



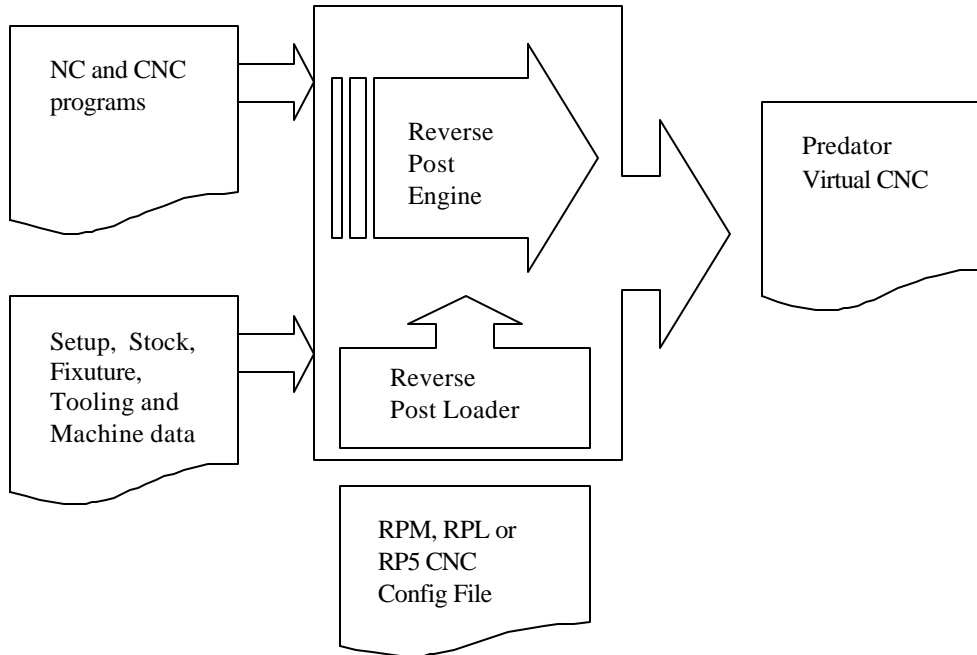
# Predator Virtual CNC Reverse Post Reference Manual

<b>Short Index:</b> .....	<b>1</b>
<b>1. Reverse Post Processing</b> .....	<b>2</b>
1.1 Process and Data Flow Structure .....	2
1.2 Features Supported:.....	2
1.3 The Configuration file: RPM, RP5, RPL & RPT: .....	4
<b>2. Reverse Post File Contents</b> .....	<b>4</b>
<b>3. RP* File format</b> .....	<b>5</b>
3.1 Section Identifiers: .....	5
3.2 Patterns – Section Details .....	8
3.3 Parameters: .....	138
3.4 Error messages:.....	158
<b>4. RP* File Example</b> .....	<b>160</b>
1.4 Error checking.....	167
<b>5. GCF File Updates Record</b> .....	<b>180</b>
5.1 January 29 1999 .....	180
5.2 February 2 1999.....	180
5.3 February 10 1999.....	180
5.3 June 11 1999.....	180
<b>Long Index:</b> .....	<b>181</b>

Created:	January 27 <sup>th</sup> 1999
1 <sup>st</sup> Rev:	February 2 <sup>nd</sup> 1999
2 <sup>nd</sup> Rev:	February 10 <sup>th</sup> 1999
3 <sup>rd</sup> Rev:	June 11 <sup>th</sup> 1999
4 <sup>th</sup> Rev:	June 29 <sup>th</sup> 1999
5 <sup>th</sup> Rev:	October 14 <sup>th</sup> 1999
6 <sup>th</sup> Rev:	November 30 <sup>th</sup> 1999
7 <sup>th</sup> Rev:	July 6 <sup>th</sup> 2000
8 <sup>th</sup> Rev:	November 27 <sup>th</sup> 2001

## 1. Reverse Post Processing

### 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	DONE
Cutter Compensation	DONE
Tool Length Compensation	DONE
Absolute vs. Incremental Programming	DONE
Measurement Unit changes	DONE
4 and 5 Axis Interpolation	DONE
Spindle and Coolant Settings	DONE

## Predator Virtual CNC – Appendix E

Work Coordinates	DONE
G10 and G54.1 Style Work Coordinates	DONE
Lathe Threading Cycles	DONE
Reference return 1 <sup>st</sup> 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
Subroutines	DONE
Sub Programs	DONE
External Sub Programs	DONE
Macro Calls	DONE
Custom Macros	DONE
Lathe Canned Cycles:	
Threading	DONE
Fixed	DONE
Finish	DONE
IF, GOTO, DO, WHILE and End Conditional Branching	DONE

### 1.2.2 Check Areas:

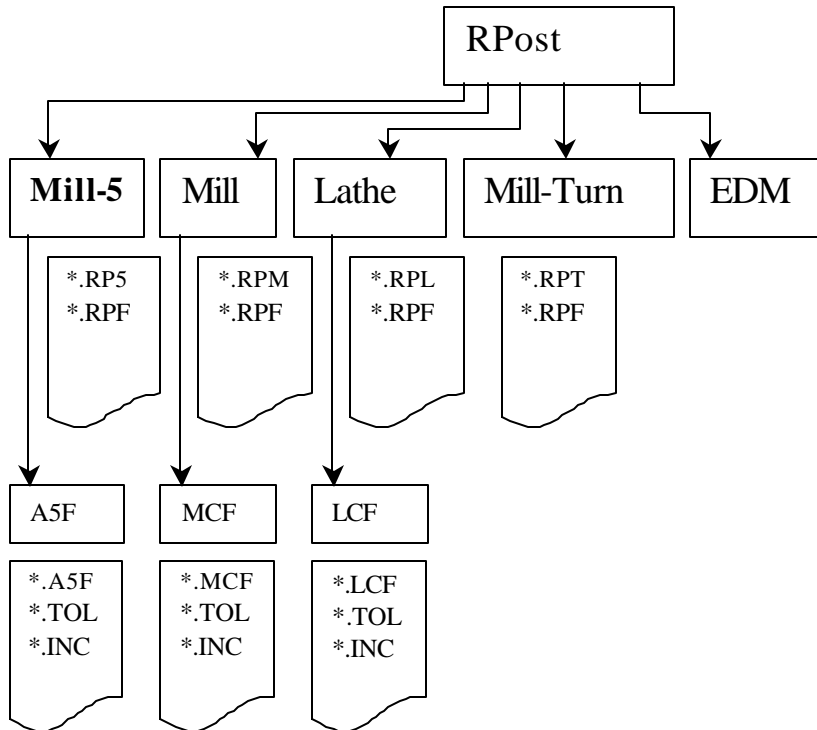
Check Spiral Moves	DONE
Check Helical Moves	DONE
Check Tool Loaded	DONE
Check Feed Defined	DONE
Check Spindle Off	DONE
Check Coolant	DONE
Check Cutter Comp Off in G2/G3	ONGOING
Check Cutter Comp Cancel While Ref Point	ONGOING
Check Cutter Comp G41/G42 without G40	ONGOING
Check Length Comp G43/G44 without G49	DONE
Polar Coordinates	DONE
Scaling	DONE
Rotation	DONE
Mill Canned Cycles:	
Square and Circular Pockets	DONE
Slot Pockets	DONE
Linear Pattern Cycles	DONE
Circular Pattern Cycles	DONE

### 1.2.3 Features not supported:

Exact Stop vs. radius	NOT DONE
2 <sup>nd</sup> Ref Points	NOT DONE
Mill Thread Cutting	NOT DONE
Mirroring	NOT DONE
Local Coordinates	NOT DONE
Functions	NOT DONE
Mill Canned Cycles:	
Threading	NOT DONE

### 1.3 The Configuration file: RPM, RP5, RPL & RPT:

The configuration file is an ASCII file that configures a particular NC control. The RP\* file may be found in one of the subdirectories of the “RPost” directory:



NOTE: The A5F, MCF and LCF directories belong to previous versions of the reverse post and may be ignored.

## 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:

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

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

```

$DATABASEPARATOR:#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 ; New
$STOCKDEF:#1 ; New
$LATHECYCLES:#1 ; New
$PROGRAMMING:#1 ; New
$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.

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
    
```

<b>MoveY</b>	; Maps the linear vertical or Y axis of motion
<b>MoveZ</b>	; Maps the linear depth or Z axis of motion
<b>MoveI</b>	; Maps the circle center horizontal or X axis of motion
<b>MoveJ</b>	; Maps the circle center horizontal or Y axis of motion
<b>MoveK</b>	; Maps the circle center depth or Z axis of motion
<b>MoveR</b>	; Maps the circle radius motion
<b>MoveA</b>	; Maps the rotary horizontal or A axis of motion
<b>MoveB</b>	; Maps the rotary vertical or B axis of motion
<b>Move4thAxis</b>	; Maps the rotary horizontal or A axis of motion
<b>Move5thAxis</b>	; Maps the rotary vertical or B axis of motion
<b>MoveXYZUVW</b>	; Maps the point and tool orientation linear motion
<b>MoveIJKUVW</b>	; Maps the center point and tool orientation circular ; motion
<b>MoveFromXYZ</b>	; Maps the general point and tool orientation linear motion
<b>MoveXYZIJK</b>	; Maps the general point and tool orientation linear motion
<b>MoveU</b>	; Maps the linear horizontal or X axis of incremental motion ; for turning
<b>MoveV</b>	; Maps the linear horizontal or Z axis of incremental motion ; for turning

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:

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

<b>CreateTool</b>	; Maps tool comments for tool size and shape definitions
<b>LoadTool</b>	; Maps loading the tool from the tool changer
<b>SelectTool</b>	; Maps selecting the tool from the tool carousel
<b>SelectAndLoadTool</b>	; Maps selecting and loading the tool from the tool changer
<b>CreateMillTool</b>	; Maps tool comments for tool size and shape definitions
<b>CreateLatheTool</b>	; Maps tool comments for tool size and shape definitions
<b>SetTLBFileName*</b>	; Maps specifying TLB filename
<b>LoadTLBFileName*</b>	; 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:

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

<b>AirBlast</b>	; Maps enabling/disabling air blast
<b>ChangePallet</b>	; Maps changing pallet
<b>CoolantON</b>	; Maps coolant on
<b>CoolantOFF</b>	; Maps coolant off
<b>CtantSurfSpeed</b>	; Maps enabling constant surface speed
<b>CtantSurfSpeedCancel</b>	; Maps cancelling constant surface speed
<b>HspindleStop</b>	; Maps horizontal spindle stop
<b>LineNumber</b>	; Maps line number control
<b>MaxSpindleSpeed</b>	; Maps maximum spindle speed
<b>OptStop</b>	; Maps optional stop
<b>Pause0</b>	; Maps program stop
<b>Pause1</b>	; Maps optional stop
<b>ProgramEnd</b>	; Maps end of program
<b>SetMCHFileName</b>	; Maps setting machine file name
<b>SetOrigRelocation</b>	; Maps setting origin relocation
<b>SetMachineDatum</b>	; Maps setting machine datum
<b>SpindleCW</b>	; Maps clockwise spindle motion
<b>SpindleCCW</b>	; Maps counter clockwise spindle motion
<b>SpindleStop</b>	; Maps spindle stop
<b>SpindleSpeed</b>	; Maps spindle speed
<b>Stop</b>	; Maps program stop
<b>ToolClamp</b>	; Maps tool clamp/unclamp
<b>VspindleStop</b>	; 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:

<b>FeedPerMinute</b>	; Maps feed rate per minute
<b>FeedPerRevolution</b>	; Maps feed rate per revolution
<b>InvTimeFeed</b>	; Maps inverse time feed
<b>OutFeed</b>	; Maps feed rate
<b>SetFeed</b>	; Maps feed rate
<b>SetMaxFeed</b>	; 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: Unsupported's 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: Ignore's 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:

<b>AxisRotation</b>	; Maps setting axis rotation
<b>CancelAxisRotation</b>	; Maps canceling axis rotation
<b>ChangeAllWorkCoord</b>	; Maps changing all work coordinates
<b>ChangeOffset</b>	; Maps work or fixture offset's X, Y and Z values
<b>DisableWorkCoord</b>	; Maps disabling work coordinates
<b>SetCoordSystem</b>	; Maps setting system coordinates
<b>SetWorkCoord</b>	; Maps up to 100 work or fixture offsets
<b>SetWorkCoord1</b>	; Maps first work or fixture offset
<b>SetWorkCoord2</b>	; Maps second work or fixture offset
<b>SetWorkCoord3</b>	; Maps third work or fixture offset
<b>SetWorkCoord4</b>	; Maps fourth work or fixture offset
<b>SetWorkCoord5</b>	; Maps fifth work or fixture offset
<b>SetWorkCoord6</b>	; Maps sixth work or fixture offset

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:

<b>SetAbsoluteZero</b>	; 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:

<b>ThreadCycle</b>	; Maps the thread cutting cycle
<b>ThreadMove</b>	; Maps the thread cutting move

NOTE: Threading's 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:

<b>ProgramStart</b>	; Maps the start of the main program
<b>SubProgramStart</b>	; Maps the start of each subprograms
<b>SubProgramEnd</b>	; Maps the end of each subprogram
<b>SubProgramCall</b>	; Maps calling a subprogram
<b>SubProgramRet</b>	; Maps returning from a subprogram
<b>SubProgramRetEnd</b>	

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:

<b>DefineGlobalVar</b>	; Maps defining a global variable
<b>DefineLocalVar</b>	; Maps defining a local variable
<b>DefineParameter</b>	; Maps defining a parameter
<b>GetActiveTool</b>	; Maps getting active tool
<b>GetActiveDCode</b>	; Maps getting active D offset value
<b>GetCurrentVar</b>	; Maps getting current variable
<b>GetCurrentReg</b>	; Maps getting current register
<b>GetGlobalVar</b>	; Maps getting a global variable
<b>GetLocalVar</b>	; Maps getting a local variable
<b>GetParameter</b>	; Maps getting a parameter
<b>GetVariable</b>	; Maps getting a variable
<b>SetGlobalVar</b>	; Maps setting a global variable
<b>SetLocalVar</b>	; Maps setting a local variable
<b>SetParameter</b>	; Maps setting a parameter
<b>SetVariable</b>	; 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 – Reserved for Backwards Compatibility

Although no longer supported it does still work. The donothing 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:

<b>Ignore</b>	; Maps CNC codes that should be ignored
---------------	---

NOTE: Ignore's 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:

<b>CommentFollows</b>	; Maps the remainder of the line as a comment
<b>CommentStarts</b>	; Maps the start of a comment
<b>CommentEnds</b>	; 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:

<b>CreateStockBox</b>	; Maps stock box creation
<b>CreateStockHole</b>	; Maps stock box hole creation
<b>CreateStockFixt</b>	; Maps fixture box creation
<b>CreateStockCyl</b>	; Maps stock cylinder creation
<b>CreateStockCylHole</b>	; Maps stock cylinder hole creation
<b>CreateStockCylFixt</b>	; Maps fixture cylinder creation
<b>CreateStockSTL</b>	; Maps STL stock creation
<b>CreateStockSTLFixt</b>	; Maps STL fixture creation
<b>SetSTKFileName*</b>	; 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:

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

<b>Condition</b>	; Maps conditional statements
<b>StartLoop</b>	; Maps the beginning of loops
<b>EndLoop</b>	; Maps the end of loops
<b>Goto</b>	; Maps goto style jump statements
<b>SetLabel</b>	; 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 exponentation are supported. All arithmetic function keywords are listed below with a short description:

<b>Add</b>	; Maps additions
<b>Subtract</b>	; Maps subtractions
<b>Multiply</b>	; Maps multiplications
<b>Divide</b>	; Maps divisions
<b>Module</b>	; Maps modulus
<b>Exponent</b>	; Maps exponentations

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:

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

<b>SINE</b>	; Maps sines
<b>COSINE</b>	; Maps cosines
<b>TANGENT</b>	; Maps tangents
<b>ARCTANGENT</b>	; Maps arc tangents
<b>SQUAREROOT</b>	; Maps square roots
<b>ABSOLUTE</b>	; Maps absolutes
<b>ROUND</b>	; Maps rounds
<b>ARCCOSINE</b>	; Maps arc cosines
<b>LOGARITHM</b>	; Maps logarithms
<b>EXPONENT</b>	; 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,delimiters	: 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:

<b>Casesens</b>	Allows the pattern to be upper, lower or both cases
<b>Concat</b>	Concatenate multiple lines together for processing
<b>Expression</b>	Math expression follows the main pattern
<b>Group</b>	Group follows the main pattern – Reserved
<b>Int</b>	Integer number follows the main pattern
<b>Ignore_case</b>	Allows the pattern to be any case
<b>Multi</b>	One or more secondary patterns follows the main pattern
<b>Null</b>	Nullifies any data following the main pattern
<b>Real</b>	Real number follows the main pattern
<b>Real2</b>	2 Place factored real number follows the main pattern
<b>Real3</b>	3 Place factored real number follows the main pattern
<b>Real4</b>	4 Place factored real number follows the main pattern
<b>String</b>	Alphanumeric string follows the main pattern
<b>Switch</b>	One of many secondary patterns follows the main pattern
<b>Void</b>	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:

<b>Int_reg</b>	Integer number follows the main pattern
<b>Real_reg</b>	Real number follows the main pattern
<b>Real_reg2</b>	2 Place factored real number follows the main pattern
<b>Real_reg3</b>	3 Place factored real number follows the main pattern
<b>Real_reg4</b>	4 Place factored real number follows the main pattern
<b>String_reg</b>	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]

### 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 is 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:

<b>Delimiters</b>	Requires the pattern to use the specified delimiters
<b>Multigstricord</b>	Requires the pattern to be in a specific order
<b>Multigstriclen</b>	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 seperators 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

$DATABASEPARATOR:#0
"\32"             : databaseparator
"\0"              : databaseparator
"\9"              : databaseparator

$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,multigstrictord : 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:

<b>AbsoluteCoord</b>	; Maps absolute coordinate programming
<b>CircularMoveSense</b>	; Maps the cw or ccw direction of circular interpolation
<b>CircleCenter</b>	; Maps circle center definition for circular interpolation
<b>CancelPolarCoord</b>	; Maps cancel modal polar style coordinate programming
<b>CancelRotation</b>	; Maps cancel modal rotation
<b>CancelScaling</b>	; Maps cancel modal scaling
<b>DefUnits</b>	; Maps either inch or metric coordinate programming
<b>Dwell</b>	; Maps dwells
<b>IncrementalCoord</b>	; Maps incremental coordinate programming
<b>Inches</b>	; Maps inch or imperial coordinate programming
<b>Millimeters</b>	; Maps metric coordinate programming
<b>MoveCircular</b>	; Maps circular interpolation
<b>MoveCircularCW</b>	; Maps clockwise circular interpolation
<b>MoveCircularCCW</b>	; Maps counter clockwise circular interpolation
<b>MoveLinear</b>	; Maps linear feed interpolation
<b>MoveRapid</b>	; Maps linear rapid interpolation
<b>Multax</b>	; Maps 4 and 5 axis programming
<b>PolarCoord</b>	; Maps modal polar style coordinate programming
<b>PolarMoveLinear</b>	; Maps polar definition for linear interpolation
<b>PolarMoveCircular</b>	; Maps polar definition for circular interpolation
<b>Rotation</b>	; Maps modal rotation
<b>Scaling</b>	; Maps modal scaling
<b>SelectPlaneXY</b>	; Maps XY plane selection for circular interpolation
<b>SelectPlaneYZ</b>	; Maps YZ plane selection for circular interpolation
<b>SelectPlaneZX</b>	; Maps ZX plane selection for circular interpolation
<b>SetABSenseCW</b>	; Maps the cw direction of rotary axis interpolation
<b>SetABSenseCCW</b>	; Maps the ccw direction of rotary axis interpolation
<b>SelectPlane</b>	; 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  
“CYCL\*DEF\*26.1\*X1\*Y1\*CCX+0\*CCY+0” : CancelScaling ; Heidenhain 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 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

### **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  
"CYCL\*DEF\*10.0",multi@24 : Rotation ; Heidenhain style  
"CYCL\*DEF\*10.1",multi@24 : Rotation ; Heidenhain 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  
"CYCL\*DEF\*26.0",multi@25   : Scaling               ; Heidenhain style  
"CYCL\*DEF\*26.1",multi@25   : Scaling               ; Heidenhain style

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 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  
**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  
**MoveY** ; Maps the linear vertical or Y axis of motion  
**MoveZ** ; Maps the linear 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

### 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 motion  
Tech Tips: Selects the lathe insert motion. 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,multigstrictord : 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,multigstrictord : 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

### MoveJKUVW

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 : MoveJKUVW ; 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,multigstrictord : 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,multigstrictord : 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,multigstrictord : 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          ; Standard style or
"R",expression,multigstrictord : MoveR          ; Variables with math functions
"L",real                : MoveR          ; Okuma 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,multigstrictord : 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,multigstrictord : 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,multigstrictord : 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

## 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  
**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  
**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

### LoadTool

Description: Maps loading the tool from the tool changer  
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

### 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

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:

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

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:

"L9801",multi@17 : CircPocketCycle () ;Fadal style  
"L9901",multi@17 : CircPocketCycle ()

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

NOTE: \_CircPocketCycle is also supported for backwards compatibility.

### **CircPocketCycleCW**

Description: Maps clockwise circular 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 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.

### DefCycle

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

```
"CYCL*DEF*1.0",multi@9 : DefCycle(peck)  
"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()
```

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

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

NOTE: \_LineCycle is also supported for backwards compatibility.

### PatternCycle

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

"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.

### RectPocketCycle

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

"L9601",multi@16 : RectPocketCycle () ;Fadal style  
 "L9701",multi@16 : RectPocketCycle ()

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:

<b>AirBlast</b>	; Maps enabling/disabling air blasts
<b>ChangePallet</b>	; Maps changing pallets
<b>CoolantON</b>	; Maps coolant on
<b>CoolantOFF</b>	; Maps coolant off
<b>CtantSurfSpeed</b>	; Maps constant surface speed
<b>CtantSurfSpeedCancel</b>	; Maps cancelling constant surface speed
<b>HSpindleStop</b>	; Maps horizontal spindle stop
<b>LineNumber</b>	; Maps line or sequence numbers
<b>MaxSpindleSpeed</b>	; Maps maximum spindle speed
<b>OptStop</b>	; Maps optional stop
<b>Pause0</b>	; Maps program stop
<b>Pause1</b>	; Maps optional stop
<b>ProgramEnd</b>	; Maps end of program
<b>SetMachineDatum</b>	; Maps setting machine datum
<b>SetMCHFileName</b>	; Maps setting machine file name
<b>SetOrigRelocation</b>	; Maps setting origin relocation
<b>SpindleCW</b>	; Maps clockwise spindle motion
<b>SpindleCCW</b>	; Maps counter clockwise spindle motion
<b>SpindleSpeed</b>	; Maps spindle speed
<b>SpindleStop</b>	; Maps spindle stop
<b>Stop</b>	; Maps program stop
<b>ToolClamp</b>	; Maps tool clamp/unclamp
<b>VSpindleStop</b>	; Maps vertical spindle stop

#### **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

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

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

### **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

### **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

### 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

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

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. 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

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  
 "M86" : ToolClamp ; Haas style

NOTE: The time for an 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

### 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

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

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

Function code: 0  
 Group code: 100800  
 RVP codes

### 3.2.5.8 GROUP: REFERENCE POINT RETURN

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

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

#### ReturnToRefP

Description: Maps return to a reference point  
 Tech Tips: Returns to a reference point. On most CNC controls this is a G28 code. An example using this function keyword includes the following:

"G28" : ReturnToRefP ; Fanuc style

Function code: 1  
 Group code: 100600  
 RVP codes

#### ReturnFromRefP

Description: Maps return from a reference point  
 Tech Tips: Returns from a reference point. On most CNC controls this is a G29 code. An example using this function keyword includes the following:

"G29" : 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:

**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

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  
 "C2" : CutterCompRight ; Dynapath style  
 "ToolComp~Right" : CutterCompRight ; Anilam 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 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:

"DIAM\_OFFSET",multi@104 : SetDiamOffset ; Predator 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  
**DisableWorkCoord** ; Maps disabling work coordinates  
**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  
**SetWorkCoord** ; Maps up to 100 work or fixture offsets  
**SetWorkCoordNext** ; Maps setting next work coordinates

**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  
 “CYCL\*DEF\*7.0”,multi@19 : ChangeOffset ; Heidenhain style  
 “CYCL\*DEF\*7.1”,multi@19 : ChangeOffset  
 “CYCL\*DEF\*7.2”,multi@19 : ChangeOffset

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

### DisableWorkCoord

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

"G60" : DisableWorkCoord

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

#### **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

### 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:

**SetAbsoluteZero** ; Maps setting the absolute zero position

#### 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:

"G52"	: 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

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  
 "G33",multi@2 :ThreadMove ; Fanuc 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:

<b>MacroCall</b>	; Maps calling a macro
<b>ProgramStart</b>	; Maps the start of the main program
<b>SubProgramCall</b>	; Maps calling a subprogram
<b>SubProgramEnd</b>	; Maps the end of each subprogram
<b>SubProgramEx</b>	; Maps calling an extended subprogram
<b>SubProgramRet</b>	; Maps returning from a subprogram
<b>SubProgramRetEnd</b>	; Maps returning end point of subroutines and sub programs
<b>SubProgramRetEx</b>	; Maps returning from an extended subprogram
<b>SubProgramStart</b>	; Maps the start of each subprograms

### 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

### 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\100"	: 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
"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	; Hedenhain style
"L",multi@6,multigstrictord	: SubProgramCall	

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
"ENS"	: SubProgramRet	; Vickers 2100 style or
"RTS"	: SubProgramRet	; Okuma style or
"RSB,"	: SubProgramRet	; G&L style or
"LBL*0"	: SubProgramRet	; Hedenhain 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"	: SubProgramStart	; Fanuc style or
","	: 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

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:

<b>DefineGlobalVar</b>	; Maps defining a global variable
<b>DefineLocalVar</b>	; Maps defining a local variable
<b>DefineParameter</b>	; Maps defining a parameter
<b>GetActiveTool</b>	; Maps getting the active tool number
<b>GetActiveDCode</b>	; Maps getting the active diameter compensation value
<b>GetCurrentReg</b>	; Maps getting current register
<b>GetCurrentVar</b>	; Maps getting current variable
<b>GetGlobalVar</b>	; Maps getting a global variable
<b>GetLocalVar</b>	; Maps getting a local variable
<b>GetParameter</b>	; Maps getting a parameter
<b>GetVariable</b>	; Maps getting a variable
<b>SetGlobalVar</b>	; Maps setting a global variable
<b>SetLocalVar</b>	; Maps setting a local variable
<b>SetParameter</b>	; Maps setting a parameter
<b>SetVariable</b>	; 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:

```
"#4001" : GetActiveTool ; Fanuc 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:

```
"#4101" : GetActiveDCode ; Fanuc style or
"ADC" : GetActiveDCode ; G&L 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
"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
    
```

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
    
```

Function code: 9  
 Group code: 101800  
 RVP codes:

### GetParameter

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

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,multigstrictord : SetGlobalVar           ; Fadal style or
"=K",expression,multigstrictord : SetGlobalVar           ; Fagor style or
"P01",expression,multigstrictord : 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.
    
```

```

Function code: 4
Group code: 101800
RVP codes:
    
```

### SetVariable

```

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

```

Function code: 11
Group code: 101800
RVP codes
    
```

## 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:

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

```

"G81.1",multi@2          : FaceDrillCycle()
"G83.1",multi@2          : FaceDrillCycle(peck)
    
```

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)  
 "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.

Function code: 10500  
 Group code: 102000  
 Parameters: finish horizontal vertical box\_strategy pre\_finish  
 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:

"G87",multi@19 : TurnGrooveCycle(vertical) ; Fanuc style or  
 "G88",multi@19 : TurnGrooveCycle(vertical)  
 "G89",multi@19 : TurnGrooveCycle(vertical)  
 "G73",multi@18 : TurnGrooveCycle(horizontal) ; Okuma style  
 "G74",multi@18 : TurnGrooveCycle(vertical)

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

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  
 "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

**Condition**

Description: Maps conditional statements  
 Tech Tips: Selects conditional statements. An example on using this function keyword includes the following:

"IF",expression,multigstrictord : Condition ; Fanuc style or  
 "WHILE",expression,multigstrictord : Condition ; Haas style or  
 "WH",expression,multigstrictord : Condition ; G&L style  
 "IFS",expression,multigstrictord : 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:

### 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 exponentation are supported. All arithmetic function keywords are listed below with a short description:

<b>Add</b>	; Maps additions
<b>Divide</b>	; Maps divisions
<b>Exponent</b>	; Maps exponentations
<b>Module</b>	; Maps moduluses
<b>Multiply</b>	; Maps multiplications
<b>Subtract</b>	; 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 exponentations  
 Tech Tips: Selects exponentations of variables and constants. An example on using this function keyword includes the following:

"^",#20 : Exponent() ; Fanuc style

Function code: 6  
 Group code: 102200  
 RVP codes:

**Module**

Description: Maps moduluses  
 Tech Tips: Selects moduluses of variables and constants. An example on using this function keyword includes the following:

"MOD",#20 : Module() ; Fanuc 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:

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

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

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

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

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

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

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:

<b>ABSOLUTE</b>	; Maps absolutes
<b>ARCCOSINE</b>	; Maps arc cosines
<b>ARCTANGENT</b>	; Maps arc tangents
<b>COSINE</b>	; Maps cosines
<b>EXPONENT</b>	; Maps exponents
<b>LOGARITHM</b>	; Maps logarithms
<b>ROUND</b>	; Maps rounds
<b>SINE</b>	; Maps sines
<b>SQUAREROOT</b>	; Maps square roots
<b>TANGENT</b>	; 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

Function code: 23  
 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

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 logarithim 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 rounding functions between variables and constants. An example on using this function keyword includes the following:

"ROUND",#10 : Round(single pre) ; Fanuc 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

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:

### 3.2.6 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 ccounter 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 4<sup>th</sup> axis change offset register  
 Tech Tips: Selects the current 4<sup>th</sup> 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_4thAxisValueaxis_w      ; Fanuc style
```

Function code: 0  
 Group code: 0  
 RVP codes

### **r\_ChangeOff\_5thAxisValue**

Description: Maps the 5<sup>th</sup> axis change offset register  
 Tech Tips: Selects the current 5<sup>th</sup> 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_5thAxisValueaxis_w      ; 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

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

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\_Peek**

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\_peek ; Fagor style or  
 "Q202=",real : r\_gencycle\_z\_peek ; 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 parameter 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\_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\_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\_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\_Peek\_Inc**

Description: Maps the peck drill increment register position  
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\_peek\_inc

Function code: 75  
Group code: 0  
RVP codes

### **r\_Peek\_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\_peek\_retract

Function code: 74  
Group code: 0  
RVP codes

### **r\_Peek\_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\_peek\_repeats ; Haas style

Function code: 241  
Group code: 0  
RVP codes

### **r\_Peek\_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\_peek\_z

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\_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\_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\_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\_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

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\_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:

“N”,string : r\_tool\_name

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 = Taperball, 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

Function code: 0  
Group code: 0  
RVP codes

### **r\_Tool\_Z\_Prog\_Point**

Description: Maps the current tool insert programming point in Z

Tech Tips: Selects the current tool insert programming point in the Z or horizontal axis. With the Predator style tool comments this is an Z. An example of using this register keyword includes the following:

“Z”,real : r\_tool\_z\_prog\_point ; Predator style

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\_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 a 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 ptttern 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.

<b>peck</b>	function(s):	<b>_DrillCycle</b>
	internal format code:	2
	description:	enables pecking
<b>bottomstop</b>	function(s):	<b>_DrillCycle</b>
	internal format code:	4
	description:	manual stop on bottom
<b>bottomCW</b>	function(s):	<b>_DrillCycle</b>
	internal format code:	8
	description:	CW rotation at bottom
<b>oriented_bottomCW</b>	function(s):	<b>_DrillCycle</b>
	internal format code:	8
	description:	CW rotation at bottom
<b>manualfeed</b>	function(s):	<b>_DrillCycle</b>
	internal format code:	8
	description:	manual feed
<b>retractfeed</b>	function(s):	<b>_DrillCycle</b>
	internal format code:	16
	description:	feed retract (default: rapid)
<b>dwll</b>	function(s):	<b>_DrillCycle</b>
	internal format code:	18
	description:	dwll

### 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

### **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\_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 & K values or  
ARCCENTER\_ABSOLUTE=OFF ; Incremental I,J & K values

Default value: OFF  
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

### **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

### **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

### **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\_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

### **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\_RETURN**

Description: Toggles the depth to return from a cycle  
Tech Tips: The I option enables returning to the initial point. The R option enables returning to the retract point. Examples using this parameter include the following:

CYCLE\_RETURN=I ; Initial depth or  
CYCLE\_RETURN=R ; Retract depth

Default value: I  
Parameter: string

NOTE: Currently this parameter is reserved for future use and has not yet been implemented.

### **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

Default value: 0.01  
Parameter: float

NOTE: Currently this parameter is reserved for future use and has not yet been implemented.

### **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: Toggles use of '.' or ',' as decimal separators  
Tech Tips:

DECIMAL\_SEPARATOR = "."  
DECIMAL\_SEPARATOR = ","

Default value: "."  
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

### **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

FEED\_RATE\_UNITS=METERS  
FEED\_RATE\_UNITS=MILIMETERS

Default value: MILIMETERS  
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

### **LATHE\_FEED\_MODE**

Description: Specifies whether lathe feed rate will be specified for minutes or for revolution  
Tech Tips:

LATHE\_FEED\_MODE=REVOLUTION  
LATHE\_FEED\_MODE=MINUTE

Default value: MINUTE  
Parameter: string

### **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

Default value: 10000  
Parameter: real

### **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 specifies 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

### MACRO\_ARG\_LIST

Description: Sets the argument order for macro calls  
Tech Tips: Selects the order of the letter addresses that are used as arguments to pass values into subroutines and sub programs via a macro call method. Examples using this parameter include the following:

MACRO\_ARG\_TYPE=TYPE2 ; Custom style Macro Calls  
MACRO\_ARG\_MAX=5  
MACRO\_ARG\_LIST=A,1,B,2,C,3,D,4,E,5

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 style macro call. The TYPE2 option allows the assignment of letter addresses to macro call arguments. 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

### 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

### **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  
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 cycle  
Tech Tips: The ON option enables canceling a cycle when rapid motion occurs. The OFF option allows rapids to occur within a 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: Toggle the reference move style  
Tech Tips: The TOOL\_CHANGE option will return the tool to the tool change position during a reference move. The NONE option will just process the reference move and will not force the tool back to the tool change position. Examples using this parameter include the following:

REFERENCE\_MOVE=NONE  
REFERENCE\_MOVE=TOOL\_CHANGE

Default value: NONE  
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\_MOVE=ALWAYS\_ABSOLUTE  
REFERENCE\_MOVE=INC\_AND\_ABSOLUTE

Default value: INC\_AND\_ABSOLUTE  
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

### **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.

### **SPINDLE\_SPEED\_UNITS**

Description: Sets spindle speed units  
Tech Tips

SPINDLE\_SPEED\_UNITS=METERS  
SPINDLE\_SPEED\_UNITS=MILIMETERS

Default value: MILIMETERS  
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\_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. 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
```

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. The

EXTERNAL option allows the sub programs to be located in separate files on the hard disk with the main program stored in it's own file.

SUBPROG\_POS=NONE ; Fanuc 3000 style or  
SUBPROG\_POS=BOTTOM ; Fanuc 0 style  
SUBPROG\_POS=EXTERNAL ; Sub Programs stored in separate files

NOTE: By definition the EXTERNAL option should only be used with sub programs and not with subroutines. Subroutines should always be stored with the main program in the same file.

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. The EXTERNAL option allows the sub programs to be located in separate files on the hard disk with the main program stored in it's own file.

SUBPROGEX\_POS=NONE ; Fanuc 3000 style or

SUBPROGEX\_POS=BOTTOM ; Fanuc 0 style  
SUBPROGEX\_POS=EXTERNAL ; Sub Programs stored in separate files

NOTE: By definition the EXTERNAL option should only be used with sub programs and not with subroutines. Subroutines should always be stored with the main program in the same file.

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

### **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.

STOCK\_COMMENTS=ON ; Recommended for most applications  
STOCK\_COMMENTS=OFF

Default value: OFF  
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 will not convert values and assumes that decimal points are used within the CNC program. 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

### **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.

TOOL\_COMMENTS=ON ; Recommended for most applications  
TOOL\_COMMENTS=OFF

Default value: OFF  
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.

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.

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

VARIABLE\_FACTOR=1.0  
VARIABLE\_FACTOR=100.0

Default value: 1.0  
Parameter: real

### **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.

VARIABLE\_SUPPORT=ON ; Recommended for most applications  
VARIABLE\_SUPPORT=OFF

Default value: OFF  
Parameter: string

### **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.

VARIABLE\_MAX=32 ; Fanuc style  
VARIABLE\_MAX=16

Default value: 64  
Parameter: int

### **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

WARNING\_AS\_ERROR=ON  
WARNING\_AS\_ERROR=OFF

Default value: OFF  
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]
@1sorted:#0
%d,int_reg          : code_num
"."                : code_and_subcode
%d,int_reg          : subcode_num

@3:#0
"G90"               : use_absolute
"G91"               : use_incremental

@4:#0
"G98"               : retract_to_init
"G99"               : retract_to_R

@2:#0
"",switch@4,casesens : r_retract_mode
"\0"                : separator
"",switch@3,casesens : r_coord_mode
"\0"                : separator
"X",real            : r_drill_pos_X
"\0"                : separator
"Y",real            : r_drill_pos_Y
"\0"                : separator
"Z",real            : r_peck_Z
;"Z",real           : r_drill_pos_Z
"\0"                : separator
"Q",real            : r_peck_inc
"\0"                : separator
"R",real            : r_peck_retract
"\0"                : separator
"L",int             : r_peck_repeats
"\0"                : separator
"P",int             : r_dwell_time

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

```

;This must be the first section  
;pattern 1 formats the numbers  
;before separator  
;separator . or ,  
;after separator  
;  
;pattern 3 is for abs/inc  
;only used in canned cycle  
;  
;pattern 4 is for retract planes  
;only used in canned cycle  
;  
;pattern 2 is for canned cycles  
;calls pattern 4 for retract plane  
;only allow G98 or G99  
;(\0=null) allows for no spaces  
;calls pattern 3 for abs/inc  
;only allow G90 or G91  
;  
;horizontal value for drill position  
;vertical value for drill position  
;standard  
;optional drill depth value  
;optionally change "Q" to "Z"  
;optionally change "R" to "Z"  
;  
;reference point return G28/G29

## Predator Virtual CNC – Appendix E

```

"\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

```

## Predator Virtual CNC – Appendix E

```

"\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 workoffsets
"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 workoffsets
"P",int             : r_work_coord
;
;Function Groups start with $
;allows for no spaces in file
;null character
$DATASEPARATOR: #0
\0,null             : data_separator
;=",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

```

## Predator Virtual CNC – Appendix E

```

"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

```

## Predator Virtual CNC – Appendix E

```

"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

```

## Predator Virtual CNC – Appendix E

```

"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
                                                                    ;
offset_axis=Z           ; X Y Z or PROG length offset
; along axis
offset_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
;
;valid type1 or type2
;macro_arg_type=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

## Predator Virtual CNC – Appendix E

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: 0x0000001L

##### 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: 0x0000002L

##### 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: 0x0000004L

##### 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: 0x0000008L

##### 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: 0x0000010L

##### 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: 0x0000020L

##### SYNTAX\_REVERSE POST\_ERR\_UNKNOWNKEYW

Description: Keyword unknown  
Error Code: 0x0000040L

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<sup>th</sup> 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<sup>th</sup> 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: 0x8000000L

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: 0x0000001L

#### 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: 0x0000002L

#### 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: 0x0000004L

#### 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: 0x0000008L

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.

## Predator Virtual CNC – Appendix E

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

## **5. GCF File. Updates Record**

### **5.1 January 29 1999**

- First draft

### **5.2 February 2 1999**

- Keywords renamed
- Documentation completed

### **5.3 February 10 1999**

- Include reverse post structure and list of supported features
- New directory structure
- Documentation revised and updated

### **5.3 June 11 1999**

- NC Error messages updated
- New functionality for existing groups
- New supported features: Subprograms & Variables
- Pattern redirector revised and updated
- Parameters revised and updated
- RP\* example revised and updated

## Long Index:

<b>1. Reverse Post Processing</b> .....	<b>2</b>
<b>1.1 Process and Data Flow Structure</b> .....	<b>2</b>
<b>1.2 Features Supported:</b> .....	<b>2</b>
<b>1.2.1 Features already supported:</b> .....	<b>2</b>
<b>1.2.2 Check Areas:</b> .....	<b>3</b>
<b>1.2.3 Features not supported:</b> .....	<b>3</b>
<b>1.3 The Configuration file: RPM, RP5, RPL &amp; RPT:</b> .....	<b>4</b>
<b>2. Reverse Post File Contents</b> .....	<b>4</b>
<b>3. RP* Files</b> .....	<b>5</b>
<b>3.1 Section identifiers:</b> .....	<b>5</b>
[Include].....	5
[Initializations] .....	6
[Patterns].....	6
[Exclusive].....	6
[Parameters].....	7
[End].....	8
<b>3.2 Patterns – Section Details</b> .....	<b>8</b>
Multiple Pattern Definition.....	8
Function Groups: .....	9
Pattern:.....	9
<b>3.2.1 Function Groups</b> .....	<b>10</b>
MOVETO .....	10
MOVETYPE .....	11
TOOLCHANGE .....	12
CANNEDCYCLES .....	12
MISCELLANEOUS .....	13
FEEDRATE.....	13
UNSUPPORTED.....	13
REFPOINTRETURN.....	14
CUTTERCOMP .....	14
IGNORE.....	14
WORKCOORD .....	15
ABSOLUTEZERO.....	15
THREADING .....	15
SUBPROGRAMS .....	15
VARIABLES .....	16
DONOTHING – Reserved for Backwards Compatibility .....	16
COMMENTS .....	16
OPERATORS – Reserved for Backwards Compatibility .....	17
FUNCTIONS – Reserved for Backwards Compatibility .....	17
<b>LATHECYCLES</b> .....	<b>17</b>
<b>PROGRAMMING</b> .....	<b>18</b>
<b>ARITHMETIC FUNCTIONS</b> .....	<b>18</b>
NOTE: The * are for ATL and NC file scanning support.....	19
<b>3.2.2 Pattern format</b> .....	<b>19</b>
<b>3.2.3 FORMAT Keywords</b> .....	<b>19</b>
Casesens .....	20

Concat.....	20
Expression.....	21
Group – Reserved for future use.....	21
Int.....	21
Int_reg.....	21
Multi@#.....	22
Null.....	22
Real.....	23
Real2.....	23
Real3.....	23
Real4.....	23
Real_reg.....	24
Real_reg2.....	24
Real_reg3.....	24
Real_reg4.....	24
String.....	24
String_reg.....	25
Switch@#.....	25
Void.....	25
3.2.4 Multiple Pattern Secondary Format Keywords.....	26
Delimiters.....	26
Multigstrictlen.....	27
Multigstrictord.....	27
<b>3.2.5 FUNCTION keywords:.....</b>	<b>27</b>
3.2.5.1 MOVETYPE Function Group.....	28
AbsoluteCoord.....	28
CircularMoveSense.....	28
CircleCenter.....	29
CancelPolarCoord.....	29
CancelRotation.....	29
CancelScaling.....	29
DefUnits.....	30
Dwell.....	30
IncrementalCoord.....	30
Inches.....	30
Millimeters.....	31
MoveCircular.....	31
MoveCircularCW.....	31
MoveCircularCCW.....	31
MoveLinear.....	32
MoveRapid.....	32
Multax.....	32
PolarCoord.....	33
PolarMoveLinear.....	33
PolarMoveCircular.....	33
Rotation.....	33
Scaling.....	33
SetABSenseCW.....	34
SetABSenseCCW.....	34
SelectPlaneXY.....	34
SelectPlaneYZ.....	34

SelectPlaneZX.....	35
SelectPlane.....	35
3.2.5.2 GROUP: MOVETO.....	35
IncMoveX.....	35
IncMoveY.....	36
IncMoveZ.....	36
InsertMove.....	36
Move4thAxis.....	36
Move5thAxis.....	37
MoveA.....	37
MoveB.....	37
MoveFromXYZ.....	37
MoveIJKUVW.....	38
MoveI.....	38
MoveJ.....	38
MoveK.....	38
MoveR.....	39
MoveX.....	39
MoveY.....	39
MoveZ.....	39
MoveXYZIJK.....	40
MoveXYZUVW.....	40
3.2.5.3 GROUP: TOOLCHANGE.....	40
CreateLatheTool.....	41
CreateMillTool.....	41
LoadTool.....	41
SelectTool.....	41
SelectAndLoadTool.....	42
SetTLBFileName.....	42
3.2.5.4 GROUP: CANNEDCYCLES.....	42
ArcCycle.....	43
ArcHoleCycle.....	43
BoreCycle.....	43
CircleCycle.....	43
CallCycle.....	43
CancelCannedCycle.....	43
CancelPatternCycle.....	44
CircPocketCycle.....	44
CircPocketCycleCW.....	44
CircPocketCycleCCW.....	44
CirclePocketCycleDef.....	45
DefCycle.....	45
DrillCycle.....	45
LineCycle.....	46
PatternCycle.....	46
RectPocketCycle.....	46
RectPocketCycleDef.....	47
ReturnToInitial.....	47
ReturnToReference.....	47
SlotPocketCycle.....	47
3.2.5.5 GROUP: MISCELLANEOUS.....	48

AirBlast .....	48
ChangePallet .....	49
CoolantON .....	49
CoolantOFF .....	49
CtantSurfSpeed .....	49
CtantSurfSpeedCancel.....	50
HSpindleStop.....	50
LineNumber .....	50
MaxSpindleSpeed .....	50
OptStop.....	50
Pause0.....	51
Pause1.....	51
ProgramEnd.....	51
SetMachineDatum .....	51
SetMCHFileName.....	52
SetOrigRelocation .....	52
SpindleCW.....	52
SpindleCCW .....	52
SpindleSpeed .....	53
SpindleStop .....	53
Stop.....	53
ToolClamp .....	53
VSpindleStop.....	53
3.2.5.6 GROUP: FEED RATE .....	54
FeedPerMinute .....	54
FeedPerRevolution.....	54
InvTimeFeed.....	54
OutFeed.....	54
SetFeed .....	55
SetMaxFeed .....	55
3.2.5.7 GROUP: UNSUPPORTED.....	55
Error.....	55
3.2.5.8 GROUP: REFERENCE POINT RETURN .....	56
ReturnToRefP.....	56
ReturnFromRefP .....	56
3.2.5.9 GROUP: CUTTER COMPENSATION.....	56
CutterCompCancel.....	57
CutterCompLeft.....	57
CutCompRight.....	57
LengthCompCancel.....	57
LengthCompMinus .....	58
LengthCompPlus .....	58
LengthOffDoubleNegative .....	58
LengthOffDoublePositive.....	58
LengthOffNegative.....	58
LengthOffPositive .....	59
SetCutterCompD .....	59
SetDiamOffset .....	59
SetLengthOffset.....	59
SetLengthCompH .....	60
3.2.5.10 GROUP: IGNORE .....	60

Ignore.....	60
3.2.5.11 GROUP: WORK COORDINATES .....	60
AxisRotation.....	61
CancelAxisRotation .....	61
ChangeOffset .....	61
ChangeAllWorkCoord.....	61
DisableWorkCoord .....	62
SetCoordSystem .....	62
SetWorkCoord1 .....	62
SetWorkCoord2.....	62
SetWorkCoord3.....	62
SetWorkCoord4.....	63
SetWorkCoord5.....	63
SetWorkCoord6.....	63
SetWorkCoord.....	63
SetWorkCoordNext.....	64
3.2.5.12 GROUP: ABSOLUTE ZERO POSITION .....	64
SetAbsoluteZero .....	64
3.2.5.13 GROUP: THREADING .....	64
ThreadCycle .....	65
ThreadMove.....	65
3.2.5.14 GROUP: SUBPROGRAM .....	65
MacroCall.....	65
ProgramStart.....	65
SubProgramCall .....	66
SubProgramEnd.....	66
SubProgramEx .....	66
SubProgramRet.....	67
SubProgramRetEnd .....	67
SubProgramRetEx.....	67
SubProgramStart.....	67
DefineGlobalVar .....	68
DefineLocalVar .....	68
DefineParameter .....	68
GetActiveTool .....	68
GetActiveDCode.....	69
GetCurrentReg .....	69
GetCurrentVar .....	69
GetGlobalVar .....	70
GetLocalVar.....	70
GetParameter .....	70
GetVariable .....	70
SetGlobalVar .....	70
SetLocalVar .....	71
SetParameter .....	71
SetVariable .....	71
3.2.5.17 GROUP: COMMENTS .....	71
CommentEnds.....	71
CommentFollows.....	72
CommentStarts.....	72
CreateStockBox.....	72

CreateStockCyl.....	73
CreateStockCylFixt.....	73
CreateStockCylHole .....	74
CreateStockFixt.....	74
CreateStockHole .....	74
CreateStockSTL .....	75
CreateStockSTLFixt .....	75
SetSTKFileName.....	75
3.2.5.21 GROUP: LATHECYCLES .....	76
EndLabel.....	76
FaceDrillCycle .....	76
FinishCycle .....	76
RoughXCycle.....	77
RoughZCycle.....	77
StartLabel .....	77
TurnCycle .....	77
TurnGrooveCycle .....	78
TurnTapCycle.....	78
3.2.5.22 GROUP: PROGRAMMING .....	78
Condition.....	79
EndLoop .....	79
Goto .....	79
SetLabel.....	79
StartLoop .....	80
3.2.5.23 GROUP: ARITHMETIC FUNCTIONS .....	80
Add.....	80
Divide.....	80
Exponent.....	80
Module .....	81
Multiply.....	81
Subtract.....	81
3.2.5.24 GROUP: BOOLEAN FUNCTIONS .....	81
And.....	82
Equal.....	82
Great.....	82
GreatOrEqual .....	82
Less .....	82
LessOrEqual.....	83
NotEqual .....	83
Or.....	83
Xor .....	83
3.2.5.24 GROUP: TRIGONOMETRIC FUNCTIONS .....	83
Absolute.....	84
ArcCosine .....	84
ArcTangent .....	84
Cosine.....	84
Exponent.....	85
Logarithm .....	85
Round .....	85
Sine.....	85
SquareRoot.....	85

Tangent.....	86
<b>3.2.6 REGISTER Keywords: .....</b>	<b>86</b>
r_Arc_CCW.....	86
r_Arc_CW.....	86
r_Arc_Radius.....	86
r_Arc_Angle.....	87
r_Arc_Normal_U.....	87
r_Arc_Normal_V.....	87
r_Arc_Normal_W.....	87
r_Axis_U.....	88
r_Axis_V.....	88
r_Axis_W.....	88
r_ChangeOff_4thAxisValue.....	88
r_ChangeOff_5thAxisValue.....	88
r_ChangeOff_Func.....	89
r_ChangeOff_Num.....	89
r_ChangeOff_ToolLength.....	89
r_ChangeOff_XValue.....	89
r_ChangeOff_YValue.....	89
r_ChangeOff_ZValue.....	90
r_Coord_Mode.....	90
r_Cycle_Num.....	90
r_Cycle_Param.....	90
r_Diam_Offset.....	91
r_Drill_Pos_A.....	91
r_Drill_Pos_B.....	91
r_Drill_Pos_X.....	91
r_Drill_Pos_Y.....	91
r_Drill_Pos_Z.....	92
r_Dwell_Int_Time.....	92
r_Dwell_Time.....	92
r_Expression.....	92
r_Feed_Rate.....	93
r_GenCycle_Angle.....	93
r_GenCycle_Angle_Y.....	93
r_GenCycle_Depth.....	93
r_GenCycle_Feed_Finish.....	93
r_GenCycle_Feed_Rough.....	94
r_GenCycle_Finishing.....	94
r_GenCycle_Inc_Move.....	94
r_GenCycle_N_Holes_X.....	94
r_GenCycle_N_Holes_Y.....	95
r_GenCycle_Plane.....	95
r_GenCycle_Pocket_R.....	95
r_GenCycle_Pocket_Alt_R.....	95
r_GenCycle_R_Allow.....	96
r_GenCycle_Retract.....	96
r_GenCycle_RetRapid.....	96
r_GenCycle_Roughing.....	96
r_GenCycle_Sense_CW.....	96
r_GenCycle_Sense_CCW.....	97

r_GenCycle_Slot_Dir.....	97
r_GenCycle_Start_Angle.....	97
r_GenCycle_Start_Pos.....	97
r_GenCycle_Tool_Radius.....	97
r_GenCycle_X-Allow.....	98
r_GenCycle_X_Dim.....	98
r_GenCycle_X_Pos.....	98
r_GenCycle_Y-Allow.....	98
r_GenCycle_Y_Dim.....	98
r_GenCycle_Y_Pos.....	99
r_GenCycle_Z-Allow.....	99
r_GenCycle_Z_Init.....	99
r_GenCycle_Z_Peek.....	99
r_GenCycle_Z_Pos.....	99
r_GenCycle_Z_Retract.....	100
r_Global_Variable_Num.....	100
r_Global_Variable_String.....	100
r_Goto_Line.....	100
r_Groove_Amount.....	100
r_Groove_Feed.....	101
r_Groove_IncX.....	101
r_Groove_IncZ.....	101
r_Groove_U.....	101
r_Groove_V.....	102
r_Groove_X.....	102
r_Groove_Z.....	102
r_Hole_Center_X.....	102
r_Hole_Center_Y.....	102
r_Hole_Diam_Circ.....	103
r_Hole_First_X.....	103
r_Hole_First_Y.....	103
r_Hole_Int_Ang.....	103
r_Hole_Num.....	103
r_Hole_Retract.....	104
r_Inches.....	104
r_Int_Ignore.....	104
r_IntVar_Local_Value.....	104
r_IntVar_Global_Value.....	104
r_IntVar_Param_Value.....	105
r_IntVar_Value.....	105
r_Last_Pos_X.....	105
r_Last_Pos_Y.....	105
r_Last_Pos_Z.....	105
r_Length_Offset.....	106
r_Local_Variable_Num.....	106
r_Local_Variable_String.....	106
r_Millimeters.....	106
r_Misc_MCH_FileName.....	106
r_Num_Diam_Offset.....	107
r_Num_Length_Offset.....	107
r_Param_Variable_Num.....	107

r_Param_Variable_String.....	107
r_Params.....	107
r_Peck_Inc.....	108
r_Peck_Retract.....	108
r_Peck_Repeats.....	108
r_Peck_Z.....	108
r_Polar_Angle.....	109
r_Polar_IncAngle.....	109
r_Polar_Radius.....	109
r_Pos_4thAxis.....	109
r_Pos_5thAxis.....	109
r_Pos_X.....	110
r_Pos_Y.....	110
r_Pos_Z.....	110
r_Pos_I.....	110
r_Pos_J.....	110
r_Pos_K.....	111
r_Pos_U.....	111
r_Pos_V.....	111
r_Prog_Name.....	111
r_Real_Ignore.....	112
r_RealVar_Param_Value.....	112
r_RealVar_Global_Value.....	112
r_RealVar_Local_Value.....	112
r_RealVar_Value.....	112
r_Reference_Pos_U.....	113
r_Reference_Pos_V.....	113
r_Reference_Pos_W.....	113
r_Reference_Pos_X.....	113
r_Reference_Pos_Y.....	113
r_Reference_Pos_Z.....	114
r_Retract_Mode.....	114
r_Rot_Angle.....	114
r_Rot_Center_X.....	114
r_Rot_Center_Y.....	114
r_Scal_Factor.....	115
r_Scal_Center_X.....	115
r_Scal_Center_Y.....	115
r_Scal_Center_Z.....	115
r_Spindle_Speed.....	115
r_Stock_Box.....	116
r_Stock_Box_Type.....	116
r_Stock_CenterX.....	116
r_Stock_CenterY.....	116
r_Stock_CenterZ.....	116
r_Stock_Cyl.....	117
r_Stock_Cyl_Axis.....	117
r_Stock_Cyl_Type.....	117
r_Stock_Diameter.....	117
r_Stock_Length.....	117
r_Stock_LengthX.....	118

r_Stock_LengthY.....	118
r_Stock_LengthZ.....	118
r_Stock_MinX.....	118
r_Stock_MinY.....	118
r_Stock_MinZ.....	119
r_Stock_Type.....	119
r_Stock_STL.....	119
r_Stock_STL_Filename.....	119
r_Stock_STL_Type.....	119
r_Stock_STK_Filename.....	119
r_String_Ignore.....	120
r_StringVar_Local_Value.....	120
r_StringVar_Global_Value.....	120
r_StringVar_Param_Value.....	120
r_StringVar_Value.....	121
r_Subprogram_EndLineRet.....	121
r_SubProgram_EndrsTimes.....	121
r_SubProgram_GotoLine.....	121
r_SubProgram_FirstLine.....	121
r_SubProgram_LastLine.....	121
r_SubProgram_Number.....	122
r_SubProgram_Start.....	122
r_SubProgram_StrStart.....	122
r_SubProgram_String.....	122
r_SubProgram_Times.....	122
r_Tapper_Radius.....	123
r_Thread_Angle.....	123
r_Thread_Allowance.....	123
r_Thread_Feed.....	123
r_Thread_First_Depth.....	124
r_Thread_MinLen.....	124
r_Thread_NTimes.....	124
r_Thread_Pos_X.....	124
r_Thread_Pos_Z.....	125
r_Thread_Total_Depth.....	125
r_Thread_Taper_Inc.....	125
r_Tool_Alpha.....	125
r_Tool_Angle.....	125
r_Tool_Ang_Orient.....	126
r_Tool_Angle_Top.....	126
r_Tool_Aux.....	126
r_Tool_Ax_Crad_Offset.....	126
r_Tool_BC_Crad_Offset.....	126
r_Tool_Corner_Rad.....	127
r_Tool_Curr_Alpha.....	127
r_Tool_Curr_Aux.....	127
r_Tool_Curr_Corner_R.....	127
r_Tool_Curr_Diam.....	128
r_Tool_Curr_Height.....	128
r_Tool_Cut_Dir.....	128
r_Tool_Diameter.....	128

r_Tool_Height .....	128
r_Tool_Holder_Angle .....	129
r_Tool_Holder_D .....	129
r_Tool_Holder_Tip.....	129
r_Tool_Holder_Type.....	129
r_Tool_Holder_H.....	130
r_Tool_Home_Num .....	130
r_Tool_Ins_Circle .....	130
r_Tool_Name.....	130
r_Tool_Number.....	130
r_Tool_NTG_Filename.....	131
r_Tool_Orient.....	131
r_Tool_Orient_X.....	131
r_Tool_Orient_Y.....	131
r_Tool_Orient_Z.....	131
r_Tool_Ref.....	131
r_Tool_Shank_D.....	132
r_Tool_Shank_H.....	132
r_Tool_TLB_Filename.....	132
r_Tool_Turret_Number.....	132
r_Tool_Turret_Pos.....	132
r_Tool_Type.....	133
r_Tool_Width .....	133
r_Tool_X_Prog_Point .....	133
r_Tool_Z_Prog_Point .....	133
r_Turn_Cycle_Depth .....	134
r_Turn_Cycle_End_Label.....	134
r_Turn_Cycle_First_Block.....	134
r_Turn_Cycle_Last_Block.....	134
r_Turn_Cycle_Ret_Ang.....	135
r_Turn_Cycle_Retract.....	135
r_Turn_Cycle_Start_Label .....	135
r_Turn_Cycle_X_Allow .....	135
r_Turn_Cycle_Z_Allow .....	135
r_Units.....	136
r_Var_Access_Mode .....	136
r_Variable_Num.....	136
r_Variable_String.....	136
<b>3.2.7 FORMAT keywords .....</b>	<b>136</b>
concat .....	137
data_separator .....	137
code_num.....	137
code_and_subcode.....	137
subcode_num .....	137
separator.....	137
label .....	137
<b>3.2.8 FUNCTION PARAMETERS Keywords:.....</b>	<b>137</b>
peck.....	138
bottomstop.....	138
bottomCW .....	138
oriented_bottomCW.....	138

manualfeed.....	138
retractfeed.....	138
dwell.....	138
<b>3.5 Parameters:.....</b>	<b>138</b>
AB_AXIS_SENSE_MODAL.....	138
A_AXIS_DEFAULT_SENSE.....	139
ARC_CHECK.....	139
ARC_TOL_INCH.....	139
ARC_TOL_MM.....	139
ARCCENTER_ABSOLUTE.....	140
BOLT_PATTERN_MODE.....	140
CIRCULAR_PARAMS.....	140
B_AXIS_DEFAULT_SENSE.....	140
CANNED_CYCLES_MODE.....	141
CHECK_COOLANT.....	141
CHECK_DIAM_OFFSET.....	141
CHECK_HELICAL_MOVE.....	141
CHECK_LENGTH_OFFSET.....	141
CHECK_SPINDLE.....	142
CHECK_SPINDLE_SPEED.....	142
CHECK_SPIRAL_MOVE.....	142
CHECK_TOOL_LOAD.....	142
CHECK_TRAVELS.....	142
CHECK_WORK_OFFSET.....	143
CONTROL_TYPE.....	143
CUTTERCOMP_ENTRY_MODE.....	143
CYCLE_DEPTH_ALWAYS_NEGATIVE.....	143
CYCLE_RETURN.....	144
CYCLE_RETRACT.....	144
DECIMAL_POINT.....	144
DECIMAL_SEPARATOR.....	144
DWELL_UNITS.....	145
ENABLE_WARNINGS.....	145
ERROR_MODE.....	145
FIRST_CODES.....	145
FEED_RATE_UNITS.....	145
IJK_MODALITY.....	146
INVERSE_TIME_FEED.....	146
JUMP_MODE.....	146
LATHE_FEED_MODE.....	146
LATHE_INSERT_MOVES.....	146
LATHE_PROG_MODE.....	147
LCYCLE_FACTOR.....	147
LOFFSET_AXIS.....	147
LOFFSET_SIGN.....	147
MACRO_ARG_LIST.....	148
MACRO_ARG_MAX.....	148
MACRO_ARG_TYPE.....	148
NC_IGNORE_CASE.....	148
NC_FORMAT.....	148
NULL_VALUES.....	149

OFFSET_INDEX_START .....	149
PARAMSBEFORE_CANNEDCYCLE .....	149
POCKET_CYCLE_MODE .....	149
POLAR_MOVE_MODE .....	150
RAPID_CANCEL CYCLE .....	150
RAPID_MOVE_MODE .....	150
REFERENCE_MOVE .....	150
REFERENCE_POINT .....	150
ROTATION_MOVE_MODE .....	151
RVP_FORMAT .....	151
SPINDLE_SPEED_UNITS .....	151
SUBPROG_CALL .....	151
SUBPROG_DEF_CALL_CODE .....	151
SUBPROG_EXTERNAL .....	152
SUBPROG_MAX .....	152
SUBPROG_MAX_REP .....	152
SUBPROG_NEST .....	153
SUBPROG_NUMBER_DIGITS .....	153
SUBPROG_NTIMES_DIGITS .....	153
SUBPROG_POS .....	153
SUBPROG_PROGNUMBER_POS .....	154
SUBPROGEX_CALL .....	154
SUBPROGEX_POS .....	154
SUBPROGEX_NUMBER_DIGITS .....	155
SUBPROGEX_NTIMES_DIGITS .....	155
SUBPROGEX_PROGNUMBER_POS .....	155
STOCK_COMMENTS .....	156
THREAD_CUTTING_FACTOR .....	156
TOOL_COMMENTS .....	156
UNIT_CHANGE .....	156
USE_DEFAULT_TOOL .....	157
VARIABLE_FACTOR .....	157
VARIABLE_SUPPORT .....	157
VARIABLE_MAX .....	157
WARNING_AS_ERROR .....	157
<b>3.4. Error messages:.....</b>	<b>158</b>
<b>3.4.1 Fatal Errors: .....</b>	<b>158</b>
BADCOMMAND: .....	158
BADFORMAT: .....	158
BADMODIF: .....	158
BADCYCLE: .....	158
BADCOMMANDINGROUP: .....	158
BADDELIMITER: .....	158
BADMAINGROUP: .....	158
BEGINBRACKETNOTFOUND: .....	158
Default .....	158
ENDBRACKETNOTFOUND: .....	158
GCF_FERROR: .....	158
GCF_TOOMANYINCLUDES: .....	159
INTERNALTABLEOVERFLOW: .....	159
NOENDQUOTES: .....	159

NOBEGINQUOTES:.....	159
NOENDPARENTHESIS:.....	159
<b>3.4.2 Non Fatal Errors:.....</b>	<b>159</b>
GCF_EOF:.....	159
PASTENDMARK: .....	159
<b>4. RP* File Example .....</b>	<b>160</b>
<b>1.4 Error checking.....</b>	<b>167</b>
<b>1.4.1 RP* Configuration Load and Syntax Errors.....</b>	<b>167</b>
1.4.1.1 RP* Errors.....	167
<b>1.4.2 NC and CNC Program Syntax and Semantic checking .....</b>	<b>169</b>
1.4.2.1 NC and CNC Program Error messages.....	170
Syntactic error codes:.....	170
Semantic error codes:.....	171
<b>5. GCF File. Updates Record.....</b>	<b>180</b>
<b>5.1 January 29 1999 .....</b>	<b>180</b>
<b>5.2 February 2 1999.....</b>	<b>180</b>
<b>5.3 February 10 1999.....</b>	<b>180</b>
<b>5.3 June 11 1999.....</b>	<b>180</b>
<b>Long Index:.....</b>	<b>181</b>