42"                   : CutterCompRight
"G43"                   : LengthCompPlus
"G44"                   : LengthCompMinus
"G45"                   : LengthOffPositive
"G46"                   : LengthOffNegative
"G47"                   : LengthOffDoublePositive
"G48"                   : LengthOffDoubleNegative
"G49"                   : LengthCompCancel
                                                ;
$WORKCOORD: #30
"G54"                   : SetWorkCoord1
"G55"                   : SetWorkCoord2
"G56"                   : SetWorkCoord3
"G57"                   : SetWorkCoord4
"G58"                   : SetWorkCoord5
"G59"                   : SetWorkCoord6
"G10",multi@13          : ChangeOffset
"G54.1",multi@14        : SetWorkCoord
;"E",int                : SetWorkCoord          ; Optional method
                                                ;
$ABSOLUTEZERO: #5
"G92"                   : SetAbsoluteZero
                                                ;
$SUBPROGRAM: #5
"O",int                 : SubProgramStart
"M99"                   : SubProgramRet
"M98",multi@6           : SubProgramCall
                                                ;
;$VARIABLES: #4
;"#",int                : GetGlobalVar
;"=",multi@7            : SetGlobalVar
                                                ;
$STOCKDEF: #5
"SBOX",multi@9          : CreateStockBox
"FBOX",multi@9          : CreateStockFixt
"HBOX",multi@9          : CreateStockHole
"SCYL",multi@10         : CreateStockCyl
```

```
"FCYL",multi@10                 : CreateStockCylFixt
"HCYL",multi@10                 : CreateStockCylHole
"SSTL",multi@11                 : CreateStockSTL
"FSTL",multi@11                 : CreateStockSTLFixt
                                                                ;
$COMMENTS:#1
"%",null                        : CommentFollows
"/",null                        : CommentFollows
"(",null                        : CommentStarts
")",null                        : CommentEnds
                                                                ;
[EXCLUSIVE]                                                     ;syntax check for multiple
                                                                ;commands per line
MOVETYPE MoveLinear MoveCircularCW MoveCircularCCW MoveRapid
MOVETYPE SelectPlaneXY SelectPlaneYZ SelectPlaneZX
CUTTERCOMP LengthCompPlus LengthCompMinus LengthCompCancel
                CutterCompCancel CutterCompLeft CutterCompRight;
                                                                ;
[PARAMETERS]
arc_tol_inch=0.005                                              ;sets arc radius check tol. inch
arc_tol_mm=0.50                                                 ;sets arc radius check tol. mm
arc_check=off                                                   ;valid ON or OFF
check_spiral_move=off                                           ;valid ON or OFF w/arc_tol_xxx
check_helical_move=off                                          ;valid ON or OFF w/arc_tol_xxx
                                                                ;
cycle_return=init                                               ;init(initial) R(Retract Point)
drillcycle_retract=0.0                                          ;retract override
                                                                ;
loffset_axis=Z                                                  ; X Y Z or PROG length offset
; along axis
loffset_sign=on                                                 ; ON or OFF w/length offset sign
check_length_offset=off                                         ;valid ON or OFF
                                                                ;
nc_format=ASCII                                                 ;valid ASCII or ASCII72$
rvp_format=ascii                                                ;valid ASCII or BINARY
                                                                ;
check_spindle=off                                               ;valid ON or OFF
check_spindle_speed=on                                          ;valid ON or OFF
                                                                ;
enable_warnings=on                                              ;valid ON or OFF
unit_change=off                                                 ;valid ON or OFF
;ON enables changes within file
check_travels=on                                                ;valid ON or OFF
;ON checks machine travels
check_work_offset=off                                           ;valid ON or OFF
;ON checks work/fixture offsets
use_default_tool=off                                            ;valid ON or OFF
;ON checks for tool select)
                                                                ;
tool_comments=ON                                                ;valid ON or OFF
;ON checks for tool definition
stock_comments=ON                                               ;valid ON or OFF
;ON checks for stock definition
                                                                ;
first_codes=G0G90G20G40G17G80                                   ;control powerup G & M codes
                                                                ;
```

```
subprog_pos=BOTTOM                          ;valid NONE TOP BOTTOM
;BOTH or EXTERNAL
subprog_call=PROG_NUM                       ;valid PROG_NUM
;PROG_LINE PROG_STRING
subprog_nest=4                              ;maximum # nested calls
subprog_max=16                              ;maximum # subprograms
subprog_number_digits=4                     ;maximum # of digits in prog. #
subprog_ntimes_digits=0                     ;maximum # of digits
subprog_max_rep=999                         ;maximum # of repeats
subprog_prognumber_pos=FIRST
subprog_external=OFF                        ;valid ON OFF
                                            ;
;macro_arg_type=type2                       ;valid type1 or type2
;TYPE1 is default Fanuc style
;macro_arg_max=5                            ;maximum # of macro
;arguments or parameters
;macro_arg_list=A,1,B,2,C,3,D,4,E,5         ;type2 style allows argument or
; parameter definition and order
                                            ;
;variable_support=ON                        ;valid ON or OFF
;variable_max=32                            ;maximum # of variables
[END]
```

1.4 Error checking

The reverse post is capable to detect errors both in the NC Code as well as errors in the configuration or RP* file.

1.4.1 RP* Configuration Load and Syntax Errors

1.4.1.1 RP* Errors

The reverse post detects a wide range of syntax and configuration errors while reading the RP* file. Errors are typically introduced when support for new G and M codes are added to an existing RP* file. Edit the offending RP* file and use the Tech Tips, outlined below, to help fix each and every error message. For example, use the included Predator Editor to edit the FADAL-CNC88.RPM file to add support for subroutine calls and continue to edit it until the reverse post stops issuing error messages.

Error codes are for internal use for OEMs and SDK developers who are using Predator Virtual CNC technology in their products. Most users of Predator Virtual CNC should ignore the internal error code values.

GCF_SECTION

Description:	Section Name Invalid
Tech Tip:	Check spelling of existing section names within the RP* file.
Error code:	0x000000ff

TOGCF_ERROR

Description:	Error opening RP* File
Tech Tip:	Check the RP* file and verify that you have permission to open the file and that it is not corrupt.
Error code:	0x80000000

TOGCF_INVALID_INPUT_FILE

Description: Error opening RPx File
Tech Tip: Check the RPx file and verify that you have permission to open the file and that it is not corrupt.
Error code: 1

TOGCF_INVALID_OUTPUT_FILE
Description: Error opening RP* File
Tech Tip: Check the RP* file and verify that you have permission to open the file and that enough free disk space is available.
Error code: 2

PASTENDMARK
Description: Error opening RP* File
Tech Tip: Check the RP* file and verify that you have permission to open the file and that it is not corrupt.
Error code: 0x80000100

BEGINBRACKETNOTFOUND
Description: Section start bracket not found
Tech Tip: Check existing section names within the RP* file and add the start bracket.
Error code: 0x80000200

ENDBRACKETNOTFOUND
Description: Section end bracket not found
Tech Tip: Check existing section names within the RP* file and add the end bracket after the section name.
Error code: 0x80000400

INTERNALTABLEOVERFLOW
Description: Internal table overflow
Tech Tip: The RP* file contains either too many RP* include files or too many multiple format entries.
Error code: 0x80000800

NOENDQUOTES
Description: String has no end quotes
Tech Tip: Check existing strings within the RP* file and add a quote after the string.
Error code: 0x80001000

NOBEGINQUOTES
Description: String has no begin quotes
Tech Tip: Check existing strings within the RP* file and add a quote before the string.
Error code: 0x80002000

BADMAINGROUP
Description: Internal group index wrong
Tech Tip: Check existing groups within the RP* file.
Error code: 0x80004000

GCF_EOF
Description: RP* end of file
Tech Tip: Technically this is not an error, it is a status message.
Error code: 0x80008000

GCF_FERROR
Description: File Open error

Tech Tip: Check the RP* file and verify that you have permission to open the file and that it is not corrupt.
Error code: 0x80010000

GCF_TOOMANYINCLUDES
Description: Exceeded maximum number of includes
Tech Tip: Merge some of the RP* include files into one larger RP* file.
Error code: 0x80020000

BADCOMMAND
Description: Command name not recognized.
Tech Tip: Check spelling of existing command names within the RP* file.
Error code: 0x80040000

BADFORMAT
Description: Format expression not recognized
Tech Tip: Check spelling of existing formats within the RP* file.
Error code: 0x80080000

BADMODIF
Description: Format modifier not recognized
Tech Tip: Check spelling of existing format modifiers within the RP* file.
Error code: 0x80100000

BADCYCLE
Description: Cycle name not recognized
Tech Tip: Check spelling of existing cycles within the RP* file.
Error code: 0x80200000

BADCOMMANDINGROUP
Description: Command group not recognized
Tech Tip: Check spelling of existing command groups within the RP* file.
Error code: 0x80400000

BADDELIMITER
Description: Delimiter not allowed
Tech Tip: Check for typos of existing delimiters within the RP* file.
Error code: 0x80800000

NOENDPARENTHESIS
Description: Parenthesis opened but not closed
Tech Tip: Check existing parenthesis within the RP* file and add the appropriate closed parenthesis.
Error code: 0x81000000

1.4.2 NC and CNC Program Syntax and Semantic checking

The reverse post detects a wide range of syntax and semantic errors while reading NC and CNC programs. These errors will optionally be appended to an error file unless the ENABLE_WARNINGS=ON parameter is found in the RP* file. Errors are typically introduced when operators and machinists edit existing NC and CNC programs or if the CAM system's post processor is having problems or is being initially configured. Edit the NC or CNC program using the Tech Tips, outlined below, to help fix each and every error message. For example, use the included Predator Editor to edit the JOB1234.NC file until the reverse post stops issuing error messages.

Error codes are for internal use for OEMs and SDK developers who are using Predator Virtual CNC technology in their products. Most users of Predator Virtual CNC should ignore the internal error code values.

1.4.2.1 NC and CNC Program Error messages

Syntactic error codes:

SYNTAX_REVERSE POST_ERR_EXCLUSIVES
- Description: Exclusive commands found in the same block or line
- Tech Tips: Check for illegal combinations of G & M codes on the same block or line. For example, it is illegal on a Fanuc control to have G0 and G1 codes in the same block or line. Other Fanuc examples would be G17, G18, and G19 codes on the same block or line.
- Error Code: 0x00000001L

SYNTAX_REVERSE POST_ERR_UNKNOWNPAT
- Description: Pattern unknown
- Tech Tips: Check for typos, rare canned cycles or other G & M codes that are not used very often and update the RP* to support or ignore them. The reverse post is seeing additional data that it is not expecting to see within a block or line.
- Error Code: 0x00000002L

SYNTAX_REVERSE POST_ERR_MISSINGPATDELIM
- Description: Missing Pattern Delimiter
- Tech Tips: Check for typos within a series of G & M codes, canned cycles or subroutines.
- Error Code: 0x00000004L

SYNTAX_REVERSE POST_ERR_MISSINGMULTIDATA
- Description: Missing Multiple Data pattern
- Tech Tips: Check for G & M codes, canned cycles or subroutine calls, etc that require secondary values that are currently not specified. For example, on a Fanuc control a M98 is specified without a P or R words on the same block or line.
- Error Code: 0x00000008L

SYNTAX_REVERSE POST_ERR_INCOMPMULTIDATA
- Description: Incomplete Multiple data pattern
- Tech Tips: Check for G & M codes, canned cycles or subroutine calls, etc that require multiple values where one of them is missing. For example, on a Fanuc control a M98 is specified with an R word but without a P word on the same block or line.
- Error Code: 0x00000010L

SYNTAX_REVERSE POST_ERR_DUPLMULTIDATA
- Description: Duplicated pattern in multiple data pattern
- Tech Tips: Check for G & M codes, canned cycles or subroutine calls, etc that require multiple values where one of them is duplicated. For example, on a Fanuc control a M98 is specified and two P words are specified on the same block or line.
- Error Code: 0x00000020L

SYNTAX_REVERSE POST_ERR_UNKNOWNKEYW
- Description: Keyword unknown
- Error Code: 0x00000040L

SYNTAX_REVERSE POST_ERR_MISSINGSTRDATA
- Description: Missing string pattern
- Tech Tips: Check for G & M codes, canned cycles or subroutine calls, etc that require secondary values that are currently not specified. For example, on a Vickers 2100 control a CLS is specified without an alphanumeric subroutine name on the same block or line.
- Error Code: 0x00000080L

SYNTAX_REVERSE POST_ERR_MISSINGINTDATA
- Description: Missing integer pattern
- Tech Tips: Check for G & M codes that require secondary values that are currently not specified. For example, on a Fanuc control a N letter is specified without an actual sequence number.
- Error Code: 0x00000100L

SYNTAX_REVERSE POST_ERR_MISSINGFLOATDATA
- Description: Missing float pattern
- Error Code: 0x00000200L

SYNTAX_REVERSE POST_ERR_INTDATAEXPECTED
- Description: Integer data expected
- Tech Tips: Check for G & M codes that require a secondary value that is currently not specified. For example, on a Fanuc control a sequence number or N is specified without a number afterwards.
- Error Code: 0x00000400L

Semantic error codes:

SEMANTIC_REVERSE POST_ERR_TOOMANYERRORS
- Description: Too many errors. Limit reached.
- Tech Tip: Fix the errors in the NC or CNC programs or verify that the correct Machine/Control combination is specified in the job.
- Error Code: 0x00000001L

SEMANTIC_REVERSE POST_ERR_TOOLNOTFOUND
- Description: Tool not found
- Tech Tip: The NC or CNC programs must have at least one tool specified.
- Error Code: 0x00000002L

SEMANTIC_REVERSE POST_ERR_TOOLNOTLOADED
- Description: Tool not loaded in tool move
- Tech Tip: The NC or CNC program must have at least one tool loaded in the spindle prior to any tool motion.
- Error Code: 0x00000004L

SEMANTIC_REVERSE POST_ERR_SPINDLEOFF
- Description: Spindle off in tool move
- Tech Tip: The NC or CNC program must have turned on the spindle prior to any tool motion.
- Error Code: 0x00000008L

SEMANTIC_REVERSE POST_ERR_SPINDLESPEEDOFF
- Description: Spindle Speed off in tool move
- Tech Tip: The NC or CNC program must have specified a spindle speed prior to any tool motion.

Error Code: 0x00000010L

SEMANTIC_REVERSE POST_ERR_NOFEEDSET
Description: Feed not set in tool move
Tech Tip: The NC or CNC program must have specified a feed rate prior to any tool motion.
Error Code: 0x00000020L

SEMANTIC_REVERSE POST_ERR_CENTERARCWRONG
Description: Center arc is wrong.
Tech Tip: The NC or CNC program has specified a circular interpolation move that exceeds the specified arc tolerance.
Error Code: 0x00000040L

SEMANTIC_REVERSE POST_ERR_CENTERARCMISSING
Description: Center of arc is missing.
Tech Tip: The NC or CNC program has specified a circular interpolation move without any center coordinates. For example, on a Fanuc control a G2 or G3 move is specified without any I, J, or K values afterwards.
Error Code: 0x00000080L

SEMANTIC_REVERSE POST_ERR_CENTERARC_R_BAD
Description: Arc Radius incorrect.
Tech Tip: The NC or CNC program has specified a circular interpolation move with an invalid radius. For example, on a Fanuc control a G2 or G3 move is specified with an R value that is too large.
Error Code: 0x00000100L

SEMANTIC_REVERSE POST_ERR_CYCLENOTFOUND
Description: Cycle not specified
Tech Tip: The NC or CNC program has specified a canned cycle that is undefined. For example, an unsupported rigid tapping cycle is specified.
Error Code: 0x00000200L

SEMANTIC_REVERSE POST_ERR_CYCLEREPEATLIMIT
Description: Cycle repeated above limit.
Tech Tips: Check canned cycles or subroutine calls that have been called too many times and exceeds the CNC control's limits. For example, on a Fadal control the maximum number of times a subroutine can be repeated is 99 times.
Error Code: 0x00000400L

SEMANTIC_REVERSE POST_ERR_HOFFSET_NOTSET
Description: H Offset value not set
Tech Tips: The NC or CNC program has not specified a length offset that is defined in the Job setup. For example, on a Fanuc control a H1 word is not specified in the CNC program.
Error Code: 0x00002000L

SEMANTIC_REVERSE POST_ERR_HOFFSET_MULTIPLEAXIS
Description: H Compensation along more than one axis.
Tech Tips: The NC or CNC program has specified a length offset along more than one axis. For example, on a mill with a Fanuc control a H1 word is specified to mean tool length offsets in the Z axis.
Error Code: 0x00004000L

SEMANTIC_REVERSE POST_ERR_HOFFSET_NOTINSETUP
Description: H Offset value not found in Setup
Tech Tips: The NC or CNC program has specified a length offset that is not defined in the Job setup. For example, on a Fanuc control a H1 word is specified in the CNC program.
Error Code: 0x00008000L

SEMANTIC_REVERSE POST_ERR_XTRAVELOUT
Description: X axis travels out of limits
Tech Tips: The NC or CNC program has specified tool motion in the X axis that exceeds the machine limits. Either move this job to a larger machine, move the fixture on the table to get more travel, or split the job into multiple setups.
Error Code: 0x00010000L

SEMANTIC_REVERSE POST_ERR_YTRAVELOUT
Description: Y axis travels out of limits
Tech Tips: The NC or CNC program has specified tool motion in the Y axis that exceeds the machine limits. Either move this job to a larger machine, move the fixture on the table to get more travel, or split the job into multiple setups.
Error Code: 0x00020000L

SEMANTIC_REVERSE POST_ERR_ZTRAVELOUT
Description: Z axis travels out of limits
Tech Tips: The NC or CNC program has specified tool motion in the Z axis that exceeds the machine limits. Either move this job to a larger machine, move the fixture on the table to get more travel, or split the job into multiple setups.
Error Code: 0x00040000L

SEMANTIC_REVERSE POST_ERR_4THAXISTRAVELOUT
Description: 4th axis travels out of limits
Tech Tips: The NC or CNC program has specified a 4^{th} axis rotary motion that exceeds the machine rotary limits. Either move this job to a machine with more rotary travel, move the fixture on the table to get more travel, or split the job into multiple setups.
Error Code: 0x00080000L

SEMANTIC_REVERSE POST_ERR_5THAXISTRAVELOUT
Description: 5th axis travels out of limits
Tech Tips: The NC or CNC program has specified a 5^{th} axis rotary motion that exceeds the machine rotary limits. Either move this job to a machine with more rotary travel, move the fixture on the table to get more travel, or split the job into multiple setups.
Error Code: 0x00100000L

SEMANTIC_REVERSE POST_ERR_CCOMP_NOFARDATA
Description: Cutter Comp cannot find data
Tech Tips: The NC or CNC program has specified a cutter compensation without all of the necessary information.
Error Code: 0x00200000L

SEMANTIC_REVERSE POST_ERR_DOFFSET_NOTSET
Description: D-Offset not set.

Tech Tips: The NC or CNC program has not specified a diameter offset that is defined in the Job setup. For example, on a Fanuc control a D1 word is not specified in the CNC program.
Error Code: 0x00400000L

SEMANTIC_REVERSE POST_ERR_DOFFSET_NOTINSETUP
Description: D-Offset value not found in Setup
Tech Tips: The NC or CNC program has specified a diameter offset that is not defined in the Job setup. For example, on a Fanuc control a D1 word is specified in the CNC program.
Error Code: 0x00800000L

SEMANTIC_REVERSE POST_ERR_OFFSETDURINGREFPOINT
Description: Offset set during free point
Error Code: 0x01000000L

SEMANTIC_REVERSE POST_ERR_ORIGINSHIFTNOTDEFINED
Description: Origin Shift not defined
Tech Tips: The NC or CNC program has not specified a work offset or origin shift. For example, on a Fanuc control a G54, G55, G56, etc word is not specified in the CNC program.
Error Code: 0x02000000L

SEMANTIC_REVERSE POST_ERR_SPIRALMOVE
Description: Spiral move detected, when no spiral moves allowed
Tech Tips: The NC or CNC program has circular interpolation values that exceed the arc tolerance. For example, on a Fanuc 6 control G2 and G3 motion supports circular arcs only and does not support spiral arcs in the CNC program.
Error Code: 0x04000000L

SEMANTIC_REVERSE POST_ERR_HELICALMOVE
Description: Helical move detected, when no helical moves allowed
Tech Tips: Check the NC or CNC program for circular interpolation values with Z axis words specified and delete them, convert them to linear motion, or change your post processor. For example, on a Fanuc control G2 and G3 motion with a Z-.5 word would cut a helix to a -.5 depth.
Error Code: 0x08000000L

SEMANTIC_REVERSE POST_ERR_SPINDLESPEED_TOHIGH
Description: Spindle speed above limit
Tech Tips: The NC or CNC program has specified a spindle speed that exceeds the machine limits. Either move this job to a machine with a faster spindle, or edit the CNC program and slow down the spindle speed.
Error Code: 0x10000000L

SEMANTIC_REVERSE POST_ERR_INVALIDWORKCOORD
Description: Work coordinates invalid
Error Code: 0x20000000L

SEMANTIC_REVERSE POST_ERR_CYCLEREFNOTDEFINED
Description: Cycle ref point undefined
Error Code: 0x40000000L

SEMANTIC_REVERSE POST_ERR_CYCLEDEPTHNOTDEFINED
Description: Cycle depth undefined

Error Code: 0x80000000L

SEMANTIC_REVERSE POST_ERR_PECKINCREMENTNOTDEFINED
Error Code: 0x80000004L
Tech Tips: Check the NC or CNC program for a drill cycles that require a peck depth which is not specified in the CNC program. For example, on a Fanuc control a G83 cycle is specified without a Q word.
Description: Cycle peck increment not defined

SEMANTIC_REVERSE POST_ERR_SUBROG_ERROR
Description: Subprogram not defined
Tech Tips: Check the NC or CNC program for subprograms. For example, on a Vickers 2100 control a DFS block is not specified in the CNC program.
Error Code: 0x40000000L

SEMANTIC_REVERSE POST_ERR_SUBPROGNOTFOUND
Description: Subprogram not found
Tech Tips: Check the NC or CNC program for subprogram calls to missing or undefined sub programs. For example, on a Fanuc control a M98 P1234 is specified in the CNC program and no P1234 exists in the CNC program.
Error Code: 0x40000001L

SEMANTIC_REVERSE POST_ERR_SUBPROGALREADYDEFINED
Description: Subprogram already defined
Tech Tips: Check the NC or CNC program for duplicate subprograms. For example, on a Vickers 2100 control multiple DFS blocks with the same subprogram name are not allowed within the same CNC program.
Error Code: 0x40000002L

SEMANTIC_REVERSE POST_ERR_SUBPROGDEFINEDBEFOREMAIN
Description: Subprogram defined before main program
Tech Tips: Check the NC or CNC program for subprograms that are defined before the main CNC program. For example, on a Vickers 2100 control the main program must come first followed by any subprograms. This requirement is configured by setting the SubProg_Pos=BOTTOM setting within the VICKERS 2100.RPM file.
Error Code: 0x40000004L

SEMANTIC_REVERSE POST_ERR_SUBPROGDEFINEDAFTERMAIN
Description: Subprogram defined after main program
Tech Tips: Check the NC or CNC program for subroutines that are defined after the main CNC program. For example, on a Fadal CNC88 control the subroutines must come first followed by the main program. This requirement is configured by setting the SubProg_Pos=TOP setting within the FADAL CNC88.RPM file.
Error Code: 0x40000008L

SEMANTIC_REVERSE POST_ERR_TOOMANYSUBPROGDEFINED
Description: Too many subprograms defined
Tech Tips: Check the NC or CNC program for more subroutines than the control supports. For example, on a Fanuc 10 control the maximum number of subroutines is 16. This maximum is configured by setting the SubProg_Max=16 setting within the FANUC 10.RPM file.
Error Code: 0x40000010L

SEMANTIC_REVERSE POST_ERR_TOOMANYSUBPROGNESTED

Description: Too many subprograms nested
Tech Tips: Check the NC or CNC program for more nested subroutines than the control supports. For example, on a Fanuc 10 control the maximum number of nested subroutines is 4. This nested maximum is configured by setting the SubProg_Nest=4 setting within the FANUC 10.RPM file.
Error Code: 0x40000020L

SEMANTIC_REVERSE POST_ERR_SUBPROGBADDEFINITION
Description: Too many subprograms nested
Tech Tips: Check the NC or CNC program for incorrect definition of the beginning of a subroutine. For example, on a Vickers 2100 control the beginning of a subroutine must start with a DFS at the beginning.
Error Code: 0x40000040L

SEMANTIC_REVERSE POST_ERR_TOOMANYSUBPROGDIGITS
Description: Too many subprogram digits defined
Tech Tips: Check the NC or CNC program for subprogram numbers larger than the control supports. For example, on a Fanuc 10 control the maximum subprogram is 9999 or four digits. This maximum is configured by setting the SubProg_Number_Digits=4 setting within the FANUC 10.RPM file.
Error Code: 0x40000080L

SEMANTIC_REVERSE POST_ERR_SUBPROGRETWITHOUTCALL
Description: Attempt to return from subprogram without previous call
Tech Tips: Check the NC or CNC program for a return from a subprogram without a subprogram call. For example, on a Fanuc 10 control a M99 without an earlier reference to an M98 is illegal.
Error Code: 0x40000100L

SEMANTIC_REVERSE POST_ERR_TOOMANYSUBPROGCALLREP
Description: Too many repetitive subprogram calls
Tech Tips: Check the NC or CNC program for the number of repetitive calls to a subprogram larger than the control supports. For example, on a Fadal control the maximum subprogram is 99 or two digits. This maximum is configured by setting the SubProg_Ntimes_Digits=2 setting within the FADAL CNC88.RPM file.
Error Code: 0x40000200L

SEMANTIC_REVERSE POST_ERR_BADSUBPROGRAMSTRUCTURE
Description: Bad subprogram structure
Error Code: 0x40000400L

SEMANTIC_REVERSE POST_ERR_VARIABLENOTFOUND
Description: Variable not found
Error Code: 0x40000800L

SEMANTIC_REVERSE POST_ERR_VARIABLETOOMANYDEFINED
Description: Too many variables defined
Tech Tips: Check the NC or CNC program for more variables than the control supports. For example, on a Fanuc 12 control the maximum number of variables is 32. This maximum limit is configured by setting the Variable_Max=32 setting within the FANUC 12.RPM file.
Error Code: 0x40001000L

SEMANTIC_REVERSE POST_ERR_VARIABLENOTSUPPORTED
Description: Variables not supported

Error Code: 0x40002000L

1.4.2.2 NC Warning messages

The reverse post detects a wide range of warnings while reading NC and CNC programs. Errors catch illegal G & M code programming based on the CNC controls capabilities. Warnings can be used to improve manufacturing processes and enforce good CNC programming practices. Warnings catch problems with CNC programs that technically are not illegal but are probably not a good idea. For example, although it is technically possible to be able to switch between inch and metric programming right in the middle of a CNC program it is probably not a good idea. For example, on a Fanuc 12 either a G70 or a G71 code should be present but not both. Discouraging this programming technique can be configured by setting the Unit_Change=Off setting within the FANUC 12.RPM file.

Warnings messages are typically introduced when operators and machinists edit existing NC and CNC programs or if the CAM system's post processor is having problems or is being initially configured. Edit the NC or CNC program using the Tech Tips, outlined below, to help fix each and every warning message. For example, use the included Predator Editor to edit the JOB1234.NC file until the reverse post stops issuing warning messages.

Error codes are for internal use for OEMs and SDK developers who are using Predator Virtual CNC technology in their products. Most users of Predator Virtual CNC should ignore the internal error code values.

REVERSE POST_WARN_CODEIGNORED
- Description: Code ignored
- Tech Tips: Allows specific codes to be ignored by Predator Virtual CNC. For example, on a Fanuc 12 control sequence numbers are not relevant to verification or CNC simulation. Ignoring sequence numbers can be configured by setting the "N",int :_ignore setting within the FANUC 12.RPM file.
- Error Code: 0x00000001L

REVERSE POST_WARN_UNITSIGNORED
- Description: Unit change ignored
- Tech Tips: Allows a unit change to cause a warning by Predator Virtual CNC. For example, on a Fanuc 12 either a G70 or a G71 code should be present but not both. Discouraging this programming technique can be configured by setting the Unit_Change=Off setting within the FANUC 12.RPM file.
- Error Code: 0x00000002L

REVERSE POST_WARN_TRAVELSIGNORED
- Description: Machine Travels ignored
- Tech Tips: Allows exceeding the machine's travel limits to cause a warning by Predator Virtual CNC. For example, on a Fadal 4020 VMC exceeding 40 inches of motion in the X axis should display a warning so a second setup or use of a larger capacity machine is used. Encouraging this programming technique can be configured by setting the Check_Travels=On setting within the FADAL CNC88.RPM file.
- Error Code: 0x00000004L

REVERSE POST_WARN_SPINDLEOFF
- Description: Spindle not set
- Tech Tips: Allows any tool motion prior to turning on the spindle to cause a warning by Predator Virtual CNC. For example, on a Fanuc 12 either a M3 or a M4 code should be present before any X, Y, or Z tool motion. Encouraging this programming technique can be configured by setting the Check_Spindle=On setting within the FANUC 12.RPM file.

Error Code: 0x00000008L

REVERSE POST_WARN_SPINDLESPEEDOFF
Description: Spindle speed not set
Tech Tips: Allows any tool motion prior to spindle rotation greater then zero to cause a warning by Predator Virtual CNC. For example, on a Fanuc 12 a S5000 or appropriate spindle speed code should be present before any X, Y, or Z tool motion. Encouraging this programming technique can be configured by setting the Check_Spindle_Speed=On setting within the FANUC 12.RPM file.
Error Code: 0x00000010L

REVERSE POST_WARN_NOFEEDSET
Description: Feed not set
Error Code: 0x00000020L

REVERSE POST_WARN_HOFFSET_NOTSET
Description: Length offset not set
Error Code: 0x00000040L

REVERSE POST_WARN_HOFFSET_MULTIPLEAXIS
Description: Length offset defined in more than one axis
Error Code: 0x00000080L

REVERSE POST_WARN_HOFFSET_NOTINSETUP
Description: Length offset not in setup
Error Code: 0x00000100L

REVERSE POST_WARN_DOFFSET_NOTSET
Description: Diameter offset not set
Error Code: 0x00000200L

REVERSE POST_WARN_DOFFSET_NOTINSETUP
Description: Diameter offset not in setup
Error Code: 0x00000400L

REVERSE POST_WARN_ORIGINSHIFTNOTDEFINED
Description: Origin shift not defined
Error Code: 0x00000800L

REVERSE POST_WARN_SPIRALMOVE
Description: Spiral move
Tech Tips: Allows spiral interpolation greater then the arc tolerance to cause a warning by Predator Virtual CNC. For example, on a Fanuc 3000 spiral interpolation is not possible only circular interpolation should be used. Encouraging this programming technique can be configured by setting the Check_Spiral_Move=On setting within the FANUC 3000.RPM file.
Error Code: 0x00001000L

REVERSE POST_WARN_HELICALMOVE
Description: Helical move
Tech Tips: Allows helical interpolation greater then the arc tolerance to cause a warning by Predator Virtual CNC. For example, on a Fanuc 3000 helical interpolation is not possible only circular interpolation should be used. Encouraging this programming technique can be configured by setting the Check_Helical_Move=On setting within the FANUC 3000.RPM file.

Error Code: 0x00002000L

REVERSE POST_WARN_SPINDLESPEED_TOHIGH
Description: Spindle speed too high
Tech Tips: Allows exceeding the machine's travel limits to cause a warning by Predator Virtual CNC. For example, on a Fadal 4020 VMC exceeding S10000 should display a warning. Encouraging this machine check can be configured by editing the machine definition within Predator Virtual CNC or manually within the FADAL 4020 VMC.MCH.
Error Code: 0x00004000L

REVERSE POST_WARN_INVALIDWORKCOORD
Description: Invalid work coordinates
Error Code: 0x00008000L

Long Index:

1. Introduction **2**
1.1 Process and Data Flow Structure **3**
1.2 Features Supported: **3**
1.2.1 Features already supported: **3**
1.2.2 Check Areas: **4**
1.2.3 Features not supported: **5**
1.3 Reverse Post files: **5**
2. Reverse Post File Contents **5**
3. RP* Files **6**
3.1 Section identifiers: **6**
[Include] 7
[Initializations] 7
[Patterns] 7
[Exclusive] 8
[Parameters] 8
[End] 9
3.2 Patterns – Section Details **10**
Multiple Pattern Definition 10
Function Groups: 10
Pattern: 11
3.2.1 Function Groups **11**
MOVETO 12
MOVETYPE 13
TOOLCHANGE 13
CANNEDCYCLES 14
MISCELLANEOUS 14
FEEDRATE 15
UNSUPPORTED 15
REFPOINTRETURN 15
CUTTERCOMP 16
IGNORE 16
WORKCOORD 16
ABSOLUTEZERO 17
THREADING 17
SUBPROGRAMS 17
VARIABLES 17
DONOTHING 18
COMMENTS 18
OPERATORS – Reserved for Backwards Compatibility 18
FUNCTIONS – Reserved for Backwards Compatibility 19
LATHECYCLES **19**
PROGRAMMING **19**
ARITHMETIC FUNCTIONS **20**
NOTE: The * are for ATL and NC file scanning support. **21**
3.2.2 Pattern format **21**
3.2.3 FORMAT Keywords **21**
Casesens 22

Concat 22
Expression 23
Group – Reserved for future use 23
Int 23
Int_reg 23
Multi@# 24
Null 24
Optional2 24
Optional3 25
Optional4 25
Real 25
Real2 26
Real3 26
Real4 26
Real_reg 26
Real_reg2 27
Real_reg3 27
Real_reg4 27
String 27
String_reg 27
Switch@# 28
Void 28
3.2.4 Multiple Pattern Secondary Format Keywords 29
Delimiters 29
Multigstrictlen 30
Multigstrictord 30
3.2.5 FUNCTION keywords: 30
3.2.5.1 MOVETYPE Function Group 30
AbsoluteCoord 31
CircularMoveSense 31
CircleCenter 31
CancelPolarCoord 32
CancelRotation 32
CancelScaling 32
DefUnits 32
Dwell 33
IncrementalCoord 33
Inches 33
Millimeters 33
MoveCircular 34
MoveCircularCW 34
MoveCircularCCW 34
MoveLinear 35
MoveRapid 35
Multax 35
PolarCenterAbs 35
PolarCenterInc 36
PolarCoord 36
PolarMoveLinear 36
PolarMoveCircular 36
Rotation 36

RotationInc 37
Scaling 37
ScalingInc 37
SetABSenseCW 37
SetABSenseCCW 38
SelectPlaneXY 38
SelectPlaneYZ 38
SelectPlaneZX 38
SelectPlane 39
3.2.5.2 GROUP: MOVETO 39
IncMoveX 39
IncMoveY 40
IncMoveZ 40
InsertMove 40
Move4thAxis 40
Move5thAxis 41
MoveA 41
MoveB 41
MoveFromXYZ 41
MoveInMCHCoord 42
MoveIJKUVW 42
MoveI 42
MoveJ 42
MoveK 43
MoveR 43
MoveX 43
MoveX- 43
MoveY 44
MoveY- 44
MoveZ 44
MoveZ- 44
MoveXYZIJK 45
MoveXYZUVW 45
PolarAngle 45
PolarRadius 45
ProcessParamCycle 46
3.2.5.3 GROUP: TOOLCHANGE 46
CreateLatheTool 46
CreateMillTool 47
GoHome 47
LoadTool 47
LoadTool2 47
LoadTool3 47
LoadTool4 48
SelectTool 48
SelectAndLoadTool 48
SetTLBFileName 48
3.2.5.4 GROUP: CANNEDCYCLES 49
ArcCycle 49
ArcHoleCycle 49
BoreCycle 50

CircleCycle 50
CallCycle 50
CancelCannedCycle 50
CancelPatternCycle 51
CircPocketCycle 51
CircPocketCycleCW 51
CircPocketCycleCCW 51
CirclePocketCycleDef 52
CycleDepth 52
DefCycle 52
DrillCycle 53
LineCycle 53
LinearHoleCycle 53
MatrixHoleCycle 53
PatternCycle 54
RectInFaceCycle 54
RectOutFaceCycle 54
RectPocketCycle 54
RectPocketCycleDef 55
ReturnToInitial 55
ReturnToReference 55
SlotPocketCycle 56
3.2.5.5 GROUP: MISCELLANEOUS 56
AirBlast 57
ChangePallet 57
CoolantON 58
CoolantOFF 58
CtantSurfSpeed 58
CtantSurfSpeedCancel 58
DisableRotaryComp 58
EnableRotaryComp 59
HSpindleStop 59
LineNumber 59
LoadMachine 59
MaxSpindleSpeed 60
OptStop 60
Pause0 60
Pause1 60
ProgramEnd 61
RestoreMachine 61
SetMachineDatum 61
SetMCHFileName 61
SetOrigRelocation 62
SpindleCW 62
SpindleCCW 62
SpindleSpeed 62
SpindleStop 63
SpindleStop2 63
SpindleStop3 63
SpindleStop4 63
Stop 64

ToolClamp 64
VSpindleStop 64
ZHome 64
3.2.5.6 GROUP: FEED RATE 65
FeedPerMinute 65
FeedPerRevolution 65
InvTimeFeed 65
OutFeed 65
SetFeed 66
SetMaxFeed 66
3.2.5.7 GROUP: UNSUPPORTED 66
Error 66
3.2.5.8 GROUP: REFERENCE POINT RETURN 67
ReturnToRefP 67
ReturnFromRefP 67
3.2.5.9 GROUP: CUTTER COMPENSATION 67
CutterCompCancel 68
CutterCompLeft 68
CutCompRight 68
LengthCompCancel 68
LengthCompMinus 69
LengthCompPlus 69
LengthOffDoubleNegative 69
LengthOffDoublePositive 69
LengthOffNegative 69
LengthOffPositive 70
SetCutterCompD 70
SetDiamOffset 70
SetLengthOffset 70
SetLengthCompH 71
3.2.5.10 GROUP: IGNORE 71
Ignore 71
3.2.5.11 GROUP: WORK COORDINATES 71
AxisRotation 72
CancelAxisRotation 72
ChangeOffset 72
ChangeAllWorkCoord 73
ChangeProgramOrigin 73
DisableWorkCoord 73
LocalCoordSys 73
SetCoordSystem 73
SetWorkCoord1 74
SetWorkCoord2 74
SetWorkCoord3 74
SetWorkCoord4 74
SetWorkCoord5 75
SetWorkCoord6 75
SetWorkCoord7 – SetWorkCoord100 75
SetWorkCoord 76
SetWorkCoordNext 76
Translate 76

TranslateInc 76
3.2.5.12 GROUP: ABSOLUTE ZERO POSITION 77
ChangeProgramOrigin 77
SetAbsoluteZero 77
3.2.5.13 GROUP: THREADING 77
ThreadCycle 77
ThreadMove 78
3.2.5.14 GROUP: SUBPROGRAM 78
EndMacroModalCall 78
MacroCall 78
MacroModalCall 79
ProgramStart 79
SubProgramCall 79
SubProgramCallandRotation 80
SubProgramEnd 80
SubProgramEx 80
SubProgramRet 80
SubProgramRetEnd 81
SubProgramRetEx 81
SubProgramStart 81
DefineGlobalVar 82
DefineLocalVar 82
DefineParameter 82
GetActiveTool 82
GetActiveDCode 83
GetActiveX 83
GetActiveY 83
GetActiveZ 83
GetActiveA 84
GetActiveB 84
GetActiveC 84
GetCurrentReg 84
GetCurrentVar 85
GetGlobalVar 85
GetLocalVar 85
GetParameter 86
GetVariable 86
SetGlobalVar 86
SetLocalVar 86
SetParameter 86
SetVariable 87
3.2.5.16 GROUP: DONOTHING 87
DoNothing 87
3.2.5.17 GROUP: COMMENTS 87
CommentEnds 88
CommentFollows 88
CommentStarts 88
CreateStockBox 89
CreateStockCyl 89
CreateStockCylFixt 89
CreateStockCylHole 90

CreateStockFixt 90
CreateStockHole 91
CreateStockSTL 91
CreateStockSTLFixt 91
SetSTKFileName 92
3.2.5.21 GROUP: LATHECYCLES 92
EndLabel 92
FaceDrillCycle 92
FaceLongTurningCycle 93
FinishCycle 93
RoughXCycle 93
RoughZCycle 93
StartLabel 94
TurnCycle 94
TurnGrooveCycle 94
TurnTapCycle 95
3.2.5.22 GROUP: PROGRAMMING 95
Condition 95
EndLoop 96
Goto 96
SetLabel 96
StartLoop 96
Then 96
3.2.5.23 GROUP: ARITHMETIC FUNCTIONS 97
Add 97
Divide 97
Exponent 97
Increment 98
Integer 98
Module 98
Multiply 98
Subtract 99
3.2.5.24 GROUP: BOOLEAN FUNCTIONS 99
And 99
Equal 99
Great 100
GreatOrEqual 100
Less 100
LessOrEqual 100
NotEqual 100
Or 101
Xor 101
3.2.5.24 GROUP: TRIGONOMETRIC FUNCTIONS 101
Absolute 101
ArcCosine 102
ArcSine 102
ArcTangent 102
Cosine 102
Exponent 103
Logarithm 103
Round 103

Sine 103
SquareRoot 103
Tangent 104
3.2.5.25 GROUP: SYSTEM VARIABLES 104
Custom 104
GetActiveTool 104
GetActiveX 105
GetActiveY 105
GetActiveZ 105
GetCutterCompD1 – D50 105
GetCutterCompH1 – H50 105
3.2.6 REGISTER Keywords: 106
r_Arc_CCW 106
r_Arc_CW 106
r_Arc_Radius 106
r_Arc_Angle 107
r_Arc_Normal_U 107
r_Arc_Normal_V 107
r_Arc_Normal_W 107
r_Axis_U 107
r_ Axis_V 108
r_ Axis_W 108
r_ChangeOff_4thAxisValue 108
r_ChangeOff_5thAxisValue 108
r_ChangeOff_Func 109
r_ChangeOff_Num 109
r_ChangeOff_ToolLength 109
r_ChangeOff_XValue 109
r_ChangeOff_YValue 109
r_ChangeOff_ZValue 110
r_Coord_Mode 110
r_Cycle_Num 110
r_Cycle_Param 110
r_Diam_Offset 111
r_Drill_Pos_A 111
r_Drill_Pos_B 111
r_Drill_Pos_X 111
r_Drill_Pos_Y 111
r_Drill_Pos_Z 112
r_Dwell_Int_Time 112
r_Dwell_Time 112
r_Expression 112
r_Feed_Rate 113
r_GenCycle_Angle 113
r_GenCycle_Angle_Y 113
r_GenCycle_Depth 113
r_GenCycle_Feed_Finish 113
r_GenCycle_Feed_Rough 114
r_GenCycle_Finishing 114
r_GenCycle_Inc_Move 114
r_GenCycle_N_Holes_X 114

r_GenCycle_N_Holes_Y 115
r_GenCycle_Plane 115
r_GenCycle_Pocket_R 115
r_GenCycle_Pocket_Alt_R 115
r_GenCycle_R_Allow 115
r_GenCycle_Retract 116
r_GenCycle_RetRapid 116
r_GenCycle_Roughing 116
r_GenCycle_Sense_CW 116
r_GenCycle_Sense_CCW 116
r_GenCycle_Slot_Dir 117
r_GenCycle_Start_Angle 117
r_GenCycle_Start_Pos 117
r_GenCycle_Tool_Radius 117
r_GenCycle_X_Allow 117
r_GenCycle_X_Dim 118
r_GenCycle_X_Pos 118
r_GenCycle_Y_Allow 118
r_GenCycle_Y_Dim 118
r_GenCycle_Y_Pos 119
r_GenCycle_Z_Allow 119
r_GenCycle_Z_Init 119
r_GenCycle_Z_Peck 119
r_GenCycle_Z_Pos 119
r_GenCycle_Z_Retract 120
r_Global_Variable_Num 120
r_Global_Variable_String 120
r_Goto_Line 120
r_Groove_Amount 120
r_Groove_Feed 121
r_Groove_IncX 121
r_Groove_IncZ 121
r_Groove_Radius 121
r_Groove_U 121
r_Groove_V 122
r_Groove_X 122
r_Groove_Z 122
r_Hole_Center_X 122
r_Hole_Center_Y 123
r_Hole_Diam_Circ 123
r_Hole_First_X 123
r_Hole_First_Y 123
r_Hole_Int_Ang 123
r_Hole_Num 124
r_Hole_Retract 124
r_Inches 124
r_Int_Ignore 124
r_IntVar_Local_Value 124
r_IntVar_Global_Value 125
r_IntVar_Param_Value 125
r_IntVar_Value 125

r_Last_Pos_X 125
r_Last_Pos_Y 125
r_Last_Pos_Z 126
r_Length_Offset 126
r_Local_Variable_Num 126
r_Local_Variable_String 126
r_Millimeters 126
r_Min_Peck 127
r_Misc_MCH_FileName 127
r_Num_Diam_Offset 127
r_Num_Length_Offset 127
r_Orig_X 127
r_Orig_Y 128
r_Orig_Z 128
r_Orig_4th_axis 128
r_Orig_5th_axis 128
r_Param_Variable_Num 128
r_Param_Variable_String 129
r_Params 129
r_Peck_Dec 129
r_Peck_Inc 129
r_Peck_Retract 129
r_Peck_Repeats 130
r_Peck_Z 130
r_Polar_Angle 130
r_Polar_Center_X 130
r_Polar_Center_Y 131
r_Polar_Center_Z 131
r_Polar_IncAngle 131
r_Polar_Radius 131
r_Pos_4thAxis 131
r_Pos_5thAxis 132
r_Pos_X 132
r_Pos_Y 132
r_Pos_Z 132
r_Pos_I 132
r_Pos_J 133
r_Pos_K 133
r_Pos_U 133
r_Pos_V 133
r_Prog_Name 134
r_Real_Ignore 134
r_RealVar_Param_Value 134
r_RealVar_Global_Value 134
r_RealVar_Local_Value 134
r_RealVar_Value 135
r_Reference_Pos_4th 135
r_Reference_Pos_5th 135
r_Reference_Pos_U 135
r_Reference_Pos_V 135
r_Reference_Pos_W 136

r_Reference_Pos_X 136
r_Reference_Pos_Y 136
r_Reference_Pos_Z 136
r_Retract_Mode 137
r_Rot_Angle 137
r_Rot_Angle_X 137
r_Rot_Angle_Y 137
r_Rot_Angle_Z 137
r_Rot_Center_X 138
r_Rot_Center_Y 138
r_Scal_Factor 138
r_Scale_Factor_X 138
r_Scale_Factor_Y 138
r_Scale_Factor_Z 139
r_Scal_Center_X 139
r_Scal_Center_Y 139
r_Scal_Center_Z 139
r_Spindle_Speed 139
r_Stock_Box 140
r_Stock_Box_Type 140
r_Stock_CenterX 140
r_Stock_CenterY 140
r_Stock_CenterZ 140
r_Stock_Cyl 141
r_Stock_Cyl_Axis 141
r_Stock_Cyl_Type 141
r_Stock_Diameter 141
r_Stock_Length 141
r_Stock_LengthX 142
r_Stock_LengthY 142
r_Stock_LengthZ 142
r_Stock_MinX 142
r_Stock_MinY 142
r_Stock_MinZ 143
r_Stock_Type 143
r_Stock_STL 143
r_Stock_STL_Filename 143
r_Stock_STL_Type 143
r_Stock_STK_Filename 143
r_String_Ignore 144
r_StringVar_Local_Value 144
r_StringVar_Global_Value 144
r_StringVar_Param_Value 144
r_StringVar_Value 144
r_Subprogram_EndLineRet 145
r_SubProgram_EndrsTimes 145
r_SubProgram_GotoLine 145
r_SubProgram_FirstLine 145
r_SubProgram_LastLine 145
r_SubProgram_Number 146
r_SubProgram_Start 146

r_SubProgram_StrStart 146
r_SubProgram_String 146
r_SubProgram_Times 146
r_Tapper_Radius 147
r_Thread_Angle 147
r_Thread_Allowance 147
r_Thread_Feed 147
r_Thread_First_Depth 148
r_Thread_MinLen 148
r_Thread_NTimes 148
r_Thread_Pos_X 148
r_Thread_Pos_U 149
r_Thread_Pos_V 149
r_Thread_Pos_Z 149
r_Thread_Total_Depth 149
r_Thread_Taper_Inc 149
r_Tool_Alpha 150
r_Tool_Angle 150
r_Tool_Ang_Orient 150
r_Tool_Angle_Top 150
r_Tool_Aux 150
r_Tool_Ax_Crad_Offset 151
r_Tool_BC_Crad_Offset 151
r_Tool_Corner_Rad 151
r_Tool_Curr_Alpha 151
r_Tool_Curr_Aux 152
r_Tool_Curr_Corner_R 152
r_Tool_Curr_Diam 152
r_Tool_Curr_Height 152
r_Tool_Cut_Dir 152
r_Tool_Diameter 153
r_Tool_Height 153
r_Tool_Holder_Angle 153
r_Tool_Holder_D 153
r_Tool_Holder_Tip 154
r_Tool_Holder_Type 154
r_Tool_Holder_H 154
r_Tool_Home_Num 154
r_Tool_Ins_Circle 154
r_Tool_Name 155
r_Tool_Number 155
r_Tool_NTG_Filename 155
r_Tool_Orient 155
r_Tool_Orient_X 155
r_Tool_Orient_Y 156
r_Tool_Orient_Z 156
r_Tool_Ref 156
r_Tool_Shank_D 156
r_Tool_Shank_H 156
r_Tool_TLB_Filename 157
r_Tool_Turret_Number 157

r_Tool_Turret_Pos ... 157
r_Tool_Type ... 157
r_Tool_Width ... 158
r_Tool_X_Prog_Point ... 158
r_Tool_Z_Prog_Point ... 158
r_Turn_Cycle_Depth ... 158
r_Turn_Cycle_Excess ... 159
r_Turn_Cycle_End_Label ... 159
r_Turn_Cycle_First_Block ... 159
r_Turn_Cycle_Last_Block ... 159
r_Turn_Cycle_Ret_Ang ... 159
r_Turn_Cycle_Retract ... 160
r_Turn_Cycle_Start_Label ... 160
r_Turn_Cycle_X_Allow ... 160
r_Turn_Cycle_Z_Allow ... 160
r_Units ... 160
r_Var_Access_Mode ... 161
r_Variable_Num ... 161
r_Variable_String ... 161
3.2.7 FORMAT keywords ... 161
concat ... 161
data_separator ... 161
code_num ... 162
code_and_subcode ... 162
subcode_num ... 162
separator ... 162
label ... 162
3.2.8 FUNCTION PARAMETERS Keywords: ... 162
peck ... 162
bottomstop ... 162
bottomCW ... 162
oriented_bottomCW ... 163
manualfeed ... 163
retractfeed ... 163
dwell ... 163
3.5 Parameters: ... 163
AB_AXIS_SENSE_MODAL ... 163
A_AXIS_DEFAULT_SENSE ... 163
ADVANCED_LINE_NUMBERS ... 164
ARC_CHECK ... 164
ARC_MODALITY ... 164
ARC_TOL_INCH ... 164
ARC_TOL_MM ... 165
ARCCENTER_ABSOLUTE ... 165
AUTO_SPINDLE_COMP ... 165
AXIS_MODALITY ... 165
BOLT_PATTERN_MODE ... 166
BOLT_PATTERN_STYLE ... 166
B_AXIS_DEFAULT_SENSE ... 166
CANNED_CYCLES_MODE ... 166
CHECK_COOLANT ... 166

CHECK_DIAM_OFFSET 167
CHECK_HELICAL_MOVE 167
CHECK_LENGTH_OFFSET 167
CHECK_SPINDLE 167
CHECK_SPINDLE_SPEED 167
CHECK_SPIRAL_MOVE 168
CHECK_TOOL_LOAD 168
CHECK_TRAVELS 168
CHECK_WORK_OFFSET 168
CIRCULAR_PARAMS 169
CONTOUR_SUPPORT 169
CONTROL_TYPE 169
CUTTERCOMP_CONNECT 169
CUTTERCOMP_ENTRY_MODE 170
CUTTERCOMP_ERR_LEVEL 170
CUTTERCOMP_MODE 170
CUTTERCOMP_STYLE 170
CUTTERCOMP_TOLERANCE 171
CUTTERCOMP_USE_OFFSET 171
CYCLE_DEPTH_ALWAYS_NEGATIVE 171
CYCLE_FEED_RATE 171
CYCLE_HIRETRACT 172
CYCLE_INC_DEPTH_FROM_R_PLANE 172
CYCLE_PARAM_STYLE 172
CYCLE_RETURN 173
CYCLE_RETURN_FEED 173
CYCLE_RETRACT 173
DECIMAL_POINT 173
DECIMAL_SEPARATOR 173
DRILL_CYCLE_RAPID_AMOUNT 174
DWELL_UNITS 174
ENABLE_WARNINGS 174
ERROR_MODE 174
EXPRESSION_CLOSE 175
EXPRESSION_OPEN 175
FIRST_CODES 175
FEED_RATE_UNITS 175
FOURTH_AXIS_AS_CIRCULAR 175
IGNORE_UNTIL 176
IJK_MODALITY 176
INVERSE_TIME_FEED 176
JUMP_MODE 176
LABEL_REP 177
LABEL_FEED_FACTOR 177
LATHE_FEED_MODE 177
LATHE_GROOVE_INCX 177
LATHE_GROOVE_INCZ 177
LATHE_GROOVE_INCREMENTS 177
LATHE_INSERT_MOVES 178
LATHE_PROG_MODE 178
LCYCLE_FACTOR 178

LOADTOOL_FROM_SPINDLE 178
LOFFSET_AXIS 179
LOFFSET_SIGN 179
LOOKAHEAD_BLOCKS 179
MACRO_ARG_LIST 179
MACRO_ARG_MAX 180
MACRO_ARG_TYPE 180
MACRO_MAX 180
MACRO_MAX_PARAM 180
MACRO_MODAL_STYLE 181
MACRO_SUPPORT 181
MANUAL_TOOL_LENGTH 181
MAX_LOOPS 181
MCH_FILE 182
NC_IGNORE_CASE 182
NC_FORMAT 182
NEED_SETSPINDLE 183
NULL_VALUES 183
OFFSET_INDEX_START 183
PARAMSBEFORE_CANNEDCYCLE 183
POCKET_CYCLE_MODE 183
POLAR_MOVE_MODE 184
RAPID_CANCELCYCLE 184
RAPID_MOVE_MODE 184
REFERENCE_MOVE 184
REFERENCE_POINT 185
REFERENCE_POS_X 185
REFERENCE_POS_Z 185
ROTATION_MOVE_MODE 185
ROTARY_STYLE 185
RVP_FORMAT 186
SECONDARY_DECIMAL_SEPARATOR 186
SPINDLE_SPEED_UNITS 187
STOCK_COMMENTS 187
STRING_LABELS 187
SUBPROG_AND_LINENUM 187
SUBPROG_CALL 187
SUBPROG_DEF_CALL_CODE 188
SUBPROG_DEFEXTENSION 188
SUBPROG_EXTERNAL 188
SUBPROG_MAX 189
SUBPROG_MAX_REP 189
SUBPROG_NEST 189
SUBPROG_NUMBER_DIGITS 189
SUBPROG_NTIMES_DIGITS 190
SUBPROG_POS 190
SUBPROG_PROGNUMBER_POS 190
SUBPROGEX_CALL 190
SUBPROGEX_POS 191
SUBPROGEX_NUMBER_DIGITS 191
SUBPROGEX_NTIMES_DIGITS 191

SUBPROGEX_PROGNUMBER_POS 192
THREAD_CUTTING_FACTOR 192
TOOL_COMMENTS 192
TRAVEL_AS_ERROR 192
UNIT_CHANGE 193
USE_DEFAULT_TOOL 193
VARIABLE_FACTOR 193
VARIABLE_MAX 193
VARIABLE_SUPPORT 193
WARNING_AS_ERROR 194
WARNING_CUTCOMP_DIAMTOOLARGE 194
WORK_OFFSET 194
WRITE_COMMENTS 194
3.4. Error messages: 195
3.4.1 Fatal Errors: 195
BADCOMMAND: 195
BADFORMAT: 195
BADMODIF: 195
BADCYCLE: 195
BADCOMMANDINGROUP: 195
BADDELIMITER: 195
BADMAINGROUP: 195
BEGINBRACKETNOTFOUND: 195
Default 195
ENDBRACKETNOTFOUND: 196
GCF_FERROR: 196
GCF_TOOMANYINCLUDES: 196
INTERNALTABLEOVERFLOW: 196
NOENDQUOTES: 196
NOBEGINQUOTES: 196
NOENDPARENTHESIS: 196
3.4.2 Non Fatal Errors: 196
GCF_EOF: 196
PASTENDMARK: 196
4. RP* File Example 197
1.4 Error checking 204
1.4.1 RP* Configuration Load and Syntax Errors 204
1.4.1.1 RP* Errors 204
1.4.2 NC and CNC Program Syntax and Semantic checking 206
1.4.2.1 NC and CNC Program Error messages 207
Syntactic error codes: 207
Semantic error codes: 208
Long Index: 217