

SmartCAM®
Code Generation Guide

for Microsoft® Windows™

CAMAX
Manufacturing Technologies

The information in this document is subject to change without notice. CAMAX MANUFACTURING TECHNOLOGIES, INC. MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OR MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. CAMAX Manufacturing Technologies, Inc. shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this material. No part of this publication may be reproduced, transmitted, transcribed, stored in a retrieval system, or translated into any language in any form by any means without the written permission of CAMAX Manufacturing Technologies, Inc.

© 1996 by CAMAX Manufacturing Technologies, Inc. All rights reserved.

Software License Notice

Your license agreement with CAMAX Manufacturing Technologies, Inc., which is included in the *Installation Guide for All SmartCAM Products*, specifies the permitted and prohibited uses of the product. Any unauthorized duplication or use of SmartCAM in whole or in part or in any other storage and retrieval system is forbidden.

Licenses and Trademarks

SmartCAM, CAM Connection, and Tape-to-Shape are registered trademarks of Point Control Co., a subsidiary of CAMAX Manufacturing Technologies, Inc.; SmartCAM Advanced 3-D Machining, SmartCAM Advanced Fabrication, SmartCAM Advanced Turning, SmartCAM Advanced Wire EDM, SmartCAM Automated Shape Machining, SmartCAM Customization Toolkit, SmartCAM FreeForm Machining, SmartCAM Power Nesting, SmartCAM Production Fabrication, SmartCAM Production Milling, SmartCAM Production Turning, AutoCAD SmartCAM Connection, CAMCAD Standard, CNC Process Model, Edit Plus, and JOE are trademarks of Point Control Co., a subsidiary of CAMAX Manufacturing Technologies, Inc.

ACIS is a registered trademark of Spatial Technology, Inc. Adobe and Acrobat are registered trademarks of Adobe Systems Incorporated. Hewlett-Packard, HP, HP/UX, and HP Series 700 are trademarks of Hewlett-Packard Company. HyperHelp is a trademark of Bristol Technology, Inc. IGES/Lib is a trademarked and copyrighted product of International TechneGroup, Inc. Intel and Pentium are registered trademarks of Intel Corporation. Microsoft, MS-DOS, Windows, Windows NT, WIN32s and the Windows logo are either registered trademarks or trademarks of Microsoft Corporation. SPARC is a registered trademark of SPARC International, Inc. Solaris, SunOS and OpenWindows are trademarks of Sun Microsystems, Inc. UNIX is a registered trademark of Novell, Inc.

The license management portion of this product is based on Élan License Manager, © 1989–1996 Élan Computer Group, Inc. All rights reserved. Élan License Manager is a trademark of Elan Computer Group, Inc.

Published: June 1996



1750 Willow Creek Circle
Eugene, OR 97402-0269 USA
(541) 344-4470 • FAX (541) 342-8277
Toll Free 1-800-394-5300

7851 Metro Parkway
Minneapolis, MN 55425-1528 USA
(612) 854-5300 • FAX (612) 854-6644
Toll Free 1-800-394-5300

Contents

1 Code Generation Overview

Introduction	1-1
Differences Between Manual- and Computer-Generated Code.	1-1
Code Generation vs. Postprocessing.	1-2
Levels of Modification.	1-2
SmartCAM Files Required for Code Generation	1-3
Process Model File.	1-4
Job Operations Setup File	1-5
Machine File.	1-5
Template File	1-6
NC Code File	1-7
How SmartCAM Generates Code	1-7
Using the Process Model Database.	1-8
Using the Tool Path	1-9
Using the Process Model and Job Operations Setup Files	1-9
Using Template Words.	1-9
Transferring Tool Path Information to the Template File	1-10
Outputting Information through the Machine File.	1-11
Sending Information to the Template File	1-13
Modifying a Code Generator	1-14
Knowledge of SmartCAM Required	1-15
Modification Guidelines	1-15
Using An Existing Code Generator	1-16
Creating a Test File	1-16
Identifying Changes to Make in Code	1-16

Modifying a Code Generator (<i>cont'd</i>)	
Changing the Machine File	1-16
Changing the Template File	1-17
Practice: Making Changes for Arc Code	1-17

2 Using Machine Define

Introduction	2-1
Files Included with the Machine Define Utility	2-1
File Versions	2-2
Starting Machine Define	2-2
Starting from Program Manager	2-2
Starting from File Manager	2-3
The Machine Define Window	2-3
Components of a Machine File	2-4
The Question List Box	2-5
The Explanation Box	2-5
The Selection Box	2-5
Using The File Menu	2-6
Opening a Machine File	2-6
Saving Your Changes	2-7
Printing a Machine File	2-9
Changing the Printer Setup	2-9
Closing the Machine Define Module	2-10
Using the View Menu	2-10
Viewing the Item List	2-11
Viewing the Category List	2-11
Selecting Categories for Viewing	2-11
Using the Search Menu	2-12
Text Search Limitations	2-13
Editing a Machine File	2-14
Using the Keyboard	2-15
Error Messages	2-16

3 Machine File Overview

Introduction	3-1
Numeric Formats	3-2
Format Type	3-2
Sign Control	3-2

Numeric Formats (<i>cont'd</i>)	
Digits Allowed to Left and Right of Decimal Point	3-3
Field Width	3-3
Sign Position	3-3
Generating Code for Subroutines	3-3
Regular Subroutines.	3-4
Drill Subroutines.	3-5
Code Filtering for Mesh Polylines	3-6
Generating Code for Tool Changes and Tool Plane Changes.	3-6
Transition between Tool Planes and Tool Changes.	3-7
Generating Code for Rotary Positioning and Machining	3-9
Machine File Questions for 4th- and 5th-Axis Positioning	3-11
Machine File Questions for Rotary-Axis Machining	3-12
Template Variables for Multiaxis and Rotary Machining	3-12
Linear Equivalent Feed Rate Conversion	3-15

4 Machine File Reference

General	4-2
Block Number Control	4-4
Units and Axis Orientation	4-5
Feeds, Speeds, and Dwell	4-9
Tool Change.	4-12
Rapid Positioning	4-15
Linear Profile Moves	4-16
Circular Arc Profile Moves	4-17
Profile Contouring	4-20
Cutter Compensation	4-21
Cycle Time	4-25
Mill and Lathe Peck Drill Cycle Parameters.	4-26
Mill Cycles	4-28
Lathe Cycles	4-31
Lathe Threading.	4-31
Lathe Groove Cycle	4-36
Lathe 4-Axis	4-37
Punch Cycles	4-40
4-Axis Wire EDM	4-46
3-D Arcs and Helixes	4-47
4- and 5-Axis Positioning	4-50
Subprogramming	4-54

User Commands	4-56
Tape-to-Shape	4-57
Rotary Contouring	4-60

5 Template File Overview

Introduction	5-1
How to Read and Edit the Template File	5-1
Template File Sections	5-2
System-Defined Template Sections	5-2
User-Defined Template @ Sections	5-3
Simple Template Sections	5-4
Types of Template Words	5-4
Switches	5-5
Numeric Template Words	5-6
String Template Words	5-6
Control Words	5-7
Using Literals in an @ Section	5-7
Conditional Brackets (<>)	5-7
Logic Control	5-8
Adding Comments	5-9
Mathematical and Trigonometric Functions	5-9
Basic Statement	5-9
Math Functions	5-10
Trigonometric Functions Supported by SmartCAM	5-11
Order of Operations	5-11
System-Defined Template @ Sections	5-12
Subprogramming Application @ Sections	5-21
Sample User-Defined Template Sections	5-22

6 Template Word Reference

Alphabetical List of Template Words	6-1
Lists of Template Words by Category	6-37
Arcs	6-37
Block Number Control	6-38
Control Functions	6-39
Cycle Time Calculation	6-39
Feed Control	6-39

Lists of Template Words by Category (<i>cont'd</i>)	
Fixed Cycle, Lathe Groove (@FXD6; OD/ID groove)	
(@FXD7; face groove)	6-40
Fixed Cycle, Lathe Peck Drill (@FXD5)	6-40
Fixed Cycle, Lathe Tapping (@FXD4)	6-40
Fixed Cycle, Lathe Threading (@FXD1)	6-41
Fixed Cycle, Lathe Turning (@FXD2)	6-43
Lines	6-44
Logic Functions	6-45
Mathematical Functions	6-45
Spindle Control	6-45
String Words	6-46
Tool Control	6-46
Tool Information	6-46
Trigonometric Functions	6-47
User-Assigned Words	6-47
Work Coordinate System	6-48
User-Definable Template Variables	6-48

7 Testing and Verification

Introduction	7-1
Creating Automatic Error Reports	7-1
Isolating a Problem	7-2
Modifying an .smf File	7-3
Invalid Machine File Error	7-3
Working with Template Files	7-3
Common Mistakes	7-3
Adding Comments	7-3
Using Literals	7-3
Marking Sections in Code	7-4
Turning Section Literals On and Off	7-4
Using Literals within a Section	7-5
Testing Logic Statements	7-6
Common Code Generator Problems and Solutions	7-7
 Index	 I-1

Code Generation Overview

Introduction

SmartCAM's code generators enable you to produce NC code automatically for a variety of CNC machines. Each SmartCAM application package comes with several code generators. In addition, CAMAX Manufacturing Technologies maintains a library of code generators for many different machine and controller combinations. When a code generator passes a series of tests with benchmark parts, the code generator is posted on CAMAX's electronic bulletin board system (BBS). These code generators are available at no charge.

- ☞ Talk with your authorized SmartCAM dealer for more information about accessing the code generator library.

This chapter provides general considerations for writing and changing SmartCAM code generators. It first addresses how code generators work and then supplies a simple test case for you to work through to gain experience. You'll find the process is not very complicated once you develop some confidence in writing or modifying code generators.

Differences Between Manual- and Computer-Generated Code

There are some fundamental differences between code written manually and computer-generated code. When a programmer writes code, the programmer tries to reduce the amount of writing as much as possible. That's why machine controllers historically add more canned cycles. Less code not only reduces the amount of writing but also reduces the chances of programming errors.

Computer-generated code eliminates the need to write a program manually. Computer-generated code files may be longer than equivalent manually written files, but they do not result in additional programming work or create any differences in actual machine tool motion. Computer-generated code is consistent, and the computer is very unlikely to make a typographical error.

Keep these fundamental differences in mind when modifying your code generator. You may need to decide between very compact code (the way you'd write it manually) and a slightly longer code file.

Code Generation vs. Postprocessing

Code generation is the automatic formatting process SmartCAM uses to convert a CNC process model to NC code for your machine and controller combination. SmartCAM combines information from your process model (.pm4) file with tooling information from the job operations setup (.jof) file, tailoring it with two files containing information about your machine and programming style (the .smf file and the .tmp files, respectively).

Code generators differ from postprocessors in the following ways:

- Code generation works directly from SmartCAM's database. Postprocessing works on a cutter location file. When you create a CNC process model on your screen, code generation information is built into the process model and is there as you work. In conventional programming, postprocessing is an extra step that happens last.
- SmartCAM users can create and edit their code to suit their preferences and requirements.

Understanding how SmartCAM's code generators work enables you to make changes in the way SmartCAM outputs code. The relative ease of changing a code generator depends on the complexity of your task. Making simple modifications to existing code generators generally requires no advanced skills. Conversely, writing a complex code generator can be time-consuming and requires some special knowledge. The more you know about your machine tool, SmartCAM, and computers, the easier it is to modify code generators.

Levels of Modification

There are two levels of modification in this chapter. Review your code, see what needs to be changed, and review the descriptions for level-1 and level-2 requirements.

Level-1 Modifications

Simple modifications are referred to as level-1 code generator work. Level-1 work involves code generators that are producing code but need minor modifications to work correctly for your machine. Examples of level-1 changes include the following:

- Block number control
- Numeric formats
- Relocation of some G and M codes
- Minor alteration of fixed cycles

You don't need computer programming skills to make level-1 modifications, but you should understand your machine tool requirements. You can't tell SmartCAM how to format code correctly if you don't know what that correct code is. You also need some minimal knowledge about how SmartCAM code generators work and how to alter them.

Level-2 Modifications


Level-2 work is more involved and includes making extensive alterations to an existing code generator or creating a new code generator. You may need to work with math and logic statements and make major alterations to code generator files. The following are examples of level-2 work:

- Math and logic functions
- Customized template file sections
- Special user-assigned template words
- Major alterations of fixed cycles

People making level-2 modifications need a thorough understanding of SmartCAM and their machine tools. Some knowledge of a computer programming language such as BASIC is also very helpful. If you plan to work a lot with code generators, we recommend you attend a SmartCAM code generator course.

SmartCAM Files Required for Code Generation

This section explains the files SmartCAM needs to generate code. Information for the code generation process comes from four SmartCAM files. The process model and job operations setup files are specific to your job. The machine and template files are specific to your machine tool and controller combination. Figure 1-1 shows the files necessary for SmartCAM to produce code and their location on your computer. It also shows some information SmartCAM uses to generate code. Notice that SmartCAM combines the information from these files to produce a fifth file, containing the NC code for your machine tool.

 The SmartCAM Path directory structure you used to install your applications is the default path to the various file locations.

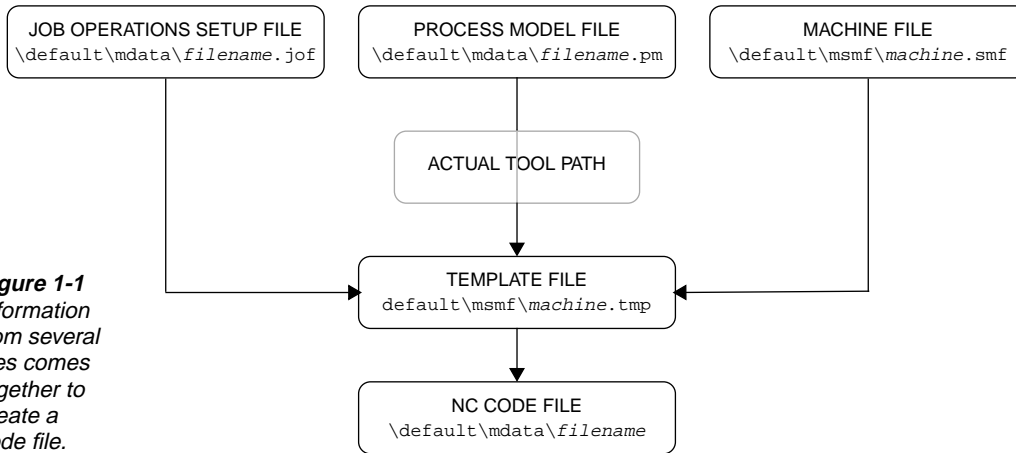


Figure 1-1
Information from several files comes together to create a code file.

Process Model File

The process model file contains the tool path information for the part model you have built in SmartCAM. SmartCAM graphically displays this file on the screen as a CNC process model. When you are ready to produce code for this model, select Code from the Process menu. The control panel asks for the name of the code generator (machine and template) files. Then it generates code in the correct format for the designated machine. The code is stored as an ASCII file with a path and name of your choice.

Each 3-D process model file has a .pm4 extension. Though you can save process model files in any directory, they are typically stored in application-specific subdirectories of your SmartCAM Path directory:

SmartCAM Application	Location of Process Model Files ^a
FreeForm Machining	\\mill\\ffdata or ffdata_m
Advanced 3-D Machining	\\mill\\amdata or amdata_m
Advanced Fabrication	\\fab\\afdata or afdata_m
Advanced Turning	\\turn\\atdata or atdata_m
Advanced Wire EDM	\\wire\\awdata or awdata_m
Production Milling	\\mill\\mdata or mdata_m
Production Turning	\\mill\\tdata or tdata_m

a. Directory names ending with “_m” contain metric process model files.

Process model tool paths are typically made up of series of elements with various properties, including the following:

- Tool type
- Work plane
- Offset
- Tool number

During code generation, the process model database outputs information about each element to the code generator for processing. This may include the following information:

- Coordinate position of the move
- Type of move (for example, rapid, linear, or circular)
- Offset direction of the tool

Job Operations Setup File

The job operations setup file contains the list of steps you need to produce the part. Each step consists of a tool and an operation used to perform one step in the machining of a part. Each step contains data about the type, size, and model of the tool as well as the tool angles and feeds that your controller uses for the tool. Each job operations setup file has a `.jof` extension. Though you can store job operations setup files in any directory, you will usually find them in the same directory as your process model files. SmartCAM sends all the job operations setup file information to the code generator, where it becomes part of the database the code generator accesses when it produces code.

Machine File

Machine files contain information specific to the machine for which you are generating code. The machine file is a comprehensive list of questions about your machine tool configuration, the G and M codes it uses, and numeric formats.

All machine files have an `.smf` extension. Though you can save machine files in any directory, they are usually stored in application-specific subdirectories of the `SmartCAM Path` directory:

SmartCAM Application	Location of Machine Files
FreeForm Machining	<code>\mill\msmf</code>
Advanced 3-D Machining	<code>\mill\msmf</code>
Advanced Fabrication	<code>\fab\afsmf</code>
Advanced Turning	<code>\turn\atsmf</code>
Advanced Wire EDM	<code>\wire\awsmf</code>
Production Milling	<code>\mill\msmf</code>
Production Turning	<code>\turn\tsmf</code>

Template File

Template files contain information specific to code format for the machine tool you are using. All template files have a `.tmp` extension and are normally in the same directory as the machine file. The template file contains information that determines the code output sequence and format. It also contains math functions and logic statements. The arrangement and format of this file is the key to producing code exactly as you want it.

If you are modifying a code generator at level-1, you probably won't make many changes to a template file. Those modifications will be limited to adding, removing, or rearranging the words in this file. However, if you are building a level-2 code generator, most of your time will be spent working with this file.

A template file is divided into sections. In each section, SmartCAM uses template words, such as `#MOV`, and literals, such as the letter `X`, which may be within conditional brackets (`<>`). Each section contains information that outputs code for a specific operation. For example, the `@LINE` template section contains information for outputting a linear cutting move (see figure 1-2). The contents and arrangement of this `@LINE` section determine the format of the code output.

Figure 1-2
Each section in a template file begins with @.

```

TEMPLATE FILE \mill\msmf\machine.tmp
@LINE
<#DCOMP> <D#DOFF> <#MOV> <X#XPOS> <Y#YPOS> <Z#ZPOS> <F#FEED>

```

☞ Template words are used like variables in computer languages. Template sections are used like functions, procedures, and subroutines in computer languages.

Figure 1-3 shows a template section that contains information for code output for a line.

Template sections are easy to identify, because they begin with the `@` symbol. The template words in the `@LINE` section, and their location within this section, determine what the NC code looks like.

SmartCAM reads template sections the same way a book is read in western languages: from left to right and from top to bottom. Figure 1-3 shows how the template file processes the information as formatted by the machine file and sends it to the NC code file.

Figure 1-3
The code output follows the format of the template file.

```

TEMPLATE FILE \mill\msmf\machine.tmp
@LINE
<#DCOMP> <D#DOFF> <#MOV> <X#XPOS> <Y#YPOS> <Z#ZPOS> <F#FEED>

NC CODE FILE \mill\mdata\filename
G41      D3      G01      X5.0      Y0.25      Z0.0      F10.0

```

The template word `#MOV` equals 1 when it outputs the information for a line to the template file. Each time SmartCAM encounters a line element in the process model file, it processes the `@LINE` section, updating template words and making them available for output.

When **#MOV** is encountered in the **@LINE** section, the string entered in machine file question 85 (typically, **G01**) is output.

NC Code File

The NC code file contains the machine code required to produce the part you have modeled. NC code files typically have no extension. You can output an NC code file to any directory.

SmartCAM brings together information from all the previous file types to produce an NC code file. You can view or easily change this file with SmartCAM's Edit Plus text editor or another ASCII text editor. See the [SmartCAM Edit Plus User Guide](#) for details about using the utility.

SmartCAM also includes a communications utility that enables you to transmit the code directly to your tape punch, DNC network, or machine without leaving the SmartCAM environment. See the [SmartCAM Communicate User Guide](#) for details about using the utility.

How SmartCAM Generates Code

This section discusses how SmartCAM uses the job operations setup, process model, template, and machine files to generate code. The way SmartCAM shares information between the files is important, because it is central to the intelligence and utility built into code generators. You need to make sure that these files are correct in order for accurate code to be generated. That's why SmartCAM's code generation is an information-formatting process and not a programming language.

Interactions between the process model and the machine files are more complex than those between the process model and the job operations setup file. Settings in the machine file determine how code is output. They help determine what you will see when you view the tool path in the process model.

When SmartCAM writes the NC code file, it reads the values of the template words and uses the template file as the pattern to follow when generating code. SmartCAM takes the first element in the database and generates code for it completely before moving to the next element. This is called *linear processing*. SmartCAM continues processing elements one at a time until it reaches the end of the file. The examples in this section were selected to help you build a conceptual model of a code generator. Although the examples are simplified, the ideas presented are similar, even in instances where complex requirements must be met.

Look at figure 1-4 to get a conceptual overview of code generation. SmartCAM and the job operations setup file direct the process through the CNC process model you create on the screen. The job operations setup file and process model information specify how you want to machine a part. The machine file defines the file formats, machine code, and canned cycle information. Template files output the machine file and database information in the order your machine and controller recognize. It is not a lengthy process; it happens almost instantaneously.

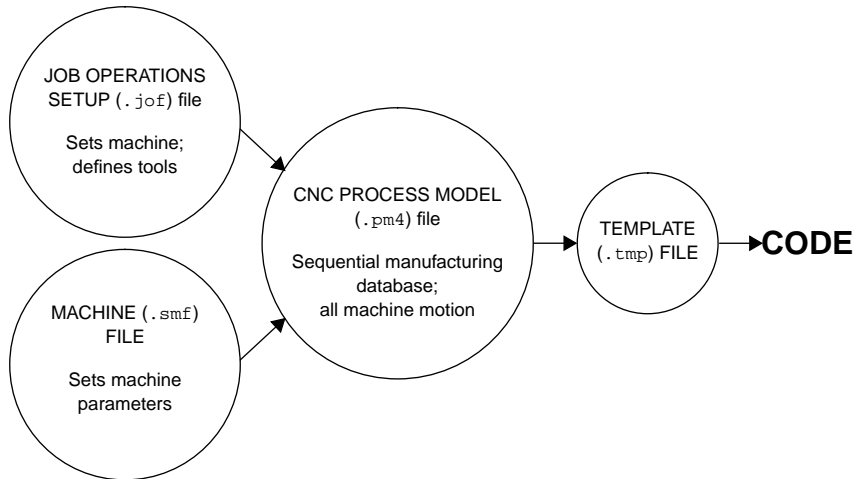


Figure 1-4
The template file determines the output sequence and format for the code file.

Show Path and the Code modeling tool can both graphically simulate the machining of the part. The primary difference between the two is that Code generates machine code while Show Path does not. When you select Process—Code, the template file acts as a filter to generate the NC code.

☞ For more information about Show Path and Code, see the reference manual for your application.

Using the Process Model Database

When you build a CNC process model, you are working with a graphic representation of the database for that model. You refine the model until it meets the design and tool path requirements. Once you are satisfied with the model, you select Process—Code and SmartCAM produces NC code for your machine tool to cut the part.

SmartCAM's database is usually hidden from view. However, you can view this database by pressing **F7** or by selecting Element Data from the Utilities menu. SmartCAM lists all the elements in the database as well as any associated properties, such as offset direction and clearance value. These are displayed in sequential order and will be machined in the same order. (For more information, see Element Data in the online help or reference manual for your application.)

☞ If your part information comes directly from a CAD system, use a CAM Connection to transfer this data. A CAM Connection converts your CAD database into a SmartCAM database. SmartCAM takes the information from the CAD drawing and, optionally, resequences it into an appropriate order for machining processes. (You can easily rearrange tool path and properties within SmartCAM too.)

Using the Tool Path

When generating code for a part, SmartCAM must convert the information about each element in the process model file into NC code to run your machine. Before this can happen, SmartCAM must do the following:

- Calculate the actual tool path with information from the database.
- Send the tool path information to the files used to generate code.

To simulate the tool path correctly, SmartCAM uses information from the job operations setup file. The actual tool path for the tool is calculated using tool information from the job operations setup file. You can view database information that SmartCAM will use to generate code when you use Show Path on a process model.

Using the Process Model and Job Operations Setup Files

The job operations setup file contains information about the steps used to cut your part. The Process Model file associates each tool path element to a step in the job operations setup file. Since each step contains the information about a specific tool, this ensures that the correct tooling information is available for the template file.

Each tool used in your process model is identified by a number. When SmartCAM generates code for an element, it passes the tool number to the job operations setup file. The tool number value is carried with the template word **#TOOL**. Once the job operations setup file recognizes which tool is active, it passes all information about that tool to the template file.

The job operations setup file also contains information such as the measurement system (inch or metric), machine type, and the name of the `.smf` file. The `.smf` file sets the defaults and assigns the machining parameters for your machine.

Using Template Words

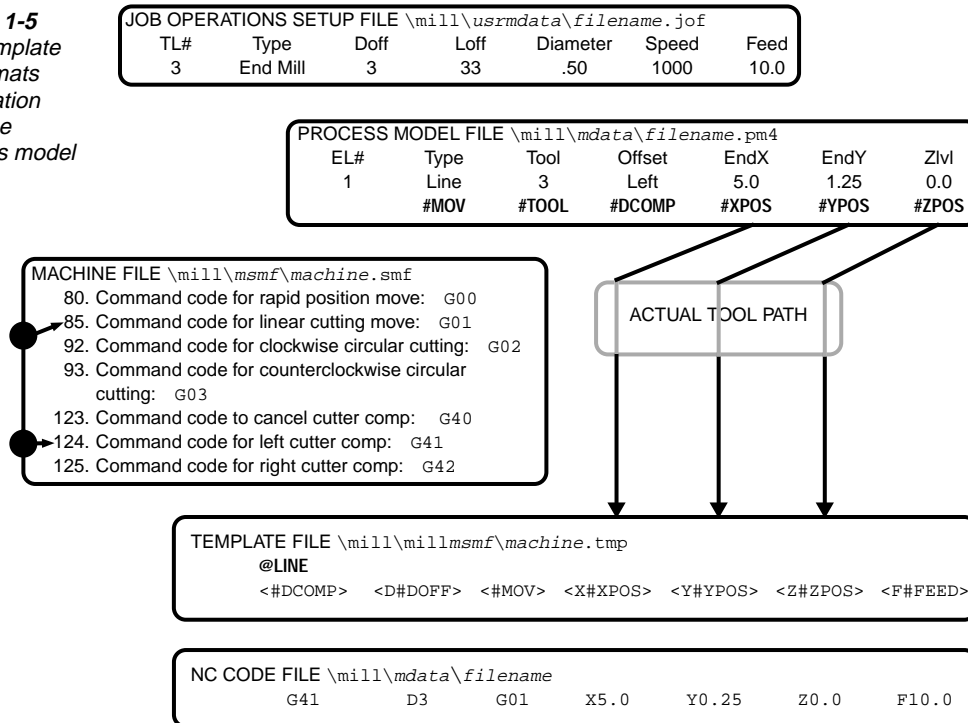
SmartCAM uses many template words to transfer information to the template file. You can see template words in any template file. All template words begin with the **#** symbol. For example, **#XPOS** is a template word that contains information about the current X position.

After SmartCAM calculates tool path information (with the help of the job operations setup file), it transfers that information through the machine file, where it is converted into a format your machine understands. Next it passes the correctly sequenced and formatted information through a template file for conversion to machine code. Then SmartCAM saves it in an ASCII file, ready to send to your tape punch or machine at your command.

Transferring Tool Path Information to the Template File

Figure 1-5 shows how SmartCAM transfers tool path information to a template file. Here, the template file section @LINE outputs code only for lines.

Figure 1-5
The template file formats information from the process model file.



There are template words in SmartCAM for any process model file element. The following list and figure 1-6 explain the use of some of those template words:

Information	Template Word
End X	#XPOS
End Y	#YPOS
End Z	#ZPOS
Line length	#LNLEN
Line angle	#LNANG
X Offset Vector Amount	#XOV
Y Offset Vector Amount	#YOV

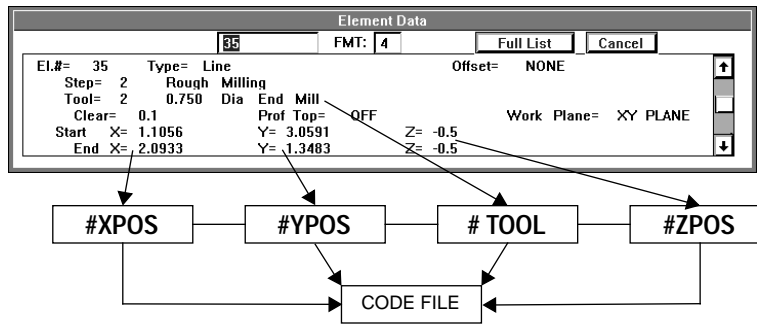


Figure 1-6
The template gets information from the database.

Outputting Information through the Machine File

The machine file has a list of questions about your machine and controller that are necessary to produce NC code. For example, the process model database might know that an element is a line, but it does not know the linear cutting code for your machine tool. This information is retrieved from question 85 of the machine file.

When SmartCAM generates code for a process model element, it transfers a numeric value for the appropriate template word through the machine file to the template file. The switches in the machine file direct the appropriate value to the template file.

An example is the template word **#MOV**, which contains information about the type of element SmartCAM is processing. **#MOV** is a four-way switch. Its value depends on the type of element SmartCAM is processing. **#MOV** can have one of the four following values:

#MOV Value	Question No.	Meaning	Typical Output
0	80	Rapid positioning move	G00
1	85	Linear cutting move	G01
2	92	Clockwise cutting move	G02
3	93	Counterclockwise cutting move	G03

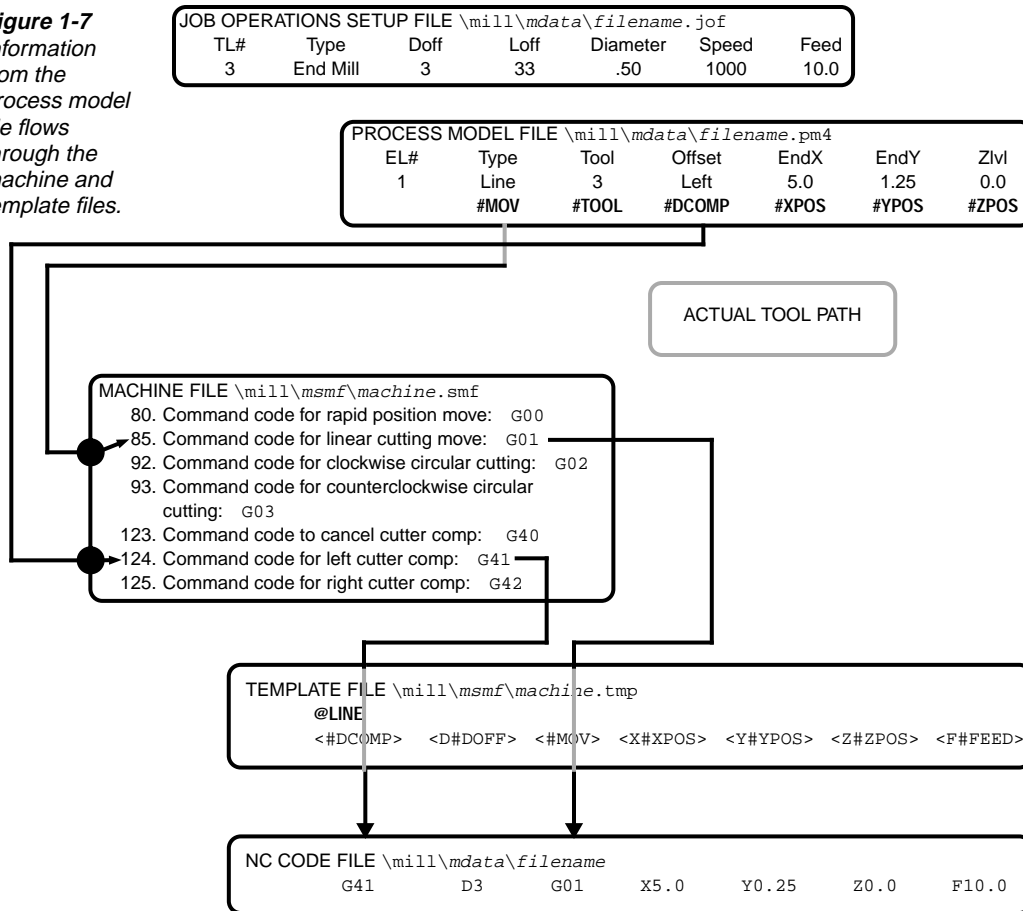
For a line, SmartCAM sets the value of **#MOV** to 1 and passes this setting to the machine file. The machine file sees the value of **#MOV** and translates that into the linear cutting move code (G01), which it sends to the template file.

SmartCAM uses the template word #DCOMP in a similar way. The following table shows how the template word #DCOMP acts like a switch:

#DCOMP Value	SMF Question	Meaning	Typical Output
0	123	No cutter comp	G40
1	124	Cutter comp left	G41
2	125	Cutter comp right	G42

Figure 1-7 illustrates information flow from the process model through the machine and template files. When testing for a current condition in the template file, examine the value of these switches. You can easily access the machine file settings and make any necessary changes.

Figure 1-7
Information from the process model file flows through the machine and template files.



Sending Information to the Template File

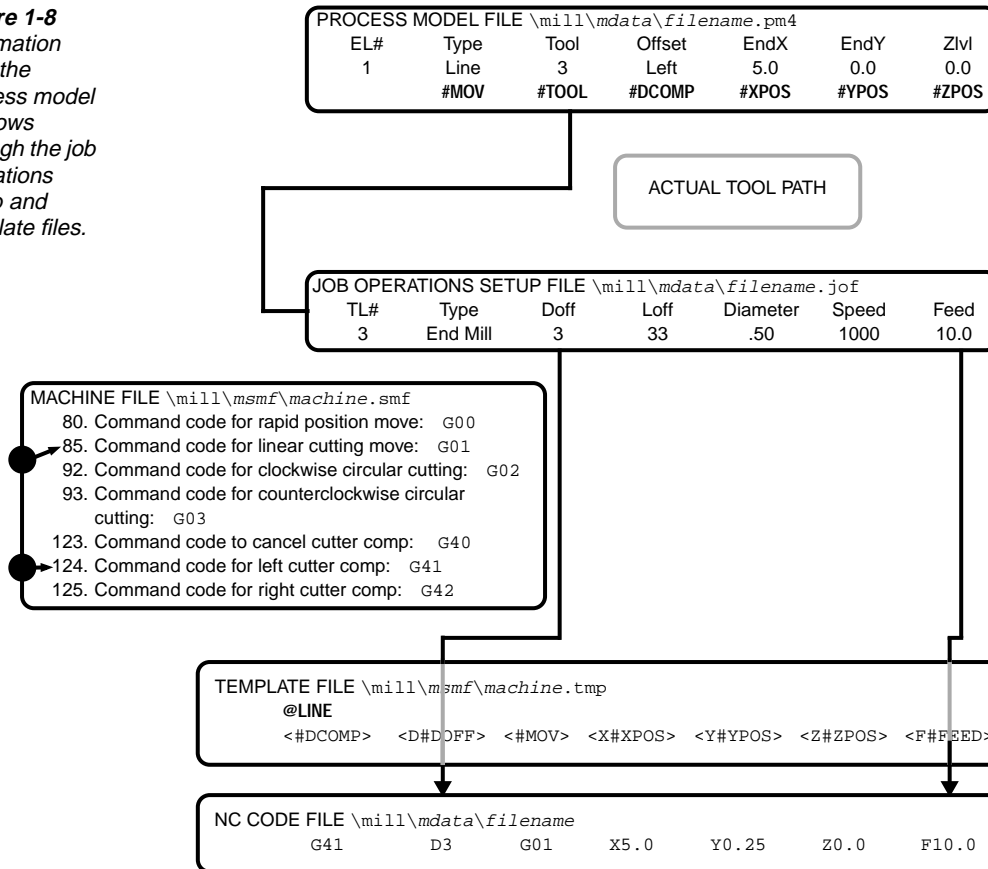
SmartCAM sends several types of information from the machine file to the template file. As in the #MOV example, most of this information depends on what machining operation SmartCAM is processing. Other types of information in the machine file are block number formats, positioning-move formats, and some default file names.

The job operations setup file also uses template words to send information to the template file. The following are some template words that can be sent from the job operations setup file to the template file:

Template Word	Description
#TOOL	Current tool number
#TLDIA	Tool diameter
#TLEN	Tool length
#TWID	Tool width
#FEED	Tool feed
#TLOP	Tool operation (based on the tool type in the job operations setup file)

When coding a file, the job operations setup file makes all the information for the active tool available to the template file. This information includes the tool number and the offset registers assigned to the tool. Figure 1-8 shows how this information flows to the template file and to the NC code file.

Figure 1-8
Information from the process model file flows through the job operations setup and template files.



Modifying a Code Generator

SmartCAM code generators are very flexible. They can generate code in almost any format—not just G and M codes using the RS-274D standard, but also conversational language code, special commands such as event types, and tab-sequential formats. You can create a code generator for your machine if you can create a process model file that accurately describes your part's tool path and manually write a program for your machine.

You must have a thorough knowledge of your machine and controller to modify or write a custom code generator. If you have a new machine or have been programming with another system, you may need to familiarize yourself with a new set of operational commands and parameters.

If you already have a code generator that is producing code close to what you need, it should need only minor changes. Before you start any modifications, identify and document any changes in the code that are required. You may need to use a different numeric format, change the way the code generator produces block numbers, or add or rearrange codes at a tool change.

Knowledge of SmartCAM Required

To modify a code generator, you need to know the following:

- How to work with the files described in “SmartCAM Files Required for Code Generation,” on page 1-3
- How to use the process model file and job operations setup file
- How to read and edit the machine and template files

Modification Guidelines

With a basic understanding of your machine tool and SmartCAM, you should be able to make level-1 modifications (see page 1-2). The following is an abbreviated list of the steps related to modifying code generators:

1. Find a code generator that produces code close to what you want.
2. Create a simple test part (or series of parts to check specific functions) to check the code generator and any changes you make to the code generator.
3. Clearly identify what sections of code need changes. Note the difference between what specific codes are and what they should be.
4. For each identified change in step 3 check whether you must make appropriate changes in the machine (.smf) file.
5. Test the result of each change immediately after you make it. Continue testing until you finish making all the changes to the machine file.
6. Make appropriate changes in the template (.tmp) file, testing the result of each change as you make it.
7. Repeat steps 3 through 6 until the code generator works correctly.

The following sections detail some important considerations for you to keep in mind when performing these steps.

Using An Existing Code Generator

The SmartCAM library of code generators supports most popular machines and controllers. Your dealer can often provide a code generator that will produce NC code compatible with your machine's requirements with no modifications or only minor ones. Documentation standards and benchmark parts are also available from your dealer. The benchmark parts demonstrate how the code generators were proven. Each benchmark part file includes a part drawing, a process model file, and a job operations setup file.

There may be some differences in programming style between the way you write code and how these sample code generators produce code. Generally, you can alter one of these code generators to match your requirements.

Creating a Test File

You may want to use benchmark part files to test your code generator. If you use your own part file to verify your code generator, use a simple part. The file should enable you to check the code easily when you make a change to the code generator.


If your code generator needs only a few changes, you may want to create simple files to test the functions that need changing. For example, if you need to change coding for arcs, build a short test file that includes only arcs to see the results of any change quickly.

Identifying Changes to Make in Code

Identify what part of the code you want to change before making changes. Ask yourself the following questions:

- Do I need to modify tool changes only?
- Do I need to change certain numeric formats only?
- Do I need to change the way the code generates block numbers?

Make a printout of a code file you've written manually for the test file, marking any changes. Make one change to the code generator and then test it before proceeding to the next change.

 You can have SmartCAM generate an error file when it generates code. See chapter 7 for information.

Changing the Machine File

When you know which changes to make, alter the code generator to reflect the change. Start with the machine (.smf) file. The machine file contains questions affecting final code output. You may need to make changes only to the machine file to make the code generator produce the correct code.

Some people who spend hours working on a template file discover they could have answered a single question in the machine file to make the same change. For instance, all machine file questions relating to arcs are in one section. If you need to make changes for how arc code is generated, you need to answer only the questions in that section.

Just as you would work on only one code issue at a time, change only one machine file question at a time. Run your test file and check the results after each change. Continue this process until you have the correct output.

Changing the Template File

If you change the machine file and it doesn't produce the results you want, work on the template file. When you work on the template file, make only one change at a time and then test the change to make sure it works. When you verify that a change works, move to the next change.

Practice: Making Changes for Arc Code

This section describes how to make a simple change to a code generator. The code generator for this example comes with SmartCAM in the `smf` subdirectory under the following file names:

- `m_fanuc.tmp`
- `m_fanuc.smf`

This code generator produces code for a controller similar to a Fanuc 11M. The following discussion describes how to change the way a code generator produces code for an arc.

The Problem

Assume the sample code generator produces code exactly the way you want for everything except arcs. To test any changes you make, create a sample process model file that contains only one arc, using the settings shown in figure 1-9.

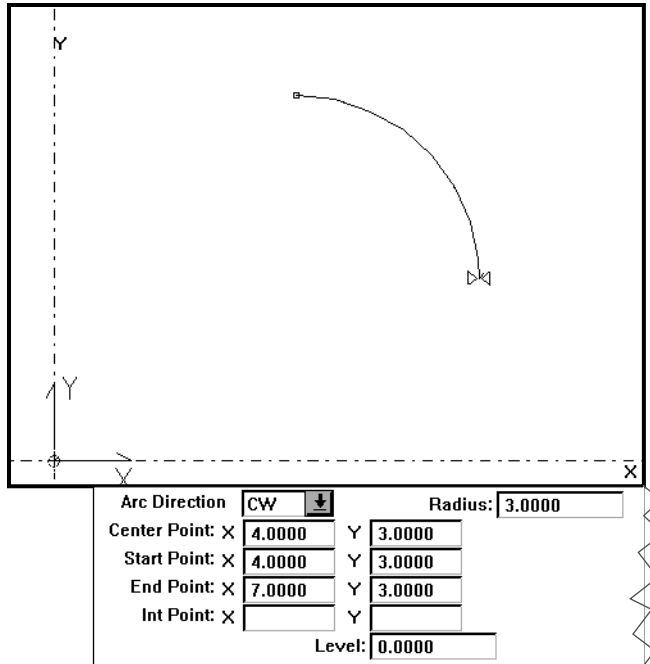


Figure 1-9
Select File—
New before
creating the
arc.

The code generator writes code for this arc as follows:

```
G02 X7.0 Y3.0 I0.0 J-3.0 F0.014
```

where

G02 is a clockwise cutting code (G03 for counterclockwise);

X7.0 is the end X of the arc;

Y3.0 is the end Y of the arc;

I0.0 is the incremental X distance from the start to the center of the arc;

J0.0 is the incremental Y distance from start to the center of the arc; and

F0.014 is the feed rate of the move

Let's suppose the machine requires code for this arc as follows:

```
G22 X7.0 Y3.0 I4.0 J3.0 R3.0 F30
```

where:

G22 is the clockwise cutting code (G23 for counterclockwise);
 X7.0 is the end X of the arc;
 Y3.0 is the end Y of the arc;
 I4.0 is the absolute position of the arc center X;
 J3.0 is the absolute position of the arc center Y;
 R3.0 is the arc radius; and
 F30 is the feed rate of the move

Changing the Machine File

To access the `m_fanuc.smf` machine file, follow these steps:

1. Access the Machine Define utility.
2. Select **File—Open SMF**.
3. Select `c:\mill\smf_dir\m_fanuc.smf` (where *smf_dir* is the machine file subdirectory for your application).
4. Select **OK**.

☞ For complete information on using Machine Define, see chapter 2.

Since you need to change only the way the code generator writes code for arcs, you need to look at machine file questions 90–109. The following are the three questions that require new answers:

92. Command code for clockwise circular cutting:

Example:

G02 Standard code

Modification:

Enter the machine clockwise circular cutting code: G22

93. Command code for counterclockwise circular cutting:

Example:

G03 Standard code

Modification:

Enter the machine counterclockwise circular cutting code: G23

96. Arc center location mode:

Choices:

- <0> Always incremental distance to center
- <1> Absolute location when in absolute positioning mode (G90)
- <2> Always absolute location regardless of positioning mode

Modification:

Select <2>, because this machine requires the arc center location always to be the absolute location, regardless of the positioning mode you are in.

After you make these changes, save the file as `test.smf`.

Changing the Template File

Since you are changing only the way the code generator produces code for arcs, you need to change only one section of the template file. To access the `m_fanuc.tmp` file, follow these steps:

1. Access SmartCAM's Edit Plus utility or another ASCII text editor.
2. Select **File—Open**.
3. Select `c:\mill\msmf\m_fanuc.tmp` (where *smf_dir* is the machine file subdirectory for a milling application).
4. Select **OK**.

☞ For complete information on using Edit Plus, see the [SmartCAM Edit Plus User Guide](#).

Look at the **@ARC** section in the `m_fanuc.tmp` file, which should appear as follows:

```
< #MOV>< X#XPOS>< Y#YPOS>< I#XCTR>< J#YCTR>< F#FEED>
```

The following template words are relevant to the example:

#MOV	Move preparatory code
#XPOS & #YPOS	End position of arc
#XCTR & #YCTR	Center position of arc
#ARAD	Arc radius

You have already set most of these items correctly by changing the three questions in the machine file. The endpoint of the arc is set by **#XPOS** and **#YPOS**. You don't need to change that. The template word **#MOV** outputs the circular cutting code (G22 or G23). This is set with machine file questions 92 and 93. The template file is unchanged. Question 96 sets the arc center coordinates output by **#XCTR** and **#YCTR**, so these remain unchanged.

The feed rate set by #FEED also remains unchanged. However, there is no template word in the @ARC section to output the radius of an arc. Looking at our list of template words relating to an arc, we see that #ARAD will output the arc radius. Add this word to the @ARC section as follows:

```
<#MOV>< X#XPOS>< Y#YPOS>< I#XCTR>< J#YCTR>< R#ARAD>< F#FEED>
```

We've added a space and a literal, R, in front of the template word #ARAD. This causes the arc radius output to be preceded by a space and the letter R (for example, R3). Pay attention to where this template word is placed in the line. It should appear in the order that code is to be produced.

Running Your Test File

Run the test file to verify the changes you just made. Because the @ARC section outputs code for clockwise and counterclockwise arcs, your test file should contain both types of arcs. If one of the output words is not what you expect, check your machine file setting. Then check to see that the content and spelling of your template words are correct. If you are still having problems, consult your reseller or CAMAX Manufacturing Technologies Technical Services for more information.

Using Machine Define

Introduction

Before you can generate code for a machine, you need to specify certain information about the machine. This information is contained in an ASCII text file called a *machine file*. These are also called *.smf files*, because they have an *.smf* extension.

Your SmartCAM application comes with *.smf* files for several common machines. Depending on your preferences and the requirements of your part, you may need to customize an *.smf* file. The Machine Define module for Windows provides a simple way for you to make changes to your machine file.

Each machine file contains “answers” to “questions” about a specific machine. How you answer those questions tells SmartCAM how to format code for your machine. Using Machine Define, you can open an *.smf* file, find a question whose answer you want to change, and edit the setting. After making all the changes you want, save the machine file and you are ready to generate code for that machine.

Files Included with the Machine Define Utility

The table below identifies and describes the types of files that are included with Machine Define.

File	Description
wmdef.exe	Executable file. This file runs the Machine Define program.
*.cat	Category files. These files identify the questions that are in each category.
*.des	Description file. This file contains the text of the questions and their associated explanations.

File	Description
*.smf	Machine files. These files contain the answers to the questions. Each machine has its own file.
*.trn	Transformation files. These files map the questions to appear in the correct numerical order in the Machine Define window with their associated answers.

All .smf files are stored in application-specific subdirectories of your sm8 directory (see page 1-5). All other files are located in the sm8 directory.

File Versions

SmartCAM supports two versions of machine files: v4.0 and v5.0. Version-4 files are used by software released before December 1993. Version-5 files are used by software released in December 1993 or later.

You can update a v4.0 file to v5.0 or change a v5.0 file to v4.0 by selecting the appropriate Save As option in the File menu (see “Saving Your Changes,” beginning on page 2-7). When converting a file to a different version, use a different name or be sure you have a good backup copy of the original file.

Starting Machine Define

If you have already started Windows on your computer, you can start Machine Define from the Program Manager, the File Manager, or select Start—Programs—SmartCAM—Machine Define in Windows 95.

Before you can start Machine Define, it must be installed properly on your system. You may have installed Machine Define during the installation of your SmartCAM application for Microsoft Windows. If not, you will need to install it now. Please see the *Installation Guide for All SmartCAM Products*.

Starting from Program Manager

To start Machine Define using the program icon:

1. Open the Microsoft Windows Program Manager window, if it is not already open.
2. Open the SmartCAM group window, if it is not already open.
3. Double-click on the Machine Define icon (see figure 2-1).

Figure 2-1
The Machine Define icon appears in your SmartCAM group.



To start Machine Define using the Run dialog box:

1. Open the Microsoft Windows Program Manager window, if it is not already open.
2. Select **File—Run** from the Program Manager menu to open the Run dialog box.
3. In the **Command Line** input field, type the full path and filename (including the extension) of the Machine Define executable file.
4. To open a machine file at the same time, type a space and **d:\SmartCAM Path\filename.smf**, where
 - d:** is the drive on which the .smf file is located
 - SmartCAM Path** is the path to the directory where your .smf file is located; and
 - filename** is the file you want to open.
5. Select **OK**.

Starting from File Manager

To open Machine Define using the executable file icon:

1. Open a File Manager window for the directory that contains the Machine Define executable file (`wmdef.exe`).
2. Double-click on the icon for the file.

☞ You can also start Machine Define by double-clicking on the icon for a machine file that you have associated with the executable file. (For information about associating files, see your *Microsoft Windows User's Guide*.)

The Machine Define Window

The Machine Define application window uses the same border and window elements as other Microsoft Windows applications (see figure 2-2). If you do not know how to use basic window elements, how to move around in a window, or how to make selections in a menu, see your *Microsoft Windows User's Guide*.

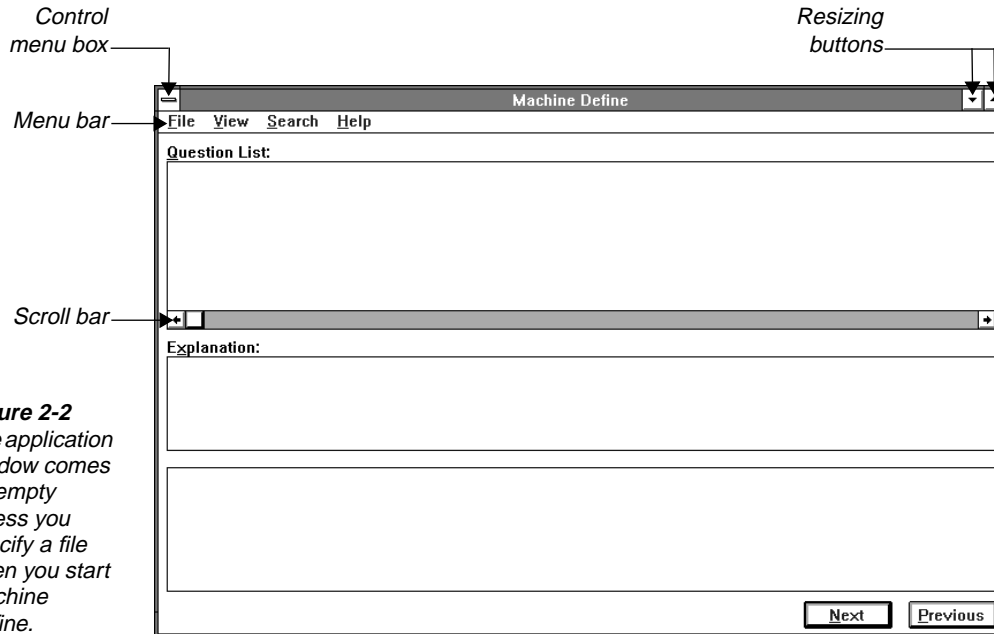


Figure 2-2
The application window comes up empty unless you specify a file when you start Machine Define.

The Machine Define menu has four options: File, View, Search, and Help. When you select an option, a menu for that option drops down from the menu bar.

The Help option works just like Help in other Windows applications. If you are not familiar with Windows Help, see your user's guide. The other options are covered in detail in the rest of this chapter.

The Machine Define window also has two action buttons: Next and Previous. After opening a machine file, you can use these buttons to move through the question list. Pressing Next moves you down the list, and pressing Previous moves you back up. Because Next is the default, you can press **ENTER** to select it.

Components of a Machine File

Inside the Machine Define window are three boxes: the question list box, the explanation box, and the selection box. These boxes are empty until you open a machine file. Then the information in the file displays in the boxes (see figure 2-3).

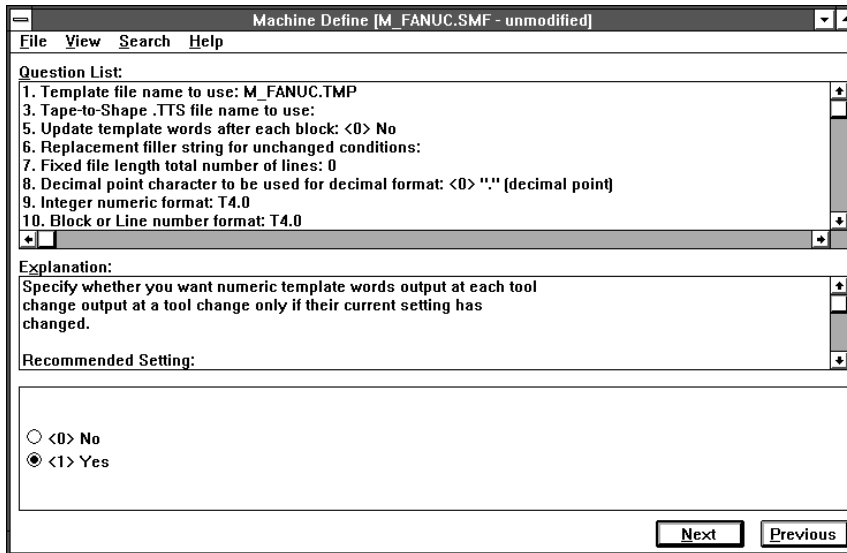


Figure 2-3
 When you open a machine file, the window boxes show information about the machine.

The Question List Box

The question list box contains the questions and current settings for the selected machine. Each question ends with a colon (:). The information following the colon is the current setting for that question. When you open a machine file, the first question in the list is highlighted.

The Explanation Box

The explanation box provides instructions for setting the highlighted question. These instructions may include a list of applicable machine types, an explanation of the question and what choices you have for settings, details on what effect each setting will have, recommendations for settings, references to related settings, and other information.

The Selection Box

The selection box provides either an input field for entering text (see figure 2-4) or buttons for choosing a specific option (see figure 2-5).

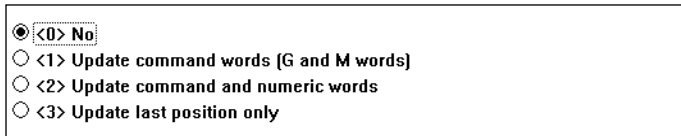
Figure 2-4
 This type of selection box lets you type information.

Enter:

Each selection box shows you the default or current setting for the highlighted question.

Figure 2-5

This type of selection box lets you pick an option.

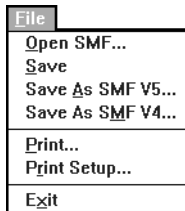


Using The File Menu

The File menu provides options for opening, printing, and saving machine files (see figure 2-6). Only the Open SMF, Print Setup, and Exit options are available if no file is loaded.

Figure 2-6

The File menu drops down from the menu bar when you select it.

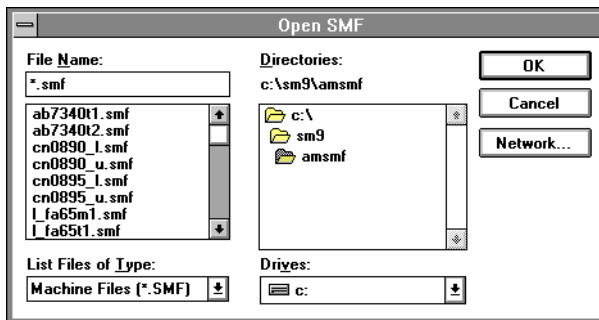


Opening a Machine File

1. Select **File—Open SMF** to bring up the Open SMF dialog box (see figure 2-7).

Figure 2-7

You can either type a path and filename or select them with the mouse.

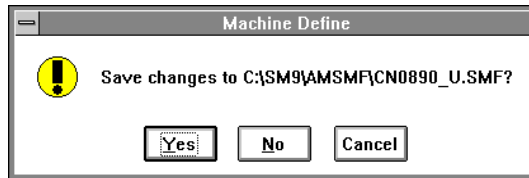


2. Type the name of a machine file (.smf file) in the File Name input field, or select one from the list. If the file you want is not in the current directory, include the full path in the input field, or select the correct directory from the list. (For the locations of the machine files, see the table on page 1-5.)
3. Select **OK** to read in the machine file.

If you previously opened a file and changed any of the settings without saving them, a dialog box will give you the chance to save that file before loading the new one (see figure 2-8).

Figure 2-8

If you try to open a new file without saving the current one, you have a chance to save it.



The information in the file you loaded fills the Machine Define window (as described in “Components of a Machine File,” beginning on page 2-4). The filename displays in the title bar at the top of the window. You can now check and change settings in the machine file.

Saving Your Changes

After changing settings in a machine file, you need to save the .smf file back to disk. You can save the file in its current format (version 4.0 or version 5.0) or change it to a different format. (See page 2-2 for information about the two formats.)

To save the file in its current format:

Select **File—Save** to write the file back to disk after changing settings.

SmartCAM will save your edited file and leave it on the screen.

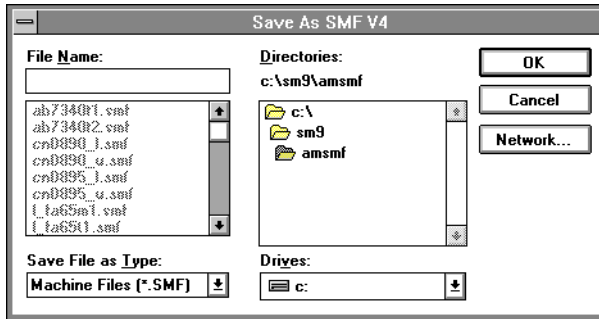
☞ You will not have a chance to save the file with a new name or verify that you want to overwrite the original file on the disk. If you made any changes that you didn’t mean to save, change the settings back and save the file again.

To save the file in a different format:

1. Select **File—Save As SMF V5.**
or
Select **File—Save As SMF V4.**

The Save As dialog box for the version you selected will appear (see figure 2-9). Use this dialog box to select the drive, path, and filename for the document.

Figure 2-9
Before saving a file to a different version, make a backup copy of the original file.



2. Enter the new name for the active document, or select one from the list of filenames. To save the file to a different directory, enter the complete path for the new directory in the File Name field. You can also use the Drives and Directories fields to navigate to a new directory.
3. Select **OK**.

If you are saving a version-4 file as version 5 or vice versa, a message box will appear asking you to confirm that you want to convert the file (see figure 2-10). If you select Yes, the active document will be saved with the name you specified.

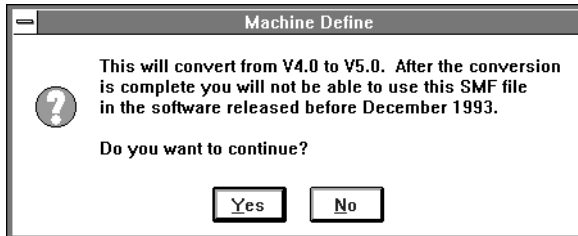
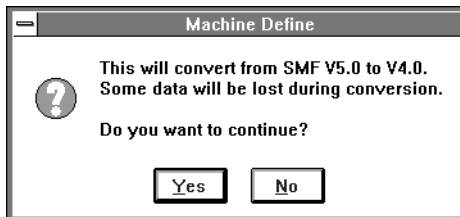
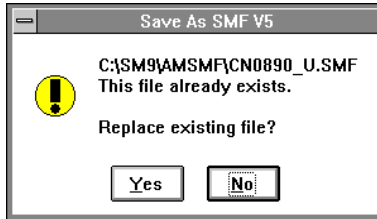


Figure 2-10
Check the dates of your SmartCAM applications before converting a machine file.



If a file already exists with the name you are trying to use, a message box will be displayed (see figure 2-11). To overwrite the file, select Yes. If you do not want to overwrite the file or change the version, select No. You will then be returned to the Save As dialog box.

Figure 2-11
Be sure not to accidentally overwrite an existing file.

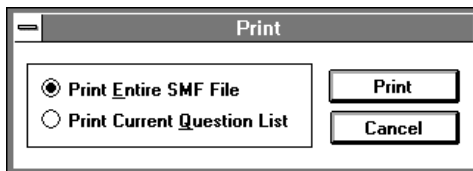


Printing a Machine File

Use the Print dialog box to print either the entire current machine file or only the current question list from the file.

1. Select **File—Print**. The Print dialog box displays on the screen (see figure 2-12).

Figure 2-12
Print either part or all of a machine file.

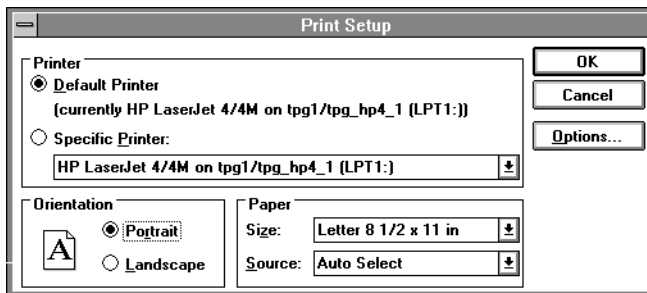


2. Indicate whether you want to print the question list for the entire machine file or only the current question list.
3. Select the **Print** button.

Changing the Printer Setup

Use the Print Setup dialog box (see figure 2-13) to select a printer or to change the configuration for a printer.

Figure 2-13
Use the Print Setup dialog box to configure your print job.



To select the default printer:

1. Select **File—Print Setup**.
2. Select **Default Printer** if it is not already selected.

3. Select the orientation, paper size, and paper source if you do not want to use the defaults.
4. Select the **Options** button if you want to change any of the settings for the default printer.
5. Select **OK**.

To print on a different printer:

1. Select **File—Print Setup**.
2. Select **Specific Printer** if it is not already selected.
3. Select the printer you want from the drop-down list.
If the printer you want is not in the list, select the **Options** button to add a printer.
4. Select the orientation, paper size, and paper source if you do not want to use the defaults.
5. Select the **Options** button if you want to change any of the settings for the selected printer.
6. Select **OK**.

Closing the Machine Define Module

Select **File—Exit** to close Machine Define.

If you made any changes to the file but did not save it before trying to exit, a message box will give you a chance to save the file before exiting Machine Define.

When you close Machine Define, the following current configuration information is saved for the next time you open it:

- Window position (when opened)
- Window size (when opened)
- Open maximized or windowed
- Open as an icon or a window

Using the View Menu

When you open a machine file, the questions appear in numerical order in the question list box. This is the “item list” view. You can change the list so that the questions are arranged in categories. Items within categories are arranged in numerical order.

Use the View menu (see figure 2-14) to switch between the item list and the category list. A check mark in the menu indicates which list is currently selected. Another option in the view menu lets you select and display only the category or categories you want to work with.

Figure 2-14

The **View** menu lets you select specific categories of questions.



☞ View menu options are available only when a file is open.

Viewing the Item List

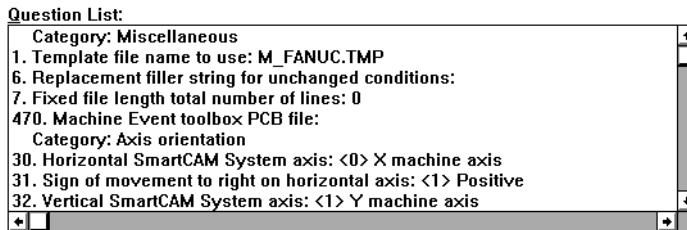
Select **View—Item List** to display all the questions in numerical order (as shown in figure 2-3).

Viewing the Category List

Select **View—Category List** to display the questions arranged by category (see figure 2-15).

Figure 2-15

View or edit an entire category of questions by using the category list.



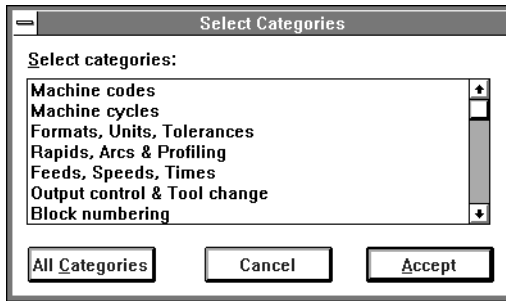
Selecting Categories for Viewing

Use the Select Categories option to choose one or more specific categories to view or to change the question list from selected categories to the full category list. The keyboard shortcut for the Select Categories option is **CTRL+C**. (For information on using keyboard shortcuts, see “Using the Keyboard,” beginning on page 2-15.)

To view specific categories:

1. Select **View—Select Categories**.
2. Select a category from the list in the Select Categories dialog box (see figure 2-16). To select more than one category one at a time, hold down the **CONTROL** key while making your selections. To select a range of consecutive categories, hold down the **SHIFT** key while moving through the list.

Figure 2-16
The *Select Categories* dialog box lets you choose specific categories of questions to appear in the list.



3. Select **Accept**.

To view all categories:

1. Select **View—Select Categories**.
2. Select **All Categories**.

Using the Search Menu

The Search menu provides two shortcuts for finding specific questions in a machine file. You can search for a specific question number, or you can search for questions that contain a given text string. (A text string is specific combination of keyboard characters. A string may include letters, numbers, punctuation marks, and spaces.) The Search menu is shown in figure 2-17.

Figure 2-17
Use the *Search* menu to find specific questions or text.



☞ Options in the Search menu are available only when a file is open.

To search for text:

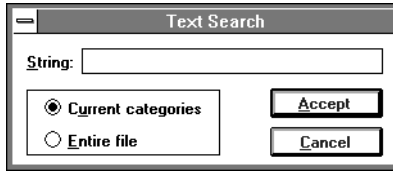
1. Select **Search—Text**.

or

Use the keyboard shortcut, **CTRL+T**. (For information on using keyboard shortcuts, see “Using the Keyboard,” beginning on page 2-15.)

2. Enter the string you want to find in the input field (see figure 2-18). You can use uppercase letters, lowercase letters, or a combination—the text search is not case sensitive.

Figure 2-18
The Current categories option is available only if you selected the Category List option in the View menu.



3. If your current View option is Item List, the search will automatically look through all the questions in the machine file. If your current View option is Category List, specify whether you want to search through the files in the current categories or through all the files. If you select Entire File, the category list will switch back to an item list.

4. Select **Accept**.

☞ When you search for text, the search begins at the current question. To search through the entire machine file, start the search at the beginning of the question list.

Text Search Limitations

- A text search will not find text in the current settings. For example, to find the following question:
 9. Integer numeric format: D3.4
 you could not search for “D3.4”. You would have to search for text that displays before the colon (:).
- A text search does not perform “full-word” searches. For example, if you search for “line,” it will also find “spline,” “lines,” and “linear.”
- A text search includes question numbers. For example, if you search for “4”, it will find questions 4, 14, 24, 34, 40, 41, 42, and so on. If you want to find a specific question number, use Goto Question.

To search for a specific question number:

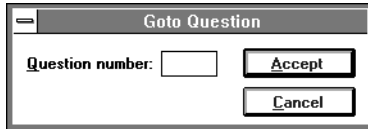
1. Select **Search—Goto Question**.

or

Use the keyboard shortcut, **CTRL+Q**. (For information on using keyboard shortcuts, see “Using the Keyboard,” beginning on page 2-15.)

2. Enter the number of the question you want to find in the input field (see figure 2-19).

Figure 2-19
If you know a question number, you can find it quickly.



3. Select **Accept**.

- ☞ You can use Goto Question anywhere in the machine file. If your current View option is Category List, you cannot go to a question that isn't in the selected categories.

Editing a Machine File

Once you load an `.smf` file into the Machine Define window, you can search for or move to any question and change its setting. The current question will be highlighted in the question list box. Information about the question and its possible settings will appear in the explanation box. The selection box provides either an input field for entering text or selector switches for choosing among specific choices.

To change a question's setting:

1. Highlight the question you want to edit.

An explanation of the question displays in the explanation box, and the current setting for the question is indicated.

2. Type the new setting in the input field.

or

Select the new setting from among the choices.

When you enter text in the input field, the current setting is deleted when you begin typing the new information. To change only certain characters in the setting, move the cursor with the **HOME**, **END**, or arrow keys before typing. Do not use the **BACKSPACE** key. For example, to change a setting of D3.4 to D3.3, either type `D3.3` or press the left arrow key once, press **DELETE** to delete the 4, and type 3.

To select from a list of choices, you can make your selection with the mouse or by moving to it with the **UP ARROW** or **DOWN ARROW** key.

3. To accept the new setting, move to another question or make a menu selection (other than Help). If you entered text, the text you entered is displayed in the question list as the setting for the question you edited. If you made a selection, the selected text is displayed.

Using the Keyboard

You can use the **ALT** key with letter keys to open menus or move around in the question list or explanation box. You can also use the **CTRL** key with letter keys to bring up certain dialog boxes without having to go through the menu. You can also use **TAB** or **SHIFT+TAB** to switch among the question list, explanation box, selection box, Next button, and Previous button.

- ☞ To use a key combination, hold down the **ALT** or **CTRL** key and then press the second key in the combination. Although letter keys are shown as uppercase letters, you do not need to press **SHIFT**.

To make menu selections:

Press **ALT** plus the key for the underlined letter in the menu name. When a menu is open, you can select an item from the menu by pressing the key for the underlined letter in the menu item. For example, press **ALT+F** to open the File menu, and then press **S** to select Save.

To open a dialog box:

Use the following key combinations to go directly to a dialog box:

Key Combination	Dialog Box
CTRL+C	Select Categories
CTRL+T	Text Search
CTRL+Q	Goto Question

To move through the question list or explanation box:

Press **ALT+Q** to move the cursor into the question list or **ALT+X** to move it into the explanation box. Then use the following keys to move around:

Key	Movement
UP ARROW	Move up one line
DOWN ARROW	Move down one line
HOME	Move to the beginning of the list or explanation
END	Move to the end of the list or explanation
PG UP	Move up one panel at a time
PG DN	Move down one panel at a time

- ☞ When you scroll through the question list, the currently highlighted question remains selected. To select another question in the list, you must click on it with the mouse or move to it using the Next or Previous button in the Machine Define window.

Error Messages

Machine Define may present the following error messages or provide while you are working:

***filename.smf*. Cannot find this file. Please verify that the correct path and filename are given.**

Problem: The file you tried to open is not in the current directory or doesn't exist. You may have typed the filename incorrectly.

Solution: Select OK to close the message window. Then select the correct directory and type the *.smf* filename correctly.

***filename.smf* is an invalid Machine File.**

Problem: The file you tried to open is not usable in Machine Define. It may be corrupt, or it may not actually be a machine file.

Solution: Select OK to close the message window. Then select another *.smf* file.

Cannot find *filename.ext*.

Problem: When trying to open an *.smf* file, Machine Define could not locate a support file, such as a *.des* or *.trn* file.

Solution: Select OK to close the message window. Then use Edit Plus or another text editor to look at your *smartcam.ini* file (which should be in your *windows* directory). Look for the keyword *SmartCAMPPath*. Make sure the drive and directory it indicates is the correct location of your Machine Define support files. If the keyword does not appear in the *smartcam.ini* file, insert it in the "Defaults" section in the following format:

```
SmartCAMPPath=drive:\path\directory
```

Then save the file, exit Windows, and restart Machine Define.

☞ See the [SmartCAM Edit Plus User Guide](#) for details about using the utility.

Category configuration file *filename.cat* not found. No categories loaded.

Problem: The *.smf* file you loaded does not have a corresponding categories file.

Solution: Select OK to close the message window.

Save changes to *filename.smf*?

Problem: You tried to leave Machine Define or open a new machine file without saving the changes to the current file.

Solution: Select Yes to save the current file. Select No to open a new file or exit Machine Define without saving your changes. Select Cancel to return to the current file.

Machine File Overview

Introduction

A machine (.smf) file is an ASCII text file containing the answers to questions that tell SmartCAM how to format code for your machine. All .smf files are stored in application-specific subdirectories of the SmartCAM Path directory (see page 1-5).

A SmartCAM utility called Machine Define enables you to view and edit .smf files. (For complete information about Machine Define, see chapter 2.) To create a new machine file, you can edit an existing machine file and save it with a different file name. There are four types of machine files to choose from:

- Mill machine files set parameters for 2-D and 3-D mills, 2-axis wire EDMs, routers, and milling functions of mill/turn machines.
- Lathe machine files set parameters for lathes.
- Punch machine files set parameters for punch presses and burners.
- EDM machine files set parameters for 4-axis wire EDM machines.

As you work through the questions for the machine file, the following parameters are established:

- The format for G and M codes your machine uses for certain functions.
- The numeric format in which information is output.
- How fixed cycles and cutter compensation outputs are handled.

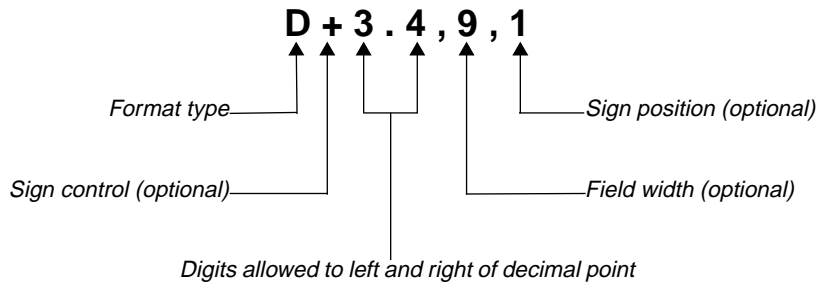
You need to ensure the accuracy of a machine (.smf) file whenever you modify or build a custom code generator. If you change versions of SmartCAM, be sure to compare your previous machine files with the questions in the new version. Each new version of SmartCAM has added capabilities that change the sequence and content of machine define questions.

Numeric Formats

Using SmartCAM's machine file, you can output the precise numeric format your machine requires. For example, you may need to display four places to the right of the decimal point for positioning moves but display feed rates with only one digit to the right of the decimal point. Most numeric formats are set in the .smf file, with some special cases requiring the use of the template file.

Use the numeric format statement shown in figure 3-1 for .smf questions that deal with feed rates, axis positioning, and other numeric output questions.

Figure 3-1
The settings you use for format questions in the machine file determine the output of the code.



Format Type

The following format types are available:

- D The number will include a decimal point.
- F All unfilled fields will be filled with zeros (no decimal point).
- T Trailing zeros will be inserted, and leading zeros will be suppressed (no decimal point).
- L Leading zeros will be inserted, and trailing zeros will be suppressed (no decimal point).
- P Trailing zeros will be inserted, and leading zeros will be suppressed (includes decimal point).
- E Leading zeros and trailing zeros will be inserted (includes decimal point).

Sign Control

Sign control determines whether a sign character should be output. The following options are available:

- (no entry) Only negative numbers will be preceded by a sign.
- + All numbers will be preceded by a sign.
- ! No numbers will be preceded by a sign.

Digits Allowed to Left and Right of Decimal Point

This field specifies the size and accuracy of the numbers. Values are rounded to the appropriate accuracy during formatting.

Field Width

This specifies the width of the numeric field for right justification of the formatted values. This is useful when the decimal points need to line up. SmartCAM fills the left of the field with space characters.

Sign Position

SmartCAM uses this setting with field width to determine the location of the sign character, as counted from the left. If you don't specify a value, SmartCAM positions the sign against the formatted value. A setting of 1 will place the sign character in the first column.

The following table shows the value 10.75 expressed in different formats:

Format	Expression
D3.4	10.75
T3.4	107500
L3.4	01075
F3.4	0107500
P3.4	10.7500
E3.4	010.7500

Generating Code for Subroutines

The use of subroutines depends on your machine's capabilities. SmartCAM addresses a variety of these capabilities with special `.smf` and `.tmp` settings. Review this section if you plan to define and use subroutines in a CNC process model and resultant machine code. See page 4-54 for the related `.smf` questions. The related template sections and words are identified in the following table. Template sections begin with the `@` character, and template words begin with the `#` sign.

Template Section or Word	What It Does
<code>@ENDDEF</code>	Ends the subroutine definition
<code>@FXD1-@FXD7</code>	Specifies the tool and properties for fixed-drill cycles
<code>@FXDDEF</code>	Outputs a drill subroutine's values

continued

Template Section or Word	What It Does
@GOSUB	Codes a subroutine call
@SUBDEF	Starts the subroutine definition
#EBLK	Identifies the ending block number
#INCLUDE()	Includes an external file for subroutines. This creates a separate file for each subroutine prior to the coding of the main program. The name of each subroutine file is the name you assigned to the subroutine while creating the model. To add a <code>.sub</code> extension to these files, include the following evaluation sequence in the @GOSUB section: <pre style="margin-left: 40px;">#EVAL(#S0=#SNAME+ ".sub") #INCLUDE(#S0)</pre>
#REPEAT	Repeats the sub call #SREPT number of times
#ROT1	Identifies the rotation-angle start
#ROT2	Identifies the rotation-angle increment
#SBLK	Identifies the starting block number
#SBTYP	Identifies the subroutine type (0=none, 1=regular, 2=drill subs)
#SNAME	Identifies the subroutine's name
#SREPT	Identifies the number of times to repeat the subroutine
#XCTR	Identifies the X pattern handle position (always absolute)
#YCTR	Identifies the Y pattern handle position (always absolute)
#ZCTR	Identifies the Z pattern handle position (always absolute)

While creating a model, you can choose to define and use subroutines that include a variety of tool path elements. You can also define subroutines that relate specifically to drilling operations.

Regular Subroutines

To create code for regular subroutines, follow these steps:

1. Make sure `.smf` and `.tmp` settings are appropriate for your machine. These include settings for where the subroutines should be placed and the type of positioning mode.
2. Review the model, making sure the geometry is accurate.
3. If you plan to use absolute values for code output, the handle point for your subroutines must be at the world coordinate system's origin point, or the code generator should output a local coordinate system.
4. Select **Process—Code** and complete the coding process.

Drill Subroutines

SmartCAM provides a method for generating code for drill subroutines that initiate the drilling within the cycle's definition and for subroutines where the drilling starts after the cycle's definition (within the subroutine). There is also a way to address those machines that require the cycle definition and subroutine call to be contained in the same block. The following are some considerations for those options:

- Machines that Initiate Machining within the Cycle Definition

For these types of machines, the fixed-cycle template sections (@FXD1–@FXD7) contain the template variables that define the cycle along with the depth of the first hole. When SmartCAM accesses that section, the code for the drill's location has already been output. SmartCAM then outputs the code that performs the machining operation using the cycle's settings. The @GOSUB section codes the subroutine call, and the code for the remainder of the holes is output through the @FXDDEF section.

Include the first hole of the second pattern in the subroutine if you use the repeat option while defining the subroutine (see figure 3-2). This is necessary because the first hole of the first pattern is machined before the subroutine is called. Question 338 must be set to 0 to remove the redundant hole from the second subroutine call, causing only one hole to be drilled at that location.

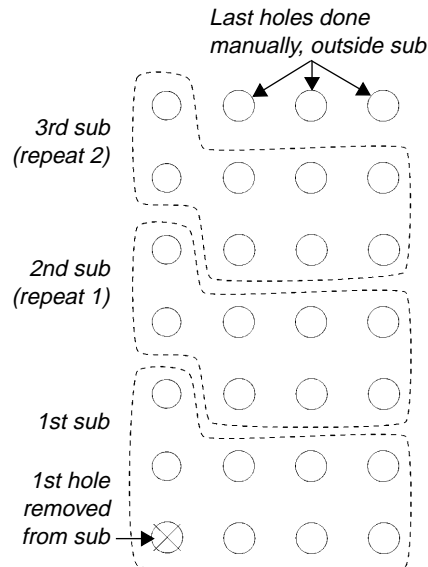


Figure 3-2
Remove the first hole for repeated drill subroutines.

- Machines that Initiate Machining after the Cycle Definition

For these machines, the fixed-cycle template section defines the cycle, but the machining code does not occur until `@GOSUB` calls the subroutine. When SmartCAM calls the subroutine, all the holes are coded using the `@FXDDEF` section. The repeat option is not available for this type of machine, and `.smf` question 338 must be set to `<1>` so that the first hole is left in place.

- Machines that Have the Cycle Definition and Sub Call in the Same Block

For these machines, include the appropriate template variables for your machine. These should include a cycle definition that calls the subroutine (`#SNAME`). The `@SUBDEF` section outputs the information relating to the position of the holes. You can also include a logic statement that tests the `#SBTYP` template word and either specifies the drill cycle and calls the drill subroutine (`#SBTYP` set to `<2>`) or outputs the normal drill-cycle statement. Place a similar logic statement in `@GOSUB` so that when `#SBTYP` is set to `<2>`, no sub call is made.

Code Filtering for Mesh Polylines

Code filtering can significantly reduce the amount of `G01` code output from mesh tool path polylines to the machine tool. To initiate code filtering, set `.smf` question 86 to `<1>` and identify the deviation tolerance with question 87. This establishes a tolerance bandwidth that controls when a point is output for a polyline.

Generating Code for Tool Changes and Tool Plane Changes

This section reviews how SmartCAM deals with tool and tool plane transitions during code generation. When SmartCAM encounters a tool change or a tool plane change during code generation, it accesses special template file sections. The order in which those sections are accessed (along with the `.smf` and `.tmp` settings) controls the format of the final code. Review this section to gain a better understanding of the process and the settings to make for your machine's code requirements.

- ☞ A model's work planes control the display of the elements on the screen. Tool planes control how SmartCAM displays the tool motion with Show Path and the code sent to the machine. Even though a work plane and a tool plane can share the same name and origin point, remember that the *tool* plane determines code output.

Tool plane changes and tool changes can occur independently or together. For example, a 4-axis machine using a 4-sided tombstone could have multiple parts on each of the four faces. The code needs to access each face and the individual parts each face contains (including related work coordinate systems). If a tool change occurs during a tool plane change (either on the face or for an index move), the code must provide the machine with the information to handle the transition properly. SmartCAM generates codes for these transitions based on the `.smf` and `.tmp` settings you use. Before generating code, make sure `.smf` question 303 reflects the way your machine expects to receive code.

Transition between Tool Planes and Tool Changes

A transition occurs in a CNC process model when the properties change from one element to the next. For instance, if a tool or tool plane assignment for element 6 differs from that for element 5, a transition must take place. SmartCAM uses the following order for processing transitions:

1. All motion for the element prior to the transition, including any clearance retractions (Z clear)
2. Any motion relating to the retraction options specified by `.smf` questions 69 or 289
3. The transition based on step 2 and the type of transition
4. All motion for the element after the transition, including any motion required to approach the start of the element

Questions 69 and 289 affect the retraction of a tool during Show Path and coding. Question 69 deals with retraction moves for tool changes, and question 289 deals with retraction during an indexing move. A setting of `<0>` requires you to add a point to the database for each retraction move. A setting of `<1>` or `<2>` generates an automatic retraction move.

The transition sections include `@TOOLCHG` (question 69), and `@TPINDX` (question 289). You use the same `#[XYZ]POS` template word for both transition sections. This enables you to make changes to question 69 or 289 without having to modify the template file.

☞ Use `#[XYZ]POS` instead of `#[XYZ]HOME` in the transition sections.

When a tool change and a tool plane change occur along with a rotary move, SmartCAM implements an order of precedence. First, SmartCAM outputs an automatic retraction point if either question 69 or question 289 specifies one. Second, a complete retract to home (setting `<1>`) takes precedence over a retract in Z (setting `<2>`).

There are various combinations of transitions and settings for the retraction questions. The following figures identify those cases along with the template sections SmartCAM calls. (The lines are not meant to imply rapid moves or Z-clear or Z-check positions.)

Tool Change or Tool Plane Change with Questions 69 and 289

Set to <0>

Show Path displays the transition, and Code processes it, in the following sequence (see figure 3-3):

1. The motion for the old element's tool or tool plane, including any retract to Z clear
2. The old tool's motion to the retract-to point (@RAP)
3. The tool change (@TOOLCHG) or tool plane change (@TPINDX)
4. The approach and motion for the new element's tool or tool plane

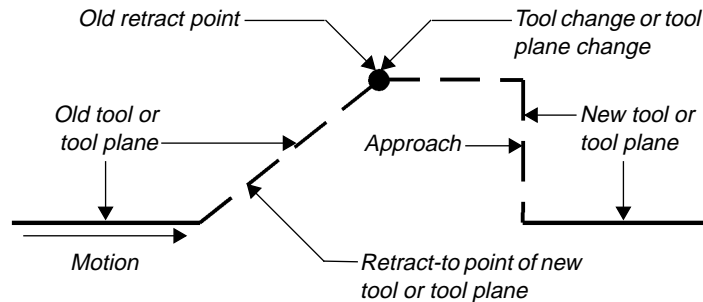


Figure 3-3
A tool change and a tool plane change can occur together.

Tool Change or Tool Plane Change with Questions 69 and 289 Set to <1> or <2>

No retraction point needs to be in the database, and retraction to home takes precedence over retraction to Z. Show Path displays the transition, and Code processes it, in the following sequence (see figure 3-4):

1. The motion for the old element's tool or tool plane, including any retraction to Z clear
2. The old tool's automatic retraction, based on question 69 or question 289 using priority process
3. The tool change @TOOLCHG), the tool plane change (@TPINDX), or both
4. The approach and motion for the new element's tool or tool plane

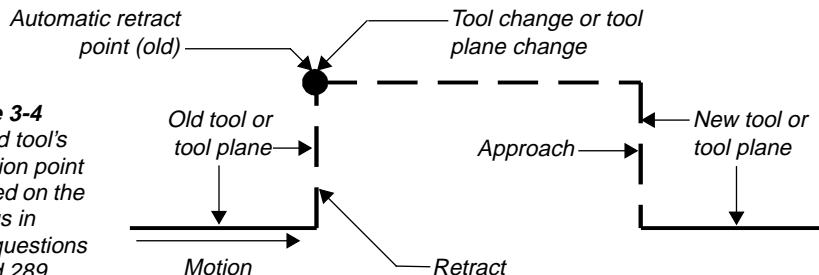
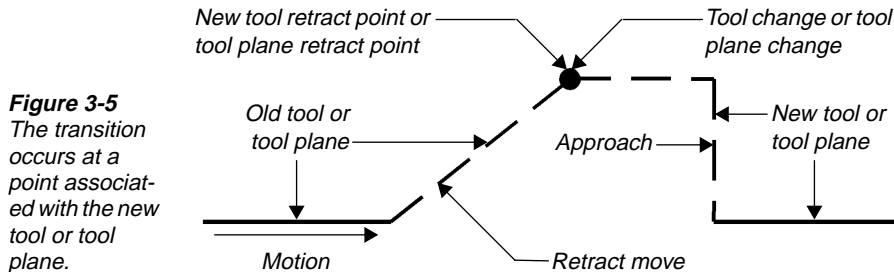


Figure 3-4
The old tool's retraction point is based on the settings in .smf questions 69 and 289.

Transition at a Point Associated with the New Tool or Tool Plane

Show Path displays the transition, and Code processes it, in the following sequence (see figure 3-5):

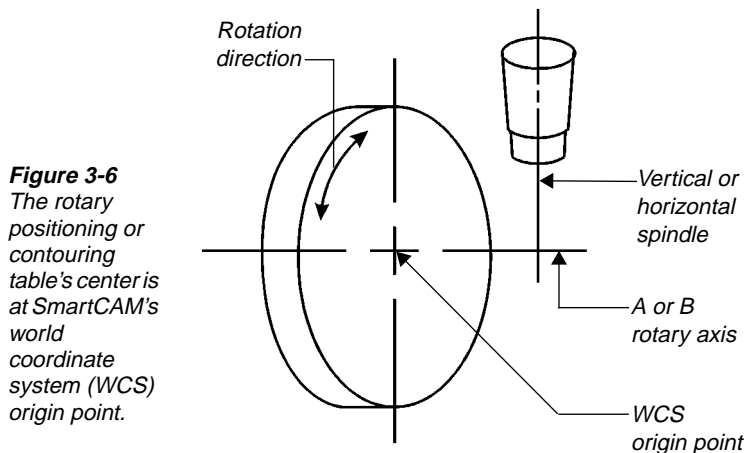
1. The motion for the old element's tool or tool plane, including any retraction to Z clear
2. The old tool retracting to the new tool point or tool plane point (@RAP)
3. The tool change (@TOOLCHG) or tool plane change (@TPINDEX)
4. The approach and motion for the new element's tool or plane



Generating Code for Rotary Positioning and Machining

With Advanced 3-D Machining, you can generate code for the following rotary table configurations:

- 4th-axis positioning for a table whose axis is centered on the X or Y axis of the machine (see figure 3-6). The center of the primary table's face typically matches the origin of SmartCAM's world coordinate system.



- Rotary contour machining where the rotary axis is centered on the X axis (A rotary axis) or Y axis (B rotary axis) (see figure 3-7). The right-hand rule determines positive rotation, with SmartCAM's world coordinate system being the origin of the rotational axis.
- 5th-axis dual rotary positioning where the table pivots for the 5th-dimensional move (see figure 3-7). SmartCAM's world coordinate system is commonly the center of the primary table. The offset distance from the pivot point to the center of the primary axis table is set with `.smf` questions 296 and 297.

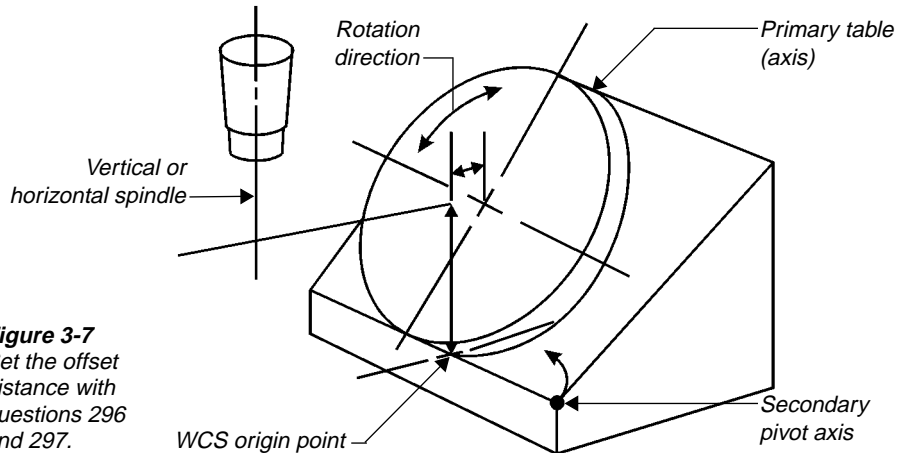
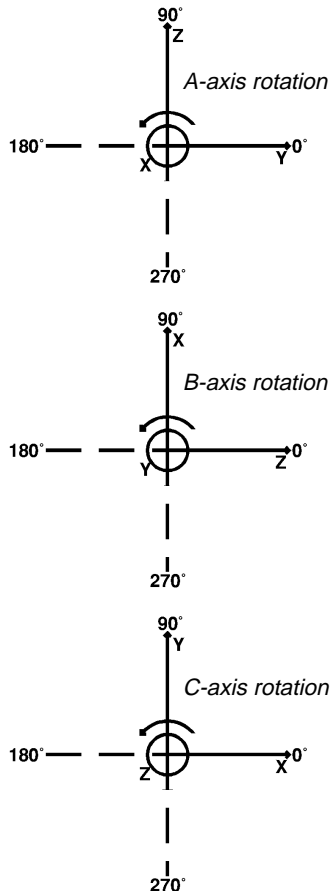


Figure 3-7
Set the offset distance with questions 296 and 297.

The orientation that Advanced 3-D Machining uses for each rotation axis determines the positive rotation angle and related code output. Figure 3-8 illustrates the orientation that Advanced 3-D Machining uses for the A, B, and C axes.



Axis	Direction of sight
A	Toward the negative X axis
B	Toward the negative Y axis
C	Toward the negative Z axis

Figure 3-8
The arrow around the center indicates the positive rotation direction.

The following sections examine the `.smf` and `.tmp` settings that relate to these different rotary machining orientations.

Machine File Questions for 4th- and 5th-Axis Positioning

To generate code correctly, you must position the CNC process model so that it aligns with the machine's rotational axes' reference origin. The machine's primary axis centerline and zero position along this centerline (generally the index table's top surface) must correspond to the world coordinate system's origin in the XY plane. The primary axis is usually the axis of a rotary table. For a 5-axis machine, the secondary axis is usually the axis that pivots the rotary table to different orientations about the spindle. The following machine define questions specify the code format for addressing the primary and secondary axes on the machine:

- 69
- 289–304

- 432–433
- 451–453

Review the various settings and the template file reference information to help you understand how to set up a code generator for your machine.

Machine File Questions for Rotary-Axis Machining

The CNC process model for rotary-axis machining should contain wrapped tool path. The center axis of the tool path must be aligned with the world coordinate system's X or Y axis. Before reviewing the other `.smf` questions that relate to rotary-axis machining, check the following questions to make sure you set them properly:

Question Recommended Setting

292	Select either setting <1> (X axis) or setting <2> (Y axis) to identify the primary rotary axis.
303	Select setting <1> for correct code output. All code values for wrapped tool path are based on the world coordinate system.

Other 4th- and 5th-axis-positioning questions that affect rotary-axis machining include the following:

- 291–292
- 294
- 299–302
- 304
- 432–433
- 451–453

Template Variables for Multiaxis and Rotary Machining

The following table identifies the template variables that affect how SmartCAM generates code for tool and tool plane transitions and rotary-axis machining. Review this list and the table that follows it to get a better understanding of the relationship of each variable to code generation.

<code>#[XYZ]ST</code>	The retract point in the old tool plane.
<code>#[XYZ]PSET</code>	The retract point in the new tool plane.
<code>#[XYZ]FO</code>	The tool plane's origin point based on the world coordinate system.

continued

#[XYZ]SET	The tool plane's origin point after any table rotation, based on the world coordinate system.
#[IJK]NO	The tool plane's normal (orientation) vector in its original (unrotated) position.
#INDX[ABC]	The angles necessary to rotate the tool plane into machining position.
#ST[ABC]	The start angle.
#[ABC]DIR	The direction of rotation.
#WKPLN	The name of the new tool plane.
#WKSCHG	<p>A flag indicating a tool change and tool plane change combined.</p> <p><0>=Tool change or tool plane change</p> <p><1>=Tool change and tool plane change</p> <p>This can be used by template file logic in the @TOOLCHG or @START section to sense when a tool plane change is combined with a tool change. Typical use is to exit template sections early or to call another template section.</p>
#TINDX	<p>A flag distinguishing tool plane transition types.</p> <p><0>=No tool plane change</p> <p><1>=Transition between parallel tool planes with the same origin</p> <p><2>=Transition between parallel tool planes with different origins (3-D offset)</p> <p><3>=Transition between nonparallel planes with the same origin (index move)</p> <p><4>=Transition between nonparallel tool planes with different origins (index move and 3-D offset)</p> <p>Use this variable with template file logic to determine when it is appropriate to exit template sections early or to call another template section. For example, if you wanted to use the @WKSYS section when a tool plane change took place without an indexing move, you could add the following test to the @TPINDX section:</p> <pre> IF <#TINDX=2> CALL @WKSYS IF <#TINDX=2> #EXIT </pre>

The following table represents template variable assignments and template processing for the different transitions, including a special transition that starts on a non-XY plane. Small letters in the first column indicate footnotes.

	Tool Change	Plane Change	Tool/Plane Change	Start in Non-XY Plane
#[XYZ]ST abcde	Retract/last pos	Retract/last pos <i>old plane</i>	Retract/last pos <i>old plane</i>	Start pos <i>new plane</i> ^f
#[XYZ]PSET ^{adeg}	Retract/last pos	Retract/last pos <i>new plane</i>	Retract/last pos <i>new plane</i>	Start pos <i>new plane</i>
#[XYZ]FO ^g	N/C	Plane loc world <i>unrotated</i>	Plane loc world <i>unrotated</i>	Plane loc world <i>unrotated</i>
#[XYZ]SET ^{gh}	N/C	Plane loc world <i>rotated</i>	Plane loc world <i>rotated</i>	Plane loc world <i>rotated</i>
#[XYZ]HOME acdei	Home pos	Home pos <i>old plane</i>	Home pos <i>old plane</i>	Home pos <i>new plane</i>
#[LJK]NO ^h	N/C	Plane normal vector	Plane normal vector	Plane normal vector
#INDX[ABC] ^{jh}	N/C	Table angles <i>new plane</i>	Table angles <i>new plane</i>	Table angles <i>new plane</i>
#WKPLN	N/C	New plane name	New plane name	New plane name
#WKSCHG	0	0	1	1
#TINDX ^h	0	1-4	1-4	1-4
.tmp processing	@TOOLCHG	@TPINDX	@TOOLCHG @TPINDX	@START

^a Output is affected by question 303 (local or world input).

^b Output is affected by questions 69 and 289.

^c Output is absolute or incremental based on #ABSI.

^d Output adds #[XYZ]POFF per question 34 when in absolute mode (#ABSI=0).

^e Updates [XYZ]POS after TPINDX closes.

^f Measures inverse of [XYZ]FO.

^g Output is always absolute (ignores #ABSI).

^h Output is 0 in SmartCAM milling.

ⁱ Version 6.0 had a problem and processed this as new.

^j See 4th- and 5th-axis template section for these and related words #[ABC]dir and #ST[ABC].

Linear Equivalent Feed Rate Conversion

The following is a description of the linear equivalent feed rate conversion used when .smf question 435 is set to <0> or <3> :

$$\mathbf{\#FEED} = F_{dpm} \frac{\sqrt{X^2 + \left(A \frac{\#436}{360}\right)^2}}{\sqrt{X^2 + \left(A \frac{\pi D}{360}\right)^2}}$$

where:

#436 = setting for .smf question 436 (rotary axis linear equivalent distance)

X = linear component of the move

A = angular component of the move

D = diameter of the tool (specified in the job operations file)

F_{dpm} = DPM feed rate (set in the job operations file)

The lower term is the actual length of the move:

$$L_{act} = \sqrt{X^2 + \left(A \frac{\pi D}{360}\right)^2}$$

The top term is the linear equivalent length of the move:

$$L_{eq} = \sqrt{X^2 + \left(A \frac{\#436}{360}\right)^2}$$

So, the equation can be written as:

$$\mathbf{\#FEED} = F_{dpm} \frac{L_{eq}}{L_{act}}$$

And since length/feed is the time for the move (T), the equation can also be written as:

$$\mathbf{\#FEED} = \frac{L_{eq}}{T}$$

since

$$T = \frac{L_{act}}{F_{dpm}}$$

Notice also that if question 436 is set to 360, the equation becomes:

$$\mathbf{\#FEED} = F_{dpm} \sqrt{\frac{X^2 + A^2}{X^2 + \left(A \frac{\pi d}{360}\right)^2}}$$

This is a common form used by many controllers.

Degrees-Per-Minute Conversion

The following is a description of the degrees-per-minute conversion when .smf question 435 is set to <2> or <3>:

$$\mathbf{\#FEED} = F_{dpm} \frac{A}{\sqrt{X^2 + \left(A \frac{\pi D}{360}\right)^2}}$$

where:

X = linear component of the move

A = angular component of the move

D = diameter of the tool (specified in the job operations file)

F_{dpm} = DPM feed rate (set in the job operations file)

The lower term is the actual length of the move:

$$L_{\text{act}} = \sqrt{X^2 + \left(A \frac{\pi D}{360}\right)^2}$$

So, the equation can be written:

$$\mathbf{\#FEED} = F_{\text{dpm}} \frac{A}{L_{\text{act}}}$$

Since the time for the move (T) is:

$$T = \frac{L_{\text{act}}}{F_{\text{dpm}}}$$

The equation can also be written:

$$\mathbf{\#FEED} = \frac{A}{T}$$

Machine File Reference

This chapter contains a complete list of `.smf` questions, explanations, and recommended settings. Use this chapter as a reference when using Machine Define to change machine file settings. (For a complete explanation of how to use the Machine Define utility, see chapter 2.)

Refer to the following list of machine file question categories when you want to look at or change questions covering a specific topic. For example, if your code generator is outputting correct code for everything except axis orientation, you need to check questions 20 through 39.

☞ When using Machine Define, you can list questions by category using the Select Categories option. However, those categories are not the same as the ones listed here.

Category	Question Numbers
General	1–9
Block number control	10–19
Units and axis orientation	20–39
Feeds, speeds, and dwell	40–59
Tool change	60–79
Rapid positioning	80–84
Linear profile moves	85–89
Circular arc profile moves	90–109
Profile contouring	110–119
Cutter compensation	120–136
Cycle time	138–149
Mill and lathe peck drill cycle parameters	150–159

Category	Question Numbers
Mill cycles	160–179
Lathe cycles	180–184
Lathe threading	185–195
Lathe groove cycle	200–204
Lathe 4-axis	210–219
Punch cycles	68, 220–249
4-axis wire EDM	259–266
3-D arcs and helixes	270–286
4- and 5-axis positioning (advanced products)	289–304
Subprogramming (advanced products)	330–338
User commands	360–371
Tape to Shape	390–412
Rotary contouring (advanced products)	430–453

☞ If you are just beginning to work with SmartCAM code generators, you should change only one .smf question at a time. Then run a test file and see how that setting changes your NC code. This technique enables you to learn more quickly how various settings affect code generation.

General

1. Template filename to use

Explanation: This question sets the default template file name to be used with this .smf file. You should set up a different template file for each machine and special setup that you use. Be sure to follow the naming convention for your operating system:

filename.tmp

Use the name that matches the template file you plan to use. You must include the .tmp extension.

Related Settings: The default .smf filename that displays when generating code is set by the machine field in the job operations file for the model.

3. Tape-to-Shape .tts filename to use

Explanation: This item specifies the name of the Tape-to-Shape .tts file SmartCAM uses with the .smf file. The name must be a valid filename with a .tts extension. Follow the naming convention for your operating system.

5. Update template words after each block

Choices:

- <0> No
- <1> Update command words (G & M words)
- <2> Update command and numeric words
- <3> Update last position only

Explanation: Template words are normally updated when they are output. Using a setting other than 0 can cause confusing results. For example, a template word might not be conditionally output because SmartCAM updated it.

Recommended Setting: <0>

6. Replacement filler string for unchanged conditions

Explanation: Indicate the filler string you want to use when unchanged conditions eliminate requirements for a code word. Controllers that require columned or tab-sequential output use this type of filler output. (For example, use a tab as a space filler when the code word is not output).

Recommended Setting: This item should be empty for most modern machines.

7. Fixed file length total number of lines

Explanation: Specify the fixed length for your NC code files. For example, all NC code files for this machine will have exactly 200 lines of code if you enter a value of 200.

Recommended Setting: Set to 0 for most machines, indicating that the files can be of any length.

8. Decimal point character to be used for decimal format

Choices:

- <0>. (decimal point)
- <1>, (comma)

Explanation: Specify the character that SmartCAM outputs as a decimal when using a decimal format. For example, specifying <1> causes SmartCAM to place a comma between integer and decimal values.

Recommended Setting: This is usually only a concern for European machines. It should be set to <0> for most applications.

9. Integer numeric format

Explanation: Specify the numeric format SmartCAM should use when outputting integer words. (See page 3-2 for information about numeric formats.) Use with template words such as #U0–#U9, #NHOL, and #NHOL2.

Block Number Control

10. Block or line number format

Explanation: Specify the numeric format for block numbers.

11. Block number code letter

Explanation: Designate the letter (usually N) to precede each block of code.

Related Settings: The character entered is output only when #ONBLK is active. The character is turned off (not output) by #OFFBLK.

12. Safe start block number code letter

Explanation: Specify a letter you want to precede every safe start block. A safe start block is a location in code where it would be safe to restart the program after it is interrupted. This is typically at a tool-change location.

Related Settings: The output of this character is dependent on the use of #SAFBLK in the template file @START or @TOOLCHG sections.

13. Start block numbers with

Explanation: Specify the starting integer number of your block numbers. For example, if you use 100, SmartCAM numbers the first block of the program as 100.

Related Settings: Question 10 sets the block number format. Question 11 sets the character that displays before the block number (for example, N). The template word #ONBLK must occur in the template file on the line that you want block numbers to start (most typically in @START). #OFFBLK will stop output.

14. Increment between block numbers

Explanation: Indicate the increment SmartCAM should use between block numbers. For example, when you enter 5, each new block is numbered 5 higher than the last.

Related Settings: #ONBLK must be used in the template file.

15. Start safe start sections with an even increment of (#SAFBLK)

Explanation: Specifies the even increment number the safe start block code will start at when using them. For example, if you use 100, each safe start section will begin with an even hundred.

16. End of block character

Explanation: Specify any special character or string of characters you want at the end of every block of code.

17. Default block delete string

Explanation: Specify a / (slash) if you plan to use the block-delete option.

Related Settings: See template words #BDSTR and #BLKDEL on page 6-3.

Units and Axis Orientation

20. Code inch or metric units

Choices:

<0> No unit conversion

<1> Inches

<2> Millimeters, metric units

Explanation: Specify the type of units you want SmartCAM to output code in. If the job operations file specifies inches and you set this question to <2>, SmartCAM converts the model geometry into metric code.

Recommended Setting: Typically <0>; the job operations file is set to the correct code units.

21. Smallest machine position increment

Explanation: Specify the smallest increment that the machine can move. The degree of accuracy in the NC code is based on your response to this question. For example, if you specify 0.0001, SmartCAM outputs code to the nearest ten-thousandth.

22. Axis position format

Explanation: Specify the numeric format SmartCAM should use for axis position values. (See page 3-2 for information about numeric formats.)

23. Angular position format

Explanation: Specify the numeric format SmartCAM should use for all angular position values. (See page 3-2 for information about numeric formats.)

24. Specify angles in range*Choices:*

- <0> Angles are always output and always take on values between 0 and +360 deg
- <1> Angles are always output and never greater than 180 or less than -180
- <2> -360 to +360 deg, start angle is always before end angle
- <3> 0 to -360 for CW arcs, 0 to +360 for CCW arcs

Explanation: For setting <2>, counterclockwise arcs will have a positive total angle and an end angle greater than the start angle. Clockwise arcs will have a negative total arc angle and an end angle numerically smaller than the start angle. For example, an arc from 45 to -45 would be expressed as 45 to 315 for counterclockwise and 45 to -45 for clockwise.

For setting <3>, start or end angle values are always less than 360 or greater than -360. An angle value of 0 is output instead of 360 or -360.

25. Default positioning mode*Choices:*

- <0> Absolute
- <1> Incremental

Explanation: Specify if the default positioning mode should be the absolute distance relative to an origin or an incremental distance relative to the last position. If you are going to use SmartCAM to generate subprograms, you may want your code to be generated as incremental. This question only affects the main program output.

Related Settings: See question 332 for subcode default positioning models.

26. Absolute positioning mode

Explanation: Enter the command code your machine uses to designate an absolute positioning mode. SmartCAM outputs the code (typically G90) when changing from incremental to absolute mode.

27. Incremental positioning mode

Explanation: Enter the code your machine uses to designate an incremental positioning mode. SmartCAM outputs the code (typically G91) when changing from absolute to incremental mode.

30. Horizontal SmartCAM system axis*Choices:*

- <0> X machine axis
- <1> Y machine axis
- <2> Z machine axis

Explanation: Designate which axis in your model file is the horizontal axis. For milling and punching it is typically the X axis; for lathe, the Z axis.

Related Settings: See question 32.

31. Sign of movement to right on horizontal axis

Choices:

<+1> Positive

<-1> Negative

Explanation: Specify whether positive or negative values are on the right of the horizontal axis.

Recommended Setting: <+1> Right side of horizontal axis is positive.

32. Vertical SmartCAM system axis

Choices:

<0> X machine axis

<1> Y machine axis

<2> Z machine axis

Explanation: Specify which axis of your model file is the vertical axis. For milling and punching it is typically the Y axis; for lathe, the X axis.

33. Sign of movement up on vertical axis

Choices:

<+1> Positive

<-1> Negative

Explanation: Specify if positive or negative values are on the upper side of the vertical axis.

Recommended Setting: <+1> Use positive values for the upper side of the vertical axis.

34. Sign to use with position offset (#XPOFF, #YPOFF, #ZPOFF)

Choices:

<-1> Subtract the offset from absolute axis position

<0> Ignore position offsets

<1> Add the offset

Machine Types: Punch, lathe, mill

Explanation: Specify the sign SmartCAM should attach to an open value. Use for punch repositioning or fixture offsets on certain machines. For mill, #ZPOFF usually holds the TLENGTH from the job operations file.

Related Settings: For lathe, set question 66 to <4>.

35. Mill machine type use 'Z' axis*Choices:*

- <0> Use 'Z' axis positioning commands
- <1> No 'Z' axis positioning

Machine Types: Mill

Explanation: Specify if you want to use Z axis positioning moves. If the Z axis is nonprogrammable, as in some drilling machines that use stops, set this item to <1>.

36. Sign of movement away from part on depth axis*Choices:*

- <+1> Positive
- <-1> Negative

Machine Types: Mill

Explanation: Specify if movement away from the part on the depth axis is positive or negative.

37. Use lathe 'X' position code as*Choices:*

- <0> Radius
- <1> Diameter
- <2> Diameter if absolute positioning mode

Machine Types: Lathe

Explanation: Specify whether the X dimension represents diameter value or radius value. If you use setting <1>, SmartCAM multiplies all the database X values by 2.

Related Settings: See question 98 for a related item about arc center values.

38. Make 'X' position values even*Choices:*

- <0> No
- <1> Yes, use an even multiple of 2 times the position unit for all X values

Machine Types: Lathe

Explanation: Set this question to <1> if your machine requires diameter values that are even multiples of the smallest machine increment.

Feeds, Speeds, and Dwell

40. Output feed rate with the first move of a new profile

Choices:

<0> No

<1> Yes

Explanation: Set to <1> if you want SmartCAM to output a feed rate code after a fast feed rate from rapid.

Related Settings: See question 81. For punches with torches or attachments, see question 233.

41. Adjust feed rate for Z depth moves by

Machine Types: Mill, Router

Explanation: Specify the feed rate for plunge cuts occurring from the ZCHK level to the first ZDEPTH. The value you enter is a percentage of the feed rate you set in the job operations file for profiling. For example, if you enter 0.5, SmartCAM outputs a feed rate that is 50% of the profiling feed rate for the first plunge move.

Related Settings: Note that you can update this factor during the creation of a model by assigning a new value to the #FDMULT template word.

42. Feed rate format

Explanation: This is the numeric format for feed rate decimal, filled, or trailing and the number of digits assumed to the left and the right of the decimal. For example, if you enter D3.1 the numeric format is decimal with 3 digits left and 1 digit right of the decimal point. (See page 3-2 for information about numeric formats.)

43. Secondary feed rate format

Explanation: Enter the numeric format for a secondary feed mode, such as the increment per hit (IPH) for punch nibbling, or the inches per revolution (IPR) feed rate for a lathe. (See page 3-2 for information about numeric formats.)

44. Feed mode 1 IPM (inches or millimeters per minute)

Explanation: Enter the command code that activates the primary feed mode for your machine. The feed mode is typically inches or millimeters per minute. G94 is a common code that machines use to activate the primary feed mode.

45. Feed mode 2 IPR (Inches or Millimeters per revolution)

Machine Types: Mill, lathe

Explanation: Enter the command code that activates the secondary feed mode for your machine. G95 is common for activating an IPR feed rate for a lathe.

47. Adjust feed rate for periphery of arcs

Choices:

<0> No

<1> Yes

<2> Use machine commands

Explanation: Use setting <1> if you want SmartCAM to adjust the feed rate for arcs. SmartCAM will automatically adjust the feed rates on arcs to maintain the same material removal rate as for linear elements.

Use setting <2> if your controller needs to receive a command for making this adjustment.

Related Settings: See questions 48 and 49.

48. Adjust feed for periphery of arcs off (#FDADJ)

Explanation: Specify the command code your machine uses to turn off adjustments for arc feed rates. For example, G72 returns some machines to constant feed for tool center.


Related Settings: The status of questions 48 and 49 can be set with a user command. Assign #FDADJ=0 for off or 1 for on. Be sure to set question 47 to <2> if you plan to use the command.

49. Adjust feed for periphery of arcs on

Explanation: Specify the command code your machine uses to turn on the feed rate adjustment for arcs. For example, G73 causes some machines to maintain constant feed for part profiles, adjusting the feed rate for the centerline of the tool.

50. Initial default dwell value (#DWELL)

Explanation: Enter the initial value for the #DWELL template word. You can change this setting later with a user command. The value you enter is the number of seconds for the dwell. For example, entering 0.2 results in a default dwell of 0.2 seconds.

 This value comes from the job operation pages when it is available.

51. Dwell time format

Explanation: Specify the format for maximum dwell time in seconds. (See page 3-2 for information about numeric formats.) For example, entering D3.1 sets the time in decimal format up to 999.9 seconds.

52. Use speeds codes from @SPEEDS table

Choices:

<0> No

<1> Yes

Machine Types: Mill, router, two-axis WEDM, lathe

Explanation: Use setting <1> for machines that require a speed code number instead of an RPM value. If you activate this setting, SmartCAM checks the job operations file for the desired RPM and then looks up the corresponding speed code from the @SPEEDS table in the template file.

53. Spindle speed format

Machine Types: Mill, router, lathe

Explanation: Enter the numeric format for spindle speed. (See page 3-2 for information about numeric formats.) For example, entering T4.0 results in a trailing format of one to four digits.

Related Settings: If your controller uses a specific code number to represent a speed, see question 52.

54. Spindle off code

Machine Types: Mill, router, lathe

Explanation: Enter the command code your machine uses to turn M05 is a standard code some machines use for the function.

55. Spindle on forward code

Machine Types: Mill, router, lathe

Explanation: Enter the command code your machine uses to turn M03 is a standard code some machines use for this function.

56. Spindle on reverse code

Machine Types: Mill, router, lathe

Explanation: Enter the command code that your machine uses to turn the spindle on in the reverse direction. SmartCAM outputs the spindle reverse command when it encounters a negative value for M04 is a standard code some machines use for this function.

57. RPM speed mode code (Cancel CSS)

Machine Types: Mill, router, lathe

Explanation: Enter the command code your machine uses to cancel constant surface speed (CSS) and return to revolutions per minute (RPM). For example, some machines use G97 to cancel CSS and return to RPM.

58. CSS speed mode code

Machine Types: Mill, router, lathe

Explanation: Enter the command code your machine uses to turn on constant surface speed (CSS) mode. For example, some machines use G96 to turn on CSS.

Tool Change

60. Reset all numeric template words at each tool change

Choices:

<0> No

<1> Yes

Explanation: Specify whether you want numeric template words output at a tool change only if their current settings have changed.

Recommended Setting: <1> This ensures that all the code includes necessary information for every tool change.

61. Default positioning command to be reset at tool change

Choices:

<0> Reset to rapid move command (G00)

<1> To force out active command with first move following change

Explanation: Select what you want the machine to do at a tool change—reset to the rapid-move command or replace the active command with the first move command after a tool change (rapid, feed, etc.).

Recommended Setting: <1>

62. Use tool change template word for switch to new tool

Choices:

<0> No

<1> Yes

Explanation: If you use setting <1>, SmartCAM updates any new tool information in the tool change routine after the control word #TLCHG is found in the template.

Recommended Setting: <0>

63. Specify next tool (#NTOOL) after the last tool used

Choices:

<0> No

<1> Yes, use the first tool

Explanation: Indicate if you want #NTOOL in the template file to output the tool number the machine uses after the next tool change. This is most often used with machines that have an automatic tool change.

Related Settings: This item is used with the #NTOOL word in the template file.

64. Tool number format

Explanation: Specify the numeric format for a tool number. (See page 3-2 for information about numeric formats.) Common settings include trailing (one or two digits allowed) or filled (the left two digits for the tool number and the right two digits for the offset number). The following are examples of possible settings:

T2.0 Trailing format, one or two digits allowed.

F2.2 Filled format will use left two digits for tool number and right two digits for offset number.

Related Settings: The filled format is typically used in turning applications.

65. Offset format

Explanation: Specify the numeric format SmartCAM should use when outputting both length and diameter offset addresses. This is the address used by the controller, not the offset value itself. (See page 3-2 for information about numeric formats.) The following are examples of possible settings:

T2.0 Trailing format, one or two digits allowed.

F2.0 Filled format will always use two digits.

66. Use of #XSET and #ZSET at tool change

Choices:

- <0> Assign #XSET & #ZSET directly from the job operations file
- <1> Previous tool's position includes #XSET & #ZSET difference
- <2> New tool's position includes #XSET & #ZSET difference
- <3> Both tool positions include #XSET & #ZSET
- <4> Add as an offset for every move

Machine Types: Lathe

Explanation: Indicate how SmartCAM should arrive at #XSET and #ZSET values. Answer <0> indicates the values for #XSET and #ZSET are those found in the job operations file.

Recommended Setting:

Use <0> if you reference tools to a known 0 coordinate on the machine tool. The #XSET and #ZSET values are output directly and used to preset the tools' position using a G92 or G50 command. The tools will start from and return to the same coordinates. During Show Path, the tools should appear at and return to the tool-change coordinates modified by the #XSET and #ZSET values.

Use <3> to use the machine's Preset capability to specify the coordinates for the new tool when it indexes at tool change. Tools are *not* referenced to a machine 0 point. During Show Path, the previous tool's and the new tool's coordinates are based on the tool-change point modified by the #XSET and #ZSET values for each tool.

Related Settings: Setting <4> works only if question 34 is not set to <0>.

67. Sign to use with XSET and ZSET values*Choices:*

<+1> Positive

<-1> Negative

Machine Types: Lathe

Explanation: Indicate which direction XSET and ZSET values identify in the job operations file. A positive setting indicates that a tool closer to the chuck (longer) has a negative ZSET value in the job operations file.

68. Use a punch off point for tool change with torch*Choices:*

<0> No

<1> Yes

Machine Types: Punch, contour machine

Explanation: For combination punch machines with torches or other attachments, this item enables a switch to the different tool without a tool change occurring in the turret. However, a setting of <1> allows you to force a tool change by using a point element.

69. Tool retraction for tool change specified*Choices:*

<0> By placing a rapid point at the desired retract position.

<1> As the Home Point defined by the first tool position in the file.

<2> Retract to Z Home at current X,Y position.

Machine Types: Mill

Explanation: Specify which of the following you want to happen:

<0> The user must insert a point prior to tool change.

<1> SmartCAM inserts a point at the same location as the position of the first tool in the database.

<2> SmartCAM outputs only a Z axis move at tool change.

The code output that results from the settings is as follows:

<0> Outputs and updates ZPOS, XPOS, and YPOS.

<1> Outputs but does not update XHOME, YHOME, and ZHOME.

<2> Outputs but does not update ZHOME.

70. TTS, tool change type*Choices:*

<0> Change tool automatically with tool number code word
 <1> Wait for a separate tool change command to activate tool number
 <2> No tool number specified. Use tool change command to index through the job operations file

Explanation: Specify what format your machine uses when initiating a tool change. Some machines automatically change tools when receiving a new tool number. Others wait for a separate tool change command, enabling the tool number to be given earlier. This is often done with machines that have automatic tool changes, because it permits a new tool to be in place when a change command is given.

71. TTS, tool code word

Explanation: Enter the code that your machine uses when outputting a new tool number. For example, enter a T if a tool number is preceded by a T.

72. TTS, tool change command

Explanation: Specify the tool change command your machine uses to indicate that the previously selected tool number is now the active tool.

Related Settings: Use M06 if question 70 is set to <1>.

Rapid Positioning

80. Rapid position move code

Explanation: Enter the command code your machine uses to indicate a move at its maximum positioning speed. For example, some machines use G00 code for rapid movement.

Related Settings: See question 81.

81. Rapid feed string

Explanation: If your machine uses a special code for indicating rapid moves, specify that code for this question. Special command codes can call for a rapid move or identify a rapid move with a special feed rate. Some machines use the same code for all linear moves, so this setting distinguishes between regular and rapid moves. Examples of this special code include:

- R Special code to indicate a rapid move
- F0 Special feed rate to indicate a rapid move

82. Show path dogleg rapid moves

Choices:

<0> No

<1> Yes, for display of rapid moves

Explanation: Identify the way you want Show Path to display rapid moves so that you match the machine's motion and enable reliable checking for collision avoidance.

83. Tool check distance above profile top surface

Machine Types: Mill, router, laser, 2-axis wire EDM

Explanation: Enter the default distance your machine stops a rapid move above the Z level of a hole to be drilled or above the profile top when cutting a profile. (This is the #CHKD word the template file uses.) For example, 0.05 indicates a stop rapid at 0.05 above the part surface.

Related Settings: See questions 117 and 173.

Linear Profile Moves

85. Linear cutting move code

Explanation: Enter the command code your machine uses for a linear cutting move made at a specified feed rate. For example, most machines use G01 for linear cutting modes.

☞ Questions 86 and 87 pertain to SmartCAM advanced products only.

86. Output G01 using polyline coordinates defined

Choices:

<0> Yes

<1> No, interpolate G01 moves with a maximum deviation value

Explanation: Code filtering can significantly reduce the amount of G01 code output from surface tool path polylines to the machine tool. To initiate code filtering, set .smf question 86 to <1> and identify the deviation tolerance with question 87. This establishes a tolerance bandwidth that controls when a point is output for a polyline.

87. Maximum G01 deviation value

Explanation: Enter the tolerance band size for the code-filtering function.

Circular Arc Profile Moves

90. Use arc command

Choices:

<0> No, interpolate arcs using line commands

<1> Yes

<3> Use arc for EMCO F1 Mill with target PE in #XPASS, #YPASS

Explanation: Indicate whether you want to output arc commands or interpolate arcs using line commands. If you choose <1> here, your code consists of only points and lines. SmartCAM codes any arcs in your model as a series of short line segments.

Related Settings: Maximum deviation is specified by question 91. If question 90 is set to <0>, SmartCAM generates linear moves or individual hits for punch. If it is set to <1>, SmartCAM uses the machine's arc command.

91. Maximum deviation from true arc

Explanation: When interpolating an arc with line commands, enter the maximum allowable amount of deviation the line segments can be from the actual arc. The smaller the increment, the longer the program and the smoother the arc.

Related Settings: This applies if question 90 is set to <0> or if question 99 applies.

92. Clockwise circular cutting code

Explanation: Enter the command code your machine uses for a clockwise arc cutting move with a specified feed rate. For example, some machines use G02 to identify a clockwise arc cutting move.

93. Counterclockwise circular cutting code

Explanation: Enter the command code your machine uses for a counterclockwise arc cutting move with a specified speed rate. For example, enter G03 if your machine uses that code for this function.

94. Arc Direction

Choices:

<1> Standard

<-1> Reversed

Explanation: If arcs are being cut in the reverse direction of how they appear in the model, use setting <-1>. This may occur when your machine views arc direction differently from the way SmartCAM views it.

95. Arc quadrants allowed*Choices:*

- <0> Single quadrants only
- <1> Multiple quadrants
- <2> 180 deg max total angle for any arc

Explanation: Specify the format your machine uses for arc coordinates. Some machines output coordinates for only a single quadrant in each program block. It then takes four program blocks to complete a full circle. Other machines use multiple quadrants, and a full circle can be programmed in one block of code.

Related Settings: See question 282.

96. Arc center location mode*Choices:*

- <0> Always incremental distance to center
- <1> Absolute location when in absolute positioning mode (G90)
- <2> Always absolute location regardless of positioning mode

Explanation: Specify the format your machine uses for arc center coordinates. Most machines specify an arc center using the incremental distance from one arc's start point to the center, even when positioning moves are specified in absolute mode. Others require the absolute position of the arc center when using absolute positioning mode.

97. Arc center defined as*Choices:*

- <0> Distance from start point to center
- <1> Distance from center to start
- <2> Unsigned distance, start to center

Explanation: Indicate the format your machine uses for defining the arc's center. Some machines specify the distance from the start to the center; some use the distance from the center to the start. These formats result in opposite signs for the center location. Other machines specify an unsigned distance from start to center.

98. Specify 'X' axis arc center location code as*Choices:*

- <0> Radial dimension
- <1> Diameter
- <2> Diameter if absolute positioning mode

Machine Types: Lathe

Explanation: Indicate the format your machine uses for the X axis arc center coordinates. You can specify arc centers as radius or diameter.

Related Settings: See question 37.

99. Maximum programmable arc radius

Explanation: Enter the maximum size of arc radius your machine accepts. A setting of 0 indicates that any arc-radius size is permissible. A setting of 50.0 indicates that all arc radii less than 50.0 use the arc command. Any larger radii are interpolated into line segments.

Related Settings: See question 91.

100. When radius used with arc cmd, indicate which center

Choices:

- <0> No special indication
- <1> Use negative radius to indicate arcs >180 deg
- <2> Use special command to indicate arcs >180 deg
- <3> Indicate arc center as left or right (as Hamil control)

Explanation: If your machine uses a radius in arc commands, indicate the format designating the center.

101. Less than 180 deg arc center indication code

Explanation: If your machine requires a command code designation to specify an arc less than 180 degrees, enter that code here. For example, some machines use C1 for this designation.

Related Settings: If your response to question 100 is <2>, questions 101 and 102 refer to an angle. If your response to question 100 is <3>, this question answers whether the center is left or right of a line drawn from start to end of the arc.

102. Greater than 180 deg arc center indication code

Explanation: See question 101. Some machines use C2 for this designation.

103. Minimum total arc angle

Explanation: Enter the minimum angle your machine allows for arcs. SmartCAM interpolates any arc with a total angle less than the value you enter into line segments instead of an arc. For example, if you enter 1.0, SmartCAM converts any arc with an angular span less than one degree into line segments.

104. Fix arc angle position

Choices:

- <0> No
- <1> Yes

Explanation: If your machine specifies arcs using start angle and end or total angle without a center specification, indicate <1> if you want SmartCAM to add a line segment at the start or end of the arc. This line segment makes up for any errors due to the rounding of the angle specification.

Related Settings: See question 103.

105. Arc angle fix tolerance

Explanation: If you are using a fix arc angle position (question 104 set to <1>), enter the tolerance that determines whether a line segment should be added. For example, 0.1 indicates that SmartCAM should add a line segment when the start or end of the arc falls short of the total angle by more than 0.1 degree.

Related Settings: See question 104.

106. Maximum deviation from true curve

Explanation: Enter the maximum amount that the arc segments for spline and ellipse curve elements can deviate from the actual path the mathematical formula generates for the curve.

Profile Contouring

110. Contour Off code

Machine Types: All

Explanation: Enter the command code your machine uses to end each contour. For example, some machines use M18 to turn off nibble on punches.

Related Settings: See template word #CUTC on page 6-5.

111. Contour On code

Machine Types: All

Explanation: Enter the command code your machine uses to start each contour. For example, some lathes and burning machines use M17 for this function.

112. Secondary Contour Off code

Machine Types: Punch, contour machine

Explanation: If your machine has two contouring tools (for example, a burner and a spot punch), enter the secondary contour-off command code your machine uses (for example, M32).

Related Settings: See template word #CUTC on page 6-5.

113. Secondary Contour On code

Machine Types: Punch, contour machine

Explanation: If your machine has two contouring tools (for example, a burner and a spot punch), enter the secondary contour-on code your machine uses (for example, M34).

Related Settings: See template word #CUTC on page 6-5.

116. Contour cutting cmd used w/profile end move (#CUTC)*Choices:*

<0> Off

<1> On

Machine Types: All

Explanation: This item should normally be set to <1>. Set it to <0> only when you would want to include the contour-off command in the same block as the final contour-cutting move of a profile.

117. Use Z position depth move with start profile*Choices:*

<0> No

<1> Yes

Machine Types: Mill, router

Explanation: Indicate where your machine looks for Z positioning depth moves. For machines that require Z depth positioning to be done with the start of the profile, set this question to <1>. For machines that require it at a Z positioning move, use setting <0>.

Recommended Setting: Start with <0> if you do not know your machine's requirement.

Related Settings: See template sections @STPROF (page 5-19) and @ZDPTHMV (page 5-21) and question 83.

Cutter Compensation

120. Code tool center*Choices:*

<0> No, code part profile

<1> Yes, code for the center of the tool

<2> Code the tool edge (lathes only)

Explanation: Indicate how SmartCAM should generate code in relationship to the tool.

Recommended Setting:

<1> In most cases, generate code for the tool's centerline with cutter compensation (question 121) for tool size and wear adjustment.

<2> For some lathes the tool edge (virtual tip) of the lathe insert is coded.

Related Settings: SmartCAM correctly calculates the offset for tool path that when coded for part profile causes errors on your controller. If you choose to code the part profile, be sure you understand the limitations of your machine's cutter-compensation feature. For punches, see question 233.

121. Turn on cutter compensation command with*Choices:*

<0> With rapid move to start of profile

<1> With 1st feed move of a profile

<2> Add a lead-in move and start command with this move

Explanation: Indicate when your machine looks for a cutter-compensation command. Most machines with cutter compensation require a move prior to the start of a profile, to establish the cutter compensation. Your setting indicates whether this happens as a rapid move or a feed move or you want SmartCAM to add a perpendicular lead-in move automatically.

Related Settings: See question 122.

122. Auto Lead-in distance

Machine Types: All except lathe

Explanation: Specify the length of the perpendicular lead-in line you want SmartCAM to insert at the beginning of offset profiles. This value applies only if question 121 is set to <2>. For example, enter 0.05 if you want SmartCAM to add a 0.05 lead-in move at the start of all profiles to establish cutter compensation.

Recommended Setting: This value should be greater than the maximum compensation value that you will use, but small enough to avoid cutter interference with other geometry when the machine starts cutting the profile.

123. Cancel - Cutter comp code

Explanation: Enter the command code your machine uses to turn off (cancel) cutter compensation. For example, some machines use G40 to turn off cutter compensation.

Related Settings: See template word #DCOMP on page 6-6.

124. Left - Cutter comp code

Explanation: Specify the command code your machine uses for identifying a left offset of the tool. For example, some machines use G41 for this function. The orientation of left or right is based on looking at the tool from behind (the tool moving away from you).

Related Settings: See template word #DCOMP on page 6-6.

125. Command code for right cutter comp

Explanation: Specify the command code your machine uses for identifying a right offset for the tool. For example, some machines use G42 for this function. The orientation of the left or right side is based on looking at the tool from behind (the tool moving away from you).

Related Settings: See template word #DCOMP on page 6-6.

126. Use sign with Tool Width to specify offset direction (#TLWD)*Choices:*

<0> No
<1> Yes

Machine Types: Mill, router, laser, 2-axis wire EDM, punch, contour machine, 4-axis wire EDM

Explanation: Indicate whether your machine needs to use a sign when you designate tool width. This is used to indicate tool offset direction as left or right as you look at the tool from behind (the tool moving away from you). For example, indicate left with a minus sign (-) and right with a plus sign (+).

127. Reverse sign to be used with #TLWD*Choices:*

<1> No
<-1> Yes

Machine Types: Mill, Router, laser, 2-axis wire EDM, punch, contour machine, 4-axis wire EDM

Explanation: Indicate whether your controller needs to use a sign when you designate tool width. This is used to indicate tool offset direction as left or right as you look at the tool from behind (the tool moving away from you). For example, indicate left with a minus sign (-) and right with a plus sign (+).

Related Settings: This setting works in conjunction with question 126 to reverse the assignment of the plus or minus sign relative to the offset direction.

128. TLWD value to use for NO Offset

Machine Types: Mill, router, laser, 2-axis wire EDM, punch, contour machine, 4-axis wire EDM

Explanation: Some machines that use the tool width to indicate the desired cutter offset will not accept a zero value if you wish to cut the center of a profile with no offset. This value gives the minimum offset amount to specify for no offset.

129. Add a corner rounding compensation command*Choices:*

<0> No
<1> Yes, add an extra block for outside corners with cutter diameter compensation

Explanation: Indicate whether your machine requires a corner-rounding compensation command. This question is used with older controllers, such as the Fanuc 3000C, that require a special corner-rounding command on outside corners. If you use setting <1>, SmartCAM uses the @CORNER section for any outside corners and calculates new #XOV, #YOV offset vectors.

130. Create compensation vectors as

Choices:

<0> Normal angle, perpendicular to the move direction

<1> Tangent angle, in the direction of the move

Explanation: Indicate how your machine wants SmartCAM to calculate the vectors for cutter compensation. Offset vectors are usually perpendicular (normal) to the cutting move.

131. Treatment of sharp corners

Choices:

<0> Roll tool on outside sharp corners

<1> Split element for slowdown move before corners

Machine Types: Router, laser, punches with torch, contour machine

Explanation: Indicate how SmartCAM should generate code for sharp corners. This controls the offset tool path creation by asking whether SmartCAM should create an arc on the outside of a sharp corner or extend the offset tool path elements so that they intersect. If you use setting <1>, SmartCAM will not create an arc for corners less than the amount specified in question 132. Instead, SmartCAM will extend the offset tool path to where they intersect, split the line elements of the corner (question 133) and set the extended data flag #slowdown to 1 so that it can be tested in the extended data file.

Related Settings: Question 132 determines the maximum angle SmartCAM will consider to be sharp. With setting <1>, use question 133 to specify the allowable slowdown distance.

132. Maximum inside element angle for a sharp corner

Machine Types: Router, laser, punches with torch, contour machine

Explanation: Specify the largest angle that SmartCAM should use when generating code for sharp corners (question 131). Enter an angle between 5 and 180 degrees.

133. Multiple of tool diameter for slowdown distance

Machine Types: Mill, Router, Laser, punches with torch, contour machine

Explanation: Enter the multiple of the tool diameter SmartCAM should use for determining the length of a slowdown move into a sharp corner (question 131). Enter a value between 0 and 10. This only applies to sharp corners comprised of line elements with lengths longer than two times the slowdown distance. SmartCAM will treat elements shorter than this as a single slowdown move.

135. TTS, cutter compensation type*Choices:*

<0> Use cutter comp commands

<1> Use comp vectors

<2> Use sign with tool width specification for direction and amount

Machine Types: Mill, router, laser, 2-axis wire EDM, lathe, punch, contour machine

Explanation: Indicate the type of cutter compensation Tape-to-Shape should use to build the original part if the part was coded for the tool centerline. Other items, such as question 120, are also considered.

Related Settings: See question 120.

136. TTS, activate cutter compensation for*Choices:*

<0> Initial compensation move has no offset

<1> Make offset prior to first move

Machine Types: Mill, router, laser, 2-axis wire EDM, lathe, punch, contour machine

Explanation: Indicate where your machine places the cutter-compensation command in relation to the profile. Some controllers activate the offset with the first feed move, while others make the offset before the move begins.

Cycle Time

138. Machine X axis rapid traverse feed rate

Explanation: If your machine supports this option, specify a feed rate for the X axis.

139. Machine Y axis rapid traverse feed rate

Explanation: If your machine supports this option, specify a feed rate for the Y axis.

140. Machine rapid traverse feed rate

Explanation: Specify the rapid traverse feed rate for all axes of your machine (as specified in the machine's manual). This is the Z-axis rapid-transverse feed rate if you use questions 138 and 139. SmartCAM uses this value in determining cycle times with Show Path. It has no affect on code output.

141. Tool change time

Explanation: Enter the average tool-change time for your machine, in minutes. For example, a setting of 0.25 would be a 1/4 of a minute (15 seconds). SmartCAM uses the value for calculating cycle times with Show Path; it does not affect code output.

142. Block time

Explanation: Enter the minimum time in minutes for positioning moves on your machine. Take into consideration acceleration, deceleration, and, in the case of a punch press, the punch and retract time. SmartCAM adds this time to each block when calculating cycle time for Show Path; it does not affect code output.

143. Maximum spindle RPM

Explanation: Enter the maximum spindle speed allowed for your machine. SmartCAM uses this value for calculating cycle times during constant surface speed cycles on lathes when the RPM change may exceed the maximum. This does not affect code output.

144. Cycle time adjustment factor

Explanation: Enter a general adjustment factor to adjust the estimated cycle time calculated by Show Path to match actual run times of your machines more closely. For example, entering 0.90 adjusts the cycle times by a factor of 0.9 (a 10% reduction). This value does not affect code output.

Mill and Lathe Peck Drill Cycle Parameters

150. Initial default peck value (#PECK)

Machine Types: Mill, router, lathe

Explanation: Specify the default peck drill cycle increment for your machine. For example, entering 0.1 results in a default increment of 0.1 for peck drill cycles. #PECK is read from the machine file any time it is called in the template field, and so is not restricted to cycle use. #PECK may also be set by a user command.

Recommended Setting: If this question is set to a known value, such as 0.5, SmartCAM can easily calculate #PECK based on tool diameter within the @TLCHG template section by using a formula such as #EVAL(#V1=#PECK*#TLDIA).

Related Settings: See #PECK on page 6-18.

Related Settings: In the template file reference section.

151. Generate individual moves for peck drill

Choices:

<0> No, use fixed cycle

<1> Yes

Machine Types: Mill, router, lathe

Explanation: Indicate whether you want to use fixed cycles or individual moves for peck drills. A setting of <1> overrides the use of fixed cycles for peck drill holes (refer to question 160). A setting of <0> calls the @FXD5 template section. You activate peck drill cycles when creating hole operations. SmartCAM starts the peck increment calculations at the Z check (#ZCHK) distance above the hole's end value.

Recommended Setting: Use <0> if the machine has a peck cycle.

Related Settings: Be sure #PECK is set to 0. For the <0> setting, SmartCAM sets #TLOP to 5 and accesses it in the template file.

152. First peck depth factor

Machine Types: Mill, router, lathe

Explanation: Enter a factor for SmartCAM to apply to the first feed move for a peck drill question. For example, a value of 2.0 makes the first peck drill two times deeper than the #PECK increment.

Related Settings: You must set question 151 to <1> for this setting to be available. If you set question 151 to <0>, a no-motion set of code (G01 and G00 on separate lines) is output for the first peck.

153. Peck retract amount

Machine Types: Mill, router, laser, lathe

Explanation: Enter the amount of retraction you want for each peck increment. If you enter a retraction amount of 0.0, the drill will return to the full retract position each time. For a value of 0.1 the drill retracts 0.1 after each peck feed move.

Related Settings: You must set question 151 to <1> for this option to be available.

154. Peck return clearance amount

Machine Types: Mill, router, laser, lathe

Explanation: Enter the amount of clearance you want between the drill and last peck depth before the drill begins to feed again. You must set question 151 to <1> for this option to be available. The value for this item should be less than the value you specify in question 153.

Recommended Setting: Set this question to 0.02 or greater.

Mill Cycles

160. Use Fixed cycles for holes

Choices:

<0> No

<1> Yes

Machine Types: Mill, router, lathe

Explanation: Indicate whether you want to use code for fixed cycles for hole operations. Many CNC controllers provide fixed cycles for hole operations such as drilling, tapping, and boring. If you set this question to <0>, SmartCAM does not access the @FXD section in the template file.

Recommended Setting: Set this question to <1> to use fixed cycles if available.

161. Fixed cycle cancel code

Machine Types: Mill, router

Explanation: Enter the command code your machine uses to cancel a fixed cycle. For example, some machines use G80 to cancel fixed cycles.

Recommended Setting: G80

Related Settings: This function is available when question 160 is set to <1>.

162. Drill fixed cycle code

Machine Types: Mill, router

Explanation: Enter the command code your machine uses for specifying drilling operations. For example, some machines use G81 for this function when using a drill or reamer.

Recommended Setting: G81

Related Settings: This option is available when question 160 is set to <1>. #TLOP=1 when a drill or reamer is used.

163. Spot drill with dwell fixed cycle code

Machine Types: Mill, router

Explanation: Enter the command code for a dwell during a fixed cycle for spot facing. For example, some machines use G82 for this function when using spot drills such as counterbore and countersinks.

Related Settings: This option is available when question 160 is set to <1>. Use the #DWELL factor from question 50. In this case the #DWELL factor is not added to estimated cycle time. The maximum #DWELL is 99. Used by end mill, ball mill, counterbore, countersink, and spot drill when #TLOP=2. #DWELL may also be set by a user command for various dwell valves in SmartCAM.

164. Tap fixed cycle code

Machine Types: Mill, router

Explanation: Enter the command code your machine uses for a tapping fixed cycle. For example, some machines use G84 for the function.

Related Settings: This option is available when question 160 is set to <1>. When canned cycles are not used, SmartCAM treats the hole as a drill cycle. Only Z motion is output, with no affect on spindle rotation.

Recommended Setting: G84

165. Bore fixed cycle code

Machine Types: Mill, router

Explanation: Enter the command code your machine uses for a boring fixed cycle. For example, some machines use G86.

Recommended Setting: G86

Related Settings: This option is available if question 160 is set to <1>. When the machine is not using G86, the tool rapid retracts from the bottom of the hole. #TLOP=4 when a boring tool is used.

166. Peck drill fixed cycle code

Machine Types: Mill, router

Explanation: Enter the command code your machine uses for a peck drill fixed cycle. For example, some machines use G83 for this option.

Recommended Setting: G83

Related Settings: Question 151 can override this setting and result in individual moves. If G83 is used, the answers to questions 152, 153, and 154 are ignored.

170. Fixed Cycle cmds specify the 'Z' check plane

Choices:

<0> No

<1> Yes

Machine Types: Mill, router

Explanation: Indicate whether you want your fixed cycle code to specify the Z-check (ZCHK) position of your tool. ZCHK is the Z level where rapid feed stops and cutting feed rate begins.

Recommended Setting: <1> uses #ZCHK in @FXD—typically, the R word in the canned cycle.

Related Settings: If this questions is set to <0>, SmartCAM outputs a line of code specifying a Z-level retraction before each fixed cycle statement. If canned cycles are not used, SmartCAM outputs the Z level anyway. See question 83.

171. Check plane 'Z' position specified as*Choices:*

- <0> Absolute Z position regardless of positioning mode
- <1> Incremental Z position when in incremental mode

Machine Types: Mill, router

Explanation: Indicate how your machine uses the #ZCHK value. Most machines base ZCHK on the active work plane (incremental mode). There are some machines that specify ZCHK as the absolute distance from the part's Z0.

172. Fixed cycle Depth 'Z' specified as*Choices:*

- <0> Absolute position
- <1> Incremental depth including sign
- <2> Unsigned incremental depth
- <3> Absolute or incremental based on positioning mode

Machine Types: Mill, router

Explanation: Indicate what method your machine uses for arriving at a Z depth for a fixed cycle. The z word in the canned cycle typically contains the depth value.

Related Settings: The answer to this question is in effect even if you do not use canned cycles (see question 160).

173. After fixed cycle 'Z' position returns to*Choices:*

- <0> Z_Check
- <1> Z_Clear
- <2> Switchable using #RTNLVL commands. See .smf questions 174 & 175

Machine Types: Mill, router

Explanation: Indicate what retraction position your machine returns to after a drill cycle. If your controller returns to the #ZCHK, SmartCAM automatically inserts a rapid Z move to the Z-clear location specified in the model before making any X,Y moves.

174. Fixed cycle return to R level (Z_Check)

Machine Types: Mill, router

Explanation: If question 173 is set to <2>, enter the command code your machine uses to return to the R plane level (the profile top) at the end of a fixed drill cycle. For example, some machines use G99 for this setting.

175. Fixed cycle return to initial level (Z_Clear)

Machine Types: Mill, Router, laser

Explanation: If question 173 is set to <2>, enter the command code your machine uses to return the tool to the initial Z-clear level after a fixed drill cycle. For example, some machines use G98 for this option.

Lathe Cycles

180. Use lathe canned cycle for turn and face

Choices:

<0> No

<1> Yes

Explanation: Indicate if you want to use your lathe's canned cycle for rough turning operations. SmartCAM supports your canned cycle for turning or outputs the individual lines of code to rough or face a part. The lathe turning cycle must be a single-pass vertical or horizontal cycle.

Recommended Setting: Set this question to <1> if your machine supports canned cycles.

181. Turn cycle command code

Explanation: Enter the command code your machine uses to activate a canned turn cycle. For example, some machines use G90 for this option.

Related Settings: Question 180 needs to be set to <1>.

182. Face cycle command code

Explanation: Enter the command code your machine uses to activate a canned facing cycle. For example, some machines use G94 for this option.

Related Settings: Question 180 needs to be set to <1>.

Lathe Threading

185. Thread command type

Choices:

<0> Command for thread pass only, add rapid moves

<1> Thread cycle cmd. for each thread pass, includes rapid in cycle

<2> Single command generates entire thread cycle

Explanation: Indicate how SmartCAM should generate the geometry for the lathe thread cycle. The setting you use determines the following items:

- Which of the questions 188–195 you use
- How SmartCAM calculates some of the template words
- Whether the thread cycle is output using the **@FXD1** section or the **@RAP** and **@FXD1** sections

The following information identifies the impact each setting has on these items:

<0> For thread cycles (such as G33) that require the inclusion of rapid moves. This output produces the greatest amount of code. With this setting, the **@FXD1** section generates the thread pass only, and the **@RAP** section outputs the rapid moves in the following order:

1. **@RAP** - rapid to start of thread pass
 2. **@FXD1** - generate code for the thread pass
 3. **@RAP** - rapid move to pull away from the thread
 4. **@RAP** - rapid to position above start of thread
- Repeat 2–4 until thread is complete.

Threading template words have the following meanings in **@FXD1**:

#FTHRD	Thread lead
#XCTR	Starting X value for <i>last</i> thread pass of cycle
#ZCTR	Starting Z value for <i>last</i> thread pass of cycle
#XPASS	Ending X value for <i>last</i> thread pass of cycle
#ZPASS	Ending Z value for <i>last</i> thread pass of cycle
#XOV	Change in X (delta X) between start and end of <i>last</i> thread pass (for tapered threads)
#ZOV	Change in Z (delta Z) between start and end of <i>last</i> thread pass (for tapered threads)
#V0	Value of thread crest X dimension
#V1	Current depth of thread; absolute value, updated for each pass
#V2	Thread depth remaining; absolute value, updated for each pass
#V3	Incremental depth move to each pass depth for individual passes or to first pass depth for fixed cycle.
#V4	Number of passes at first pass depth that will fit.
#XST	Start X of thread; actual value, updated for each pass
#ZST	<i>not active</i>

<1> For thread cycles (such as G38) where each thread pass must be output but rapid moves do not need to be output with each pass. **@FXD1** is called once for each pass of the thread as follows:

1. **@RAP** - rapid to position above start of thread
 2. **@FXD1** - generate code for first thread pass
 3. **@FXD1** - generate code for next thread pass
- Repeat 3 until thread is complete.

Threading template words have the following meanings in **@FXD1**:

#FTHRD	Thread lead
#XCTR	Starting X value for <i>each</i> thread pass of cycle
#ZCTR	<i>not active</i>
#XPASS	Ending X value for <i>each</i> thread pass of cycle
#ZPASS	Ending Z value for <i>each</i> thread pass of cycle
#XOV	Change in X (delta X) between start and end of <i>last</i> thread pass (for tapered threads)
#ZOV	Change in Z (delta Z) between start and end of <i>last</i> thread pass (for tapered threads)
#V0	Value of thread crest X dimension
#V1	Current depth of thread; absolute value, updated for each pass
#V2	Thread depth remaining; absolute value, updated for each pass
#V3	Incremental depth move to each pass depth for individual passes or to first pass depth for fixed cycle.
#V4	Number of passes at first pass depth that will fit.
#XST	Start X of thread; actual value, updated for each pass
#ZST	<i>not active</i>

<2>For thread cycles (such as G76) where the machine generates an entire thread cycle based on a single set of parameters, where each is not explicitly stated in the code. **@FXD1** is called only once per thread cycle as follows:

1. **@RAP** - rapid to position above start of thread
 2. **@FXD1** - output all parameters for the thread cycle
- Thread cycle is complete.

Threading template words have the following meanings in **@FXD1**:

#FTHRD	Thread lead
#XCTR	Starting X value for <i>last</i> thread pass of cycle
#ZCTR	Starting Z value for <i>last</i> thread pass of cycle
#XPASS	Ending X value for <i>last</i> thread pass of cycle
#ZPASS	Ending Z value for <i>last</i> thread pass of cycle

#FTHRD	Thread lead
#XOV	Change in X (delta X) between start and end of <i>last</i> thread pass (for tapered threads)
#ZOV	Change in Z (delta Z) between start and end of <i>last</i> thread pass (for tapered threads)
#V0	Value of thread crest X dimension
#V1	Absolute value of depth of <i>last</i> thread pass
#V2	Absolute value of depth of <i>first</i> thread pass
#V3	Incremental depth move to each pass depth for individual passes or to first pass depth for fixed cycle.
#V4	Number of passes at first pass depth that will fit.
#XST	Start X of <i>first</i> thread pass (includes rapid clear value)
#ZST	<i>not active</i>

If you use setting <0> or <1>, you must set questions 188–195. If you use setting <2>, questions 188–195 will be ignored. The selected setting changes how certain template words used to define the thread cycle are calculated.

186. Thread cycle command code

Explanation: Enter the command code your machine uses for thread cycles. For example, some lathes use G33 for this option.

Related Settings: This command code is output in the @FXD1 section by the template word #FXD.

187. Thread lead feed rate format

Explanation: Enter the numeric format your lathe uses for the thread lead. (See page 3-2 for information about numeric formats.) The thread lead is output by the template word #FTHRD in the @FXD1 template section.

Related Settings: The value of the thread lead is set in the model file when you define a thread. The template word #FTHRD must occur in the @FXD1 section of the template file.

188. Thread compound infeed angle

Explanation: Specify the infeed angle your machine uses for cutting a thread. For example, a value of 30.0 specifies that the start position for each thread pass is fed at an angle of 30 degrees.

Recommended Setting: 29 or 30 degrees is generally used.

Related Settings: Question 185 must be set to <0> or <1>.

189. Number of factored thread passes

Explanation: Specify the number of passes to be made using a factor value before constant volume passes begin. For example, if you enter 4, SmartCAM calculates four passes using the depth factor you specified in question 190 before starting constant volume passes.

Recommended Setting: A value of 4 is a good starting point. If you want SmartCAM to calculate all passes using the depth factor (no constant volume passes), set this to a large number (for example, 100).

Related Settings: Question 185 must be set to <0> or <1>. Question 190 sets how SmartCAM calculates the factored passes. The depth of the first thread pass is set in the model when you define the thread. SmartCAM uses this value for calculating successive passes.

190. Thread pass depth factor

Explanation: Enter the factor you want SmartCAM to use when calculating the depth of each thread pass. In the model, you enter the depth of the first pass of the threading cycle. SmartCAM reduces the depth of cut for the next number of passes (set by question 189) by the factor you enter. For example, if you enter 90, SmartCAM reduces each pass by 10% of the previous pass. If you do not want any change in depth of cut, set the factor to 1. You can use any number greater than 0 and equal to or less than 1.

Recommended Setting: Start with 0.9, and experiment if this factor is too high.

Related Settings: Question 185 must be set to <0> or <1>. Question 189 sets the maximum number of thread passes for which this item is active. SmartCAM uses the first pass depth value for determining successive pass depths.

191. Constant volume thread exponent

Explanation: Enter the exponent you want SmartCAM to use for maintaining constant volume for threading. Use constant volume for threading to maintain an even volume of material on the tool nose as the tool moves deeper into the material. For example, if you enter 0.5, SmartCAM reduces each successive thread pass by 50%. Use any number less than or equal to 1.

Constant volume is effected after the number of factored thread passes (question 189) has been executed. Question 185 must be set to <0> or <1>.

Recommended Setting: Begin with the default, 0.5. You may want to adjust this value to control the finish cut.

192. Minimum thread pass depth allowed

Explanation: Enter the smallest depth of cut you want to use when cutting for threads. For example, entering 0.001 would result in no thread pass depths smaller than 0.001.

Recommended Setting: Use 0.001 or the minimum movement (greater than 0) your machine is capable of.

Related Settings: Question 185 must be set to <0> or <1>.

193. Final thread pass depth

Explanation: Enter the final threading pass depth that you want to use. For example, entering 0.002 results in a final depth pass greater than 0.002 and less than 0.003 plus the minimum from question 92.

Recommended Setting: Use 0.001 or the minimum movement your machine is capable of.

Related Settings: Question 185 must be set to <0> or <1>.

194. Use sign with depth template words #V1 & #V2

Choices:

<0> No, keep #V1 & #V2 unsigned

<1> Yes

Explanation: Indicate if you want a sign to be sent with template words #V1 and #V2. These words contain information about the depth of the thread. See question 185.

Related Settings: Information that #V1 and #V2 contain depends upon the setting of question 185.

195. Specify 'X' axis thread crest and root dimensions as

Choices:

<0> Radius

<1> Diameter

Explanation: Indicate whether the X values output by the lathe thread section @FXD1 are radius or diameter values.

Lathe Groove Cycle

200. Center first groove pass

Choices:

<0> No

<1> Yes

Explanation: Indicate whether you want the groove tool to start in the center of the groove and work out to each side.

201. Add a finish pass across bottom of groove*Choices:*

<0> No

<1> Yes

Explanation: Indicate whether you want a finish pass at the bottom of a groove. When this question is set to <1>, SmartCAM adds an additional code line that feeds the grooving tool along the bottom of the groove.

202. Side of groove, rapid clearance amount

Explanation: Enter the distance you want the machine to move the grooving tool from the wall of the groove while rapidly retracting and the tool is inside the groove. For example, a value of 0.01 clears the side of the groove by 0.01 when retracting from inside the groove.

Recommended Setting: 0.01

Related Settings: Question 201 needs to be set to <1>.

203. Use the machine canned cycle for grooves*Choices:*

<0> No, generate each move for groove

<1> Yes

Explanation: Indicate whether you want to use the machine's canned groove cycle. If the lathe uses a canned cycle for grooving, entering <1> outputs that cycle using parameters in the @FXD6 and @FXD7 sections of the template file.

Related Settings: The controller must support this cycle with parameters of SmartCAM.

Lathe 4-Axis

210. Is 4-axis simultaneous motion possible with this control*Choices:*

<0> No

<1> Yes

Explanation: Use setting <1> if this machine is capable of 4-axis simultaneous operation. This enables the 4-Axis Synch toolbox's modeling tools in SmartCAM turning applications. If you use setting <0>, SmartCAM will ignore any 4-Axis Synch commands in the model.

211. What is the angular distance between the X-axis slides of each turret

Explanation: Use this setting in 4-axis mill-turn applications to identify how far the C axis must rotate to present the same part orientation to each turret. For example, to cross-drill a hole with the upper turret and then tap that same hole with the lower turret, how far does the C axis rotate? A setting of 180 means the turrets are 180 degrees apart.

Enter 0 for controllers that use a code to identify which turret is active and to compensate automatically for the angular spacing of the turrets (for example, the Okuma OSP5000).

212. How are multiple turrets coded

Choices:

<0> Separate NC programs—one for each turret with timing codes (e.g., Fanuc 11tta). Without timing codes, this type executes in simultaneous mode

<1> One NC program—sequence is controlled by timing codes (e.g., Okuma OSP5000). Without timing codes, this control executes in the order blocks are presented

<2> One NC program—sequence is controlled by the order of blocks in the NC code, no timing codes are used

<3> One NC program—same as <2>, but NC code blocks from each turret are never merged (e.g., G&L 8000)

Explanation: This is for 4-axis dual slide lathes. Identify how the controller expects to see NC code for each turret:

☞ Settings <0> and <1> require completion of questions 216–219.

<0> Code for each turret is sent to a separate program, with SmartCAM accessing the @Start and @End template sections for each turret. One SmartCAM code file contains both of the separate programs. SmartCAM generates the same timing codes for both turrets to ensure a sequence of operations consistent with the model. Required program numbers (for example, O1000 for Fanuc) must be incorporated into the template file.

<1> Code for each turret is sent to a single program, with timing codes controlling the sequence. @Start template section is accessed for the first turret, and @End is accessed for the last turret in the timing sequence. For simultaneous motion, SmartCAM generates matching timing codes for both turrets.

<2> A single NC code file is produced, with the program blocks arranged as necessary to control the timing between the turrets. You control the block numbering for this option with questions 10, 12, 13, and 14 in the machine file for turret 1 only. SmartCAM ignores turret 2 machine file settings for these questions. #NEXT and #EXIT do not work with this setting.

When generating code for alternating block type machines (settings <2> and <3>), SmartCAM creates an interim file named *code-file.__1* (where *codefile* is the name of your output code file). SmartCAM does not delete this file automatically. Manually delete these files at any time or automatically remove them by placing a command in the *aturn32.bat* file prior to *scloop*. For example, you could use the following command if your code files are stored in the *atdata* subdirectory of SmartCAM Path:

```
del \SmartCAM Path\atdata\*.__1
```


213. How is the spindle controlled during simultaneous motion*Choices:*

<0> Each turret uses its own spindle control codes

<1> Spindle control output to Primary turret only

<2> Spindle control for Secondary turret must match Primary turret

Explanation: This setting controls the output of spindle control codes (for example, speed mode, speed, and direction) while performing simultaneous 4-axis operations. When you use the <0> setting, SmartCAM outputs the spindle control code you enter. Settings <1> and <2> are similar, except that <2> causes SmartCAM to output code for both turrets. For Fanuc and Cincinnati 950 controls, this should be set to <1>; for Okuma, <2>. Complete questions 214 and 215 below.

 Settings <0> and <1> require completion of questions 216–219.

214. What is the code output to give Turret 1 control during CSS operation

Explanation: Enter the code that transfers spindle control to turret 1 (for example, G110). This code is output by the template word #PRMRY.

215. What is the code output to give Turret 2 control during CSS operation

Explanation: Enter the code that transfers spindle control to turret 2 (for example, G111). This code is output by the template word #PRMRY.

216. Format for timing codes

Explanation: Enter the numeric format your machine uses for timing codes. (See page 3-2 for information about numeric formats.) For example, enter T3.0 for Fanuc and Cincinnati 950 controllers, which use M100 to M999 for timing codes. Enter T4.0 for Okuma controllers, which use P0001 to P9999.

217. Start timing codes with

Explanation: Specify the value of the first timing code output in the CNC code (uses template word #TCODE). Fanuc, for example, uses M100 to M999 for timing codes; enter 100 here if your machine is equipped with a Fanuc.

218. Increment between timing codes

Explanation: Specify the difference between equipped codes (use the template word #TCODE). For example, if you set this question to 10, each timing code will be 10 higher than the one before it. It is a good idea to set this to a value greater than 1 to allow the operator to enter additional timing commands at the CNC controller, if required.

219. Maximum timing code

Explanation: Enter the highest number you want to use for timing codes. For example, if you set this to 999, SmartCAM will restart the timing code numbering after reaching 999. The next number will be the starting value from question 217.

Punch Cycles

220. Use Punch On/Off commands

Choices:

<0> No

<1> Yes

<2> No punch On command, inhibit punch command for all punch Off moves

Machine Types: Punch, contour machine

Explanation: Indicate whether you want to code for punch press on/off commands. Setting <2> indicates that your machine assumes the punch is on unless you output code to turn it off for that block.

221. Default Punch On/Off at start of program

Choices:

<0> Punch Off

<1> Punch On

Machine Types: Punch, contour machine

Explanation: Indicate whether you want your code to assign a default punch-on or punch-off command at the beginning of a program.

222. Primary Punch Off code

Machine Types: Punch, contour machine

Explanation: This is the code (for example, M85) that turns off the punch mode on a CNC punch press.

Related Settings: Question 220 needs to be set to <1> for this option to be valid. See #PUNCH on page 6-19.

223. Primary Punch On code

Machine Types: Punch, contour machine

Explanation: This is the code (for example, M75) that activates the punch mode on a CNC punch press.

Related Settings: Question 22 needs to be set to <1> for this option to be valid. See #PUNCH on page 6-19.

224. Primary Inhibit Punch code

Machine Types: Punch, contour machine

Explanation: This code (for example, G70) must be used to inhibit the punch in every block where punching is not desired. Use of this code is based on item #220, setting <2>.

Related Settings: Question 220 must be set to <2> for this option to be valid.

225. Secondary Punch (Attachment) Off code

Machine Types: Punch, contour machine

Explanation: This item is used for punch machines with an optional drilling or tapping head. Enter the code (for example, M23) that turns the attachment off if it is currently used. When the attachment is inhibited, punching is activated.

Related Settings: Question 220 must be set to <1> for this setting to be valid.

226. Secondary Punch (Attachment) On code

Machine Types: Punch, contour machine

Explanation: This item is used for punch machines with an optional drilling or tapping head. Enter the code (for example, M25) that turns the attachment option on if it is currently not used. When the attachment is activated, punching is inhibited.

Related Settings: Question 220 must be set to <1> for this setting to be valid.

227. Secondary Inhibit Punch (Attachment) code

Machine Types: Punch, contour machine

Explanation: This item is used for punch machines with an optional drilling or tapping head. Enter the command code (for example, M28) your machine uses for inhibiting the punch while using an attachment. When the attachment is activated, punching is inhibited in every block where this code occurs.

Related Settings: Question 220 must be set to <2> for this setting to be valid.

230. Use Torch for primary contour command

Choices:

<0> No

<1> Yes

Machine Types: Punch, contour machine

Explanation: This item is used for contour machines and punches with a torch attachment. It is used to control when the torch will be turned on and off. It does this by controlling the value of the template word #CUTC.

Recommended Setting: Set to <0> if the primary contour tool is a punch tool. Set to <1> if the primary contour tool is a plasma arc or torch. If you are using a punch/contour combination, include the @TORCH and @PCNCHTL sections of the template file.

Related Settings: The contour off/on codes must be set in questions 110 and 111. The template word #CUTC should appear in the @STPROF, @ENDPROF, and @END sections of the template file.

231. Use nibble Line cycle

Choices:

<0> No, generate individual hits
<1> Yes

Machine Types: Punch, contour machine

Explanation: If the machine is a punch press, you can specify whether you want to be able to program for a nibble line cycle or just for individual punch hits.

Related Settings: See question 90 for a similar question about using the arc cycle.

232. Generate individual hits to nibble profiles

Choices:

<0> No, use nibble cycle commands
<1> Yes, generate hits for punch tool profiles

Machine Types: Punch, contour machine

Explanation: This item is used for burn profiles that are cut with contour commands. Nibble profiles are nibbled with individual hits. Indicate if you want to explode profiles into individual hits for punch tools. When you use setting <1>, SmartCAM outputs individual hits for profiles based on the IPH setting in the job operations file. If you are using a contour machine, SmartCAM outputs line and arc codes.

233. Generate code for centerline of punch tools (Regardless of question #120 part profile setting)

Choices:

<0> No, code part profile per item #120
<1> Code tool center for square or rectangle tools
<2> Code tool center for all punch tools

Machine Types: Punch, contour machine

Explanation: Indicate how you want SmartCAM to generate code for the centerline of punch tools. Use <1> for punch machines that provide for cutter offset of round punch tools but not for square or rectangular punches. Use <2> to code for tool centerline when nibbling a profile using a punch tool when cutting a profile with a torch or attachment.

Related Settings: Torch and attachment tools use question 40 for output feed rate.

234. Change the nibble increment for rectangle tools*Choices:*


<0> No

<1> Yes, adjust the nibble increment used for the short side of a rectangle punch

Machine Types: Punch, contour machine*Explanation:* Indicate if you want SmartCAM to adjust the feed rate proportionally for the short side of rectangle punches.**235. Maximum distance between nibble hits***Machine Types:* Punch, contour machine*Explanation:* Specify the maximum distance between nibble hits your machine allows. If the feed rate in the job operations file is greater than the value you enter, code is output to turn off nibble mode and turn on punch mode instead.*Related Settings:* See question 239.**236. Minimum line length allowed for shearproof cycle***Machine Types:* Punch, contour machine*Explanation:* Specify the minimum line length your machine allows for shearproof cycles. SmartCAM calculates the length by using the value you enter as a proportion of the tool length. If the line length is too short, SmartCAM inserts individual punch hits instead of using the machine cycle.**237. Nibble hits per minute using nibble mode for a Punch machine***Machine Types:* Punch, contour machine*Explanation:* Enter the number of hits per minute your punch press makes in the nibble mode. SmartCAM uses this value for calculating the estimated cycle time and does not affect code output.**238. Include start position with the nibble cycle***Choices:*

<0> No

<1> Yes, Each nibble cycle will indicate start location

Machine Types: Punch, contour machine*Explanation:* Indicate whether you want to include the start position for a nibble cycle in the cycle's command. This gives you the option of specifying the cycle-start position as part of the @LINE or @ARC cycle with the #XST and #YST words instead of using the @STPROF or @RAP section. Do not use @STPROF or @RAP to move to the cycle-start location.

239. Nibble mode type*Choices:*

<0> Machine uses the same line and arc commands, it just turns nibble mode On or Off

<1> Machine uses different line and arc commands when nibble mode is on versus when it is off

<2> Generate individual hits for nibble mode off

Machine Types: Punch, contour machine

Explanation: Indicate how your machine indicates a change of nibble mode. Most punch machines can nibble at one hit rate. When you specify a longer rate, they must turn off the nibble mode and switch to a slower punch rate.

Related Settings: Questions 240 and 241 apply when you use a <0> setting. Questions 243, 244, and 245 apply when you use a <1> setting.

240. Nibble mode Off code

Machine Types: Punch, contour machine

Explanation: Enter the command code your machine uses to turn nibble mode off. For example, some punch presses use M20 for this option.

Related Settings: This setting is available only when question 239 is set to <0>.

241. Nibble mode On (normal) code

Machine Types: Punch, contour machine

Explanation: Enter the command code your machine uses to turn nibble mode on. For example, some punch presses use M22 for this option.

Related Settings: This setting is available only when question 239 is set to <0>.

243. Linear cutting move code

(Punch - nibble mode Off)

Machine Types: Punch, contour machine

Explanation: Enter the command code your machine uses for linear punch moves with nibble mode off. For example, some punch presses use G21 punch line code for this option.

Related Settings: This setting applies only when question 237 is set to <1>.

244. Clockwise circular cutting code

(Punch - nibble mode Off)

Machine Types: Punch, contour machine

Explanation: Enter the command code your machine uses for clockwise circular punch moves with nibble mode off. For example, some machines use G22 punch arc code for this setting.

Related Settings: This setting applies only when question 239 is set to <1>.

245. Counterclockwise circular cutting code

(Punch - nibble mode Off)

Machine Types: Punch, contour machine

Explanation: Enter the command code your machine uses for counterclockwise circular punch moves with nibble mode off. For example, some punch presses use G23 punch arc code for this setting.

Related Settings: This setting applies only when question 239 is set to <1>.

248. List of Punch auto-index tool station numbers

Machine Types: Punch, contour machine

Explanation: Enter a list of tool stations that you use for auto-indexing tools. You must separate tools with a comma (for example, 1,7,14,36) and use no more than 20 characters. Enter an asterisk (*) if all stations are auto-indexing. The tools will use the auto-index angle template word, #INDXA.

249. Allow uneven nibble hits at FEED spacing

Choices:

<0> No, subdivide nibble distance so that each hit is evenly spaced
<1> Yes, space nibble hits at #FEED distance between hits. Last hit may be uneven

Machine Types: Punch, contour machine

Explanation: Indicate whether you want to use uneven nibble hits at the end of feed spacing. A setting of <0> divides line length into even stepovers smaller than #FEED. A setting of <1> uses #FEED for all stepovers except for the last hit.

4-Axis Wire EDM

259. Maximum UV or QR axis travel

Machine Types: Wire EDM

Explanation: Enter the travel limits of the upper wire guide. When this limit is exceeded during Show Path or code generation, SmartCAM issues the following message: "Maximum upper guide travel exceeded."

260. Maximum allowable wire inclination angle

Machine Types: Wire EDM

Explanation: Enter the limit for the maximum wire angle allowed by your machine. Whenever this limit is exceeded during Show Path or code generation, SmartCAM issues the following message: "Inclination angle exceeds max allowed by machine."

261. Should inclination angle always be positive

Choices:

<0> No

<1> Yes

Machine Types: Wire EDM

Explanation: Some wire EDM machines do not allow negative inclination angles but instead use the wire-inclination commands specified in questions 262–264 to indicate the inclination direction. This setting affects the outputs of #QANG and #RANG.

262. Wire inclination Cancel command

Machine Types: Wire EDM

Explanation: Output by #WINCL(=0)

Example: G50

263. Wire inclination Left command

Machine Types: Wire EDM

Explanation: Output by #WINCL(=1)

Example: G51

264. Wire inclination Right command

Machine Types: Wire EDM

Explanation: Output by #WINCL(=2)

Example: G52

266. Absolute U,V axis commands*Choices:*

<0> Incremental: Program change in U,V axes relative to the previous U,V position

<1> Absolute: U,V axes values are relative to X,Y position only

Machine Types: Wire EDM

Explanation: Output form of #[UV]OV. An absolute U,V position is specified relative to the corresponding X,Y (end point) position in the same program block. An incremental U,V position is specified as the U,V position change from the last to the new position, where both the last and the new U,V positions are first calculated relative to the corresponding X,Y (endpoint) position. This is independent of the incremental or absolute positioning mode for the X,Y axes.

267. Zero length moves in 4-axis events*Choices:*

<0> Are permitted

<1> Are not permitted

Machine Types: Wire EDM

3-D Arcs and Helixes

☞ Only machine tools programming in 3-D will use 3-D-specific questions.

270. 3-D, program arc commands on orthogonal work planes*Choices:*

<0> No, arc command only on Tool_Plane

<1> Yes, also use arc command on planes orthogonal to Tool_Plane

Machine Types: Mill, router, laser, 2-axis wire EDM

Explanation: Set this to <1> to output arc commands on XY, XZ, and YZ arcs. All other nonorthogonal arcs will be output as line segments. Set this to <0> to output arc commands only for those arcs which lie in the XY plane. All others will be output as line segments.

271. 3-D, Arc Plane selection command for X,Y Plane

Machine Types: Mill, router, laser, 2-axis wire EDM

Explanation: This item specifies the XY plane selection command (typically G17) to be output when #PLANE is called in the template file.

Related Settings: See question 94 to switch the direction for the X,Y plane.

273. 3-D, arc plane selection command for X,Z plane

Machine Types: Mill, router, laser, 2-axis wire EDM

Explanation: This item specifies the XZ plane selection command (typically G18) to be output when #PLANE is called in the template file.

274. 3-D, switch arc direction on X,Z plane

Choices:

<1> Standard

<-1> Reversed

Machine Types: Mill, router, laser, 2-axis wire EDM

Explanation: Specify whether XZ arc directions are standard (clockwise) or reversed (counterclockwise).

Related Settings: See questions 92 and 93.

275. 3-D, arc plane selection command for Y,Z plane

Machine Types: Mill, router, laser, 2-axis wire EDM

Explanation: This item specifies the YZ plane selection command (typically G19) to be output when #PLANE is called in the template file.

276. 3-D, switch arc direction on Y,Z plane

Choices:

<1> Standard

<-1> Reversed

Machine Types: Mill, router, laser, 2-axis wire EDM

Related Settings: See also questions 92 and 93.

280. Use Helix command

Choices:

<0> No, interpolate Helix using line commands

<1> Yes

Machine Types: Mill, router, laser, 2-axis wire EDM

Explanation: Specifies whether helices should be coded as line segments (<0>) or helix commands (<1>).

Recommended Setting: If your controller will accept a command for helical interpolation, set this question to <1> and include an @XYHELIX section in your template file.

281. 3-D, program helix commands on orthogonal work planes*Choices:*

<0> No, helix command only on Tool_Plane

<1> Yes, also use Helix command on planes orthogonal to Tool_Plane

Machine Types: Mill, router, laser, 2-axis wire EDM

Explanation: This item specifies whether helices are allowed on the XZ and YZ planes.

Recommended Setting: If your controller allows XZ or YZ helices, set this question to <1> and modify the @XZHELIX and @YZHELIX sections of your template file.

Related Settings: Question 280 must be set to <1>.

282. 3-D, total angle allowed by a helix command*Choices:*

<0> Single quadrants of 90 degrees

<1> Up to one full revolution of 360 degrees

<2> Any number of turns

Machine Types: Mill, router, laser, 2-axis wire EDM

Explanation: This item specifies whether helices of more than one full turn are allowed.

Recommended Setting: if your controller will accept multiple turn helices, set this item to <2> and modify the @XYHELIX, @XZHELIX, and @YZHELIX sections of your template file.

Related Settings: Question 280 must be set to <1>. See also question 95.

283. Helix center location mode*Choices:*

<0> Always incremental distance

<1> Absolute location when in absolute positioning mode (G90)

<2> Always absolute location regardless of positioning mode

Machine Types: Mill, router, laser, 2-axis wire EDM

Explanation: This item establishes the method of coding the center of a helix (typically the same as arcs). See question 97 for incremental direction specifications for arcs.

Related Settings: Question 280 must be set to <1>.

284. Can spiral moves be coded with helical interpolation?*Choices:*

<0> No, helix must have a constant radius

<1> Yes

Machine Types: Mill, router, laser, 2-axis wire EDM

Explanation: This item specifies whether spiral moves will be output as line segments (<0>) or helix commands (<1>).

Related Settings: Question 280 must also be set to <1> if this item is set to <1>.

285. Clockwise helical cutting code

Machine Types: Mill, router, laser, 2-axis wire EDM

Explanation: This item specifies the command that will be output for #MOV when a clockwise helix is coded (for example, G02).

Related Settings: Question 280 must be set to <1>.

286. Counterclockwise helical cutting code

Machine Types: Mill, router, laser, 2-axis wire EDM

Explanation: This item specifies the command that will be output for #MOV when a counterclockwise helix is coded (for example, G03).

Related Settings: Question 280 must be set to <1>.

4- and 5-Axis Positioning

☞ Questions 289 to 338 pertain to SmartCAM Advanced products only.

289. 3-D, Tool retraction for rotational position specified

Choices:

<0> By placing a rapid point at the desired retract position

<1> As the Home Point defined by the first tool position in the model

<2> Retract to Z Home at current X,Y position

Explanation: Specify which of the following you want to happen:

<0> The user must insert a point prior to tool change.

<1> SmartCAM inserts a point at the same location as the position of the first tool in the database.

<2> SmartCAM outputs only a Z axis move at tool change.

The code output that results from the settings is as follows:

<0> Outputs and updates ZPOS, XPOS, and YPOS.

<1> Outputs but does not update XHOME, YHOME, and ZHOME.

<2> Outputs but does not update ZHOME.

Related Settings: For setting <1>, see template section @RAP (page 5-18) and template word #[XYZ]POS (page 6-35). For settings <2> and <3>, see template section @TOOLCHG (page 5-19) and template words #[XYZ]POS (page 6-35) and #[XYZ]HOME (page 6-34).

290. 3-D, Tool plane vector #[IJK]NO referenced along

Choices:

- <0> Z axis (default setting)
- <1> X axis
- <2> Y axis

291. 3-D, Specify index angles for new Tool Plane orientation

Choices:

- <0> As absolute or incremental based on #ABSI setting
- <1> Always absolute from the world XY Plane origin
- <2> Always incremental from the previous Tool Plane origin

Explanation: This setting controls whether the angle output for #INDX[ABC] and #ST[ABC] is an absolute or incremental value or is switchable, on the basis of the current positioning mode. This is used for both positioning (for example, @RAP and @STPROF) and feed motion (for example, @LINE and @ARC) template sections. The following is an example of how to use it in the @RAP and @LINE sections:

```
@RAP
X#XPOS Z#ZPOS C#INDXC
@LINE
X#XPOS Z#ZPOS C#INDXC
```

Related Settings: See template section @TPINDEX (page 5-19) and template words #INDX[ABC] (page 6-13) and #ST[ABC] (page 6-24).

292. 3-D, Primary rotational axis

Choices:

- <0> No rotation axis available, 3 Axis machine
- <1> A axes rotation around an axis parallel to X
- <2> B axes rotation around an axis parallel to Y
- <3> C axes rotation around an axis parallel to Z

Explanation: The world origin of the part must correspond to the primary table origin.

294. 3-D, Secondary rotational axis for 5 axis machines

Choices:

- <0> No secondary rotation axis available, 3 or 4 Axis machine
- <1> A axes rotation around an axis parallel to X
- <2> B axes rotation around an axis parallel to Y

Explanation: For a 5-axis table-indexing configuration, this axis is the one that stays parallel to the machine axis after the index takes place.

296. 3-D, (5-axis only) Distance from primary origin to secondary axis

Explanation: This question is for machines that use two rotary tables. It refers to the positive or negative distance from the primary origin to the secondary axis. This distance is as measured along a line that runs parallel to the primary axis. For example, -4.500 means the secondary axis is 4.5 below the origin of the primary axis as measured from the primary axis.

Related Settings: See template section @TPINDEX (page 5-19) and template word #[XYZ]SET (page 6-36).

297. 3-D, (5-axis only) Distance from primary axis to secondary axis

Explanation: This question is for machines that use two rotary tables and is measured along a line that runs parallel to the third axis. For example, on a machine with A-primary and B-primary tables, the measurement would be parallel to the Z axis. In this case, -2.00 means the B axis crosses 2.00 units below the A axis in -Z.

Related Settings: See template section @TPINDEX (page 5-19) and template word #[XYZ]SET (page 6-36).

298. 3-D, Table rotation angles determined by

Choices:

<0> Find closest tool plane X to world X alignment (V2.0 default)
<1> Shortest total angle rotation

Explanation: After the Z normal is aligned, SmartCAM selects the rotation angle that most closely aligns the X tool-plane axis with the X world tool-plane axis. The resulting X tool-plane orientation will be within 90° of the X world tool-plane axis

299. 3-D, Code output for rotation of tool around primary axis

Choices:

<+1> Default
<-1> Reversed

Explanation: This question controls whether SmartCAM's default index direction indicators are used or are reversed. A setting of <+1> yields the default sign indicators for #INDX[ABC] and codes for #[ABC]DIR (entered in questions 451 and 452). A setting of <-1> reverses them.

Related Settings: See questions 304, 451, and 452, template section @TPINDEX (page 5-19), and template words #INDX[ABC] (page 6-13) and #[ABC]DIR (page 6-2).

300. 3-D, Code output for rotation of tool around secondary axis*Choices:*

- <+1> Default
- <-1> Reversed

Related Settings: See questions 304, 451, and 452, template section @TPINDX (page 5-19), and template words #INDX[ABC] (page 6-13) and #[ABC]DIR (page 6-2).

301. 3-D, Primary table angular component output*Choices:*

- <+1> Output SmartCAM default angle
- <-1> Output complementary angle (360 - angle)

Explanation: For the default of 90°, a setting of <+1> yields 90, and <-1> yields 270.

Related Settings: This does not apply if question 304 is set to <2>.

302. 3-D, Secondary table angular component output*Choices:*

- <+1> Output SmartCAM default angle
- <-1> Output complementary angle (360 - angle)

Explanation: For the default of 90°, a setting of <+1> yields 90, and <-1> yields 270.

Related Settings: This does not apply if question 304 is set to <2>.

303. 3-D, output from local tool plane or world coordinate system*Choices:*

- <0> Output code from local tool plane
- <1> Output code from world XY plane coordinate system

Explanation: Use setting <0> to output local coordinate values on the basis of the tool plane's origin point and related template words. This includes when the tool plane's orientation changes when the table is rotated into that tool plane's machining position.

Use setting <1> to output world coordinate values for all tool-plane model data. This includes changes in location by virtue of table rotation to move the tool plane into machining position.

Related Settings: See questions 296 and 297.

304. 3-D, Rotary axis, allowable range of command angles*Choices:*

- <0> 0 to +359.999 (CW/CCW indicated by a code)
- <1> 0 to -359.999 for CW, 0 to +359.999 for CCW
- <2> Greater than +/- 359.999 (like linear axis)

Explanation: This setting indicates whether the controller resets the C-axis position register to 0 each time 360 is passed or allows specification of angles greater than ± 360 when the machine must move more than one full revolution. The value is output through the template word **#INDXC**. The following results occur from the settings:

<0> Direction of rotation of the C axis is controlled by the output of codes you specify with questions 451 and 452.

<1> Direction of rotation is controlled by the sign of the angle. Angular values are reset back to 0 when 359.999 is exceeded in either direction.

<2> The machine accepts angle values greater than or equal to 360 when the rotary axis must make more than one full revolution.

Subprogramming

☞ Questions 330 to 338 pertain to SmartCAM advanced products only.

330. 3-D, subroutine or subprogram support level

Choices:

<0> No sub support

<1> Sub support with no repeats

<2> Sub calls may include an optional repeat count

Machine Types: Mill, router, laser, wire EDM, 3-D only for subroutine support

Explanation: This item specifies whether your controller supports subprogramming and, if so, whether multiple loops are allowed. The following results occur from the settings:

<0> Pattern elements will be exploded during coding.

<1> If pattern repeats are used, pattern elements will be exploded during coding.

<2> Most newer controllers support subprogramming, and many allow multiple loops. An example for a Fanuc 6M would be

```
M98 P2100 L4
```

where M98 calls the subprogram, P2100 is the subprogram identifier, and L4 causes the subprogram to loop four times using **#SREPT**.

331. 3-D, output file for subprogram or subroutine code

Choices:

<0> Separate files by pattern name

<1> Add subs to the end of the main program

<2> Put subs at the beginning of the main program

<3> Program subs in the main program along with their first use

Machine Types: Mill, router, laser, 2-axis wire EDM

Explanation: Setting <3> must be handled by the template file through use of the #INCLUDE() function. All choices will create separate files with a .sub extension.

332. Positioning mode to use for subcode

Choices:

<0> Absolute

<1> Incremental

Machine Types: Mill, router, laser, 2-axis wire EDM

Explanation: This item specifies whether to output code for subprograms as absolute or incremental positioning moves. A setting of <0> requires the use of a zero preset or work-coordinate system. A setting of <1> does not.

Related Settings: Question 330 must be set to <1> or <2>.

334. 3-D, May subs be rotated?

Choices:

<0> No

<1> Yes

Machine Types: Mill, router, laser, 2-axis wire EDM

Explanation: Use setting <1> to display pattern elements rotated. The angle of rotation is passed by template words #ROT1 and #ROT2.

337. 3-D, Start first sub. program block numbers with

Machine Types: Mill, router, laser, 2-axis wire EDM

Explanation: Entering a value of 2000 will start the first sub with a block number of N2000.

338. Drill sub cycle type

Choices:

<0> Omit first hole from drill sub cycle definition

<1> Include first hole in drill sub cycle definition

Explanation: Use setting <0> if the drill cycle call initiates machining. When you use setting <1> and use subroutine repeats, SmartCAM codes all drill elements in the repeated subroutines as separate holes without a subroutine call, regardless of question 330's setting.

User Commands

360. User command word switched by template word #C0=0

Explanation: Enter the command code setting that SmartCAM outputs by the #C word. This works similar to the system switches. These are useful for outputting items such as coolant commands and spindle ranges. For example, given the following lines in a template file and with question 360 set to M15:

```
#EVAL( #C0=0 )
#C0
```

the string set above (M15) is output at the #C0 location.

These template words are interrelated in that they act like a sequential switch. You can control a higher #C word with a smaller #C word. This is best understood by studying the following chart and equivalencies:

#C Value	Sequential Value of #C0
#C0=0	0
#C0=1	1
#C1=0	2
#C1=1	3
#C2=0	4
#C2=1	5

#EVAL(#C0=2) is equivalent to #EVAL(#C1=0).

#EVAL(#C0=3) is equivalent to #EVAL(#C1=1).

#EVAL(#C0=4) is equivalent to #EVAL(#C2=0).

#EVAL(#C0=5) is equivalent to #EVAL(#C2=1).

361. User command word switched by template word #C0 = 1

Explanation: Enter the command code you want SmartCAM to output when #C0=1. For example, you might use M16 to turn on special function #0.

362. User command word switched by template word #C1 = 0

Explanation: Enter the command code you want SmartCAM to output when #C1=0. For example, you might use M17 to turn off special function #0.

363. User command word switched by template word #C1 = 1

Explanation: Enter the command code you want SmartCAM to output when #C1=1. For example, you might use M18 to turn on special function #1.

364. **User command word switched by template word #C2 = 0**

Explanation: Enter the command code you want SmartCAM to output when #C2=0. For example, you might use M20 to turn off special function #1.

365. **User command word switched by template word #C2 = 1**

Explanation: Enter the command code you want SmartCAM to output when #C2=1. For example, you might use M21 to turn on special function #2.

366. **User command word switched by template word #C3 = 0**

Explanation: Enter the command code you want SmartCAM to output when #C3=0. For example, you might use M25 to turn off special function #3.

367. **User command word switched by template word #C3 = 1**

Explanation: Enter the command code you want SmartCAM to output when #C3=1. For example, you might use M26 to turn on special function #3.

368. **User command word switched by template word #C4 = 0**

Explanation: Enter the command code you want SmartCAM to output when #C4=0. For example, you might use M35 to turn off special function #4.

369. **User command word switched by template word #C4 = 1**

Explanation: Enter the command code you want SmartCAM to output when #C4=1. For example, you might use M36 to turn on special function #4.

370. **User command word switched by template word #C5 = 0**

Explanation: Enter the command code you want SmartCAM to output when #C5=0. For example, you might use M45 to turn off special function #5.

371. **User command word switched by template word #C5 = 1**

Explanation: Enter the command code you want SmartCAM to output when #C5=1. For example, you might use M46 to turn on special function #5.

Tape-to-Shape

390. **TTS, control out**

Explanation: Enter the character that indicates the beginning of a comment in the code.

391. **TTS, control in**

Explanation: Enter the character that ends a comment. If you do not use one, the rest of the block following a control out character will be ignored.

392. **TTS, leave space chars in block to be interpreted**

400. Is rapid positioning command NON modal?

Choices:

- <0> No
- <1> Yes

Explanation: The machine function or coordinate value is active only in the block command it appears in. In cases where rapid positioning commands are not modal and need to be repeated for this type of move, set this question to <1>.

Related Settings: See questions 401–407.

401. Is linear move command NON modal?

Choices:

- <0> No
- <1> Yes

Explanation: Indicate whether your machine's linear move commands are nonmodal.

402. Are circular move commands NON modal?

Choices:

- <0> No
- <1> Yes

Explanation: Indicate whether your machine's circular move commands are nonmodal.

403. Are arc center command words NON modal?

Choices:

- <0> No
- <1> Yes

Explanation: Indicate whether your machine's arc center commands are nonmodal.

404. Are fixed cycle hole operation commands NON modal?

Choices:

- <0> No
- <1> Yes

Explanation: Indicate whether your machine's fixed-cycle hole-operation commands are nonmodal.

405. Are cutter compensation commands NON modal?

Choices:

- <0> No
- <1> Yes

Explanation: Indicate whether your machine's cutter compensator commands are nonmodal.

406. Is absolute positioning command NON modal?*Choices:*

- <0> No
- <1> Yes

Explanation: Indicate whether your machine's absolute positioning commands are nonmodal.

407. Is incremental positioning command NON modal?*Choices:*

- <0> No
- <1> Yes

Explanation: Indicate whether your machine's incremental positioning commands are nonmodal.

410. TTS, maximum cutting feed rate

Explanation: Enter the maximum feed rate for linear cutting moves. Some machines use the same move command for linear cutting moves as for rapid-positioning moves. Any move command with a feed rate greater than this value is interpreted as a rapid-positioning move.

411. TTS, use contour cutting command for interpretation*Choices:*

- <0> Not required
- <1> Use contour On command to distinguish a linear cutting move from a rapid move
- <2> Use contour On command to identify torch tool as active
- <3> Use contour On command for both <1> and <2> above

Explanation: Indicate how your machine contour-cutting commands identify when to use linear cutting moves instead of rapid-positioning moves. It also indicates whether the use of a contour on command identifies that a torch attachment is active instead of the current punch tool. Setting <2> is an alternative to using the location offset to identify a torch or attachment using the @TORCH () .tts section.

Related Settings: See questions 110–113 and 230.

412. TTS, sequence to activate contour cutting command

Machine Types: Mill, router, laser, 2-axis wire EDM, lathe, punch, contour machine

Explanation:

- <0> BOS (Beginning Of Span), before move
- <1> EOS (End Of Span), after move

Rotary Contouring

☞ Questions 430 to 453 pertain to SmartCAM advanced products only.

430. Type of primary rotary axis code supported for Wrapped geometry

Choices:

<0> Positioning only

<1> Contouring with G01 output only

<2> Contouring with G02/G03 support

Explanation: Use setting <0> for machines that allow positioning of only the C axis. SmartCAM generates an error report if profiling is done and generates code as if question 430 were set to <1>.

Use setting <1> for machines that allow simultaneous XZC linear interpolation. SmartCAM codes the lines using the @LINE section, with the cutting move command specified in question 85. Arcs are broken into small line segments based on the setting of question 91. SmartCAM outputs all wrapped-profile feed moves as G01 (it interpolates arcs into G01 moves).

Use setting <2> for machines that allow simultaneous XZC circular interpolation. Arcs are coded using the @ARC section and the cutting move commands set in questions 92 and 93. Lines are coded as with setting <1>. SmartCAM outputs G02/G03 for arcs.

☞ The only type of arc center output supported is radius (see question 100 and template word #ARAD (page 6-2).

431. Type of secondary rotary axis code supported

Choices:

<0> Positioning only

<1> Contouring with G01 output only

<2> Contouring with G02/G03 support

432. Smallest primary rotary angle increment

Explanation: Enter the minimum command increment your machine allows for the C axis. The output format is set in question 22, with the value being assigned to #INDXC.

Related Settings: This applies to question 430.

433. Smallest secondary rotary angle increment

Explanation: This is the smallest angle allowed by the machine controller.

Related Settings: This applies to question 431.

435. Rotary Axes, Feedrate type supported*Choices:*

<0> Job operation feedrate (or linear equivalent conversion when #436 > 0)

<1> Inverse Time (1/time for move) in minutes

<2> Degrees Per Minute (Mill/Turn only)

<3> Linear equivalent conversion using #436

Explanation:

<0> For all linear and rotary feed rates, the output is in distance per minute.

Machine/controller types: GE1050, GE-FANUC, CINCI

When question 436 is set to 0, the following occurs:

#FEED is unchanged (job operations file feed rate in distance per minute (DPM) or distance per revolution (DPR), including any adjustments for depth feed or arcs).

#FDMODE is unchanged.

When question 436 is set greater than 0 and there are only linear moves (no rotary moves), the following occurs:

#FEED = DPM feed (DPR feed converted to DPM).

#FDMODE = DPM mode (0).

When question 436 is set greater than 0 and there are rotary or linear/rotary moves, the following occurs:

#FEED = linear equivalent conversion (see page 3-15).

#FDMODE = DPM mode (0).

<1> For rotary and combined linear/rotary feed rates, the output is the reciprocal of the number of seconds required to complete the programmed move.

Machine/controller types: FADAL, G&L, MAHO

When there are only linear moves (no rotary moves), the following occurs:

#FEED = DPM feed (DPR feed converted to DPM).

#FDMODE = DPM mode (0).

When there are rotary or linear/rotary moves, the following occurs:

#FEED = inverse time conversion (subject to maximum and minimum in questions 440 and 442).

#FDMODE = inverse time mode (2)

<2> For rotary and combined linear/rotary feed rates, the output is degrees per minute; linear feed rates are output as distance per minute.

When there are only linear moves (no rotary moves), the following occurs:

#FEED = DPM feed (DPR feed converted to DPM).

#FDMODE = DPM mode (0).

When there are rotary or linear/rotary combination moves, the following occurs:

#FEED = DPM conversion (see page 3-16).

#FDMODE = DPM mode (0).

<3> Linear equivalent conversion using question 436.

Machine/controller types: OKUMA, Pratt & Whitney

When there are only linear moves (no rotary moves), the following occurs:

#FEED = DPM feed (DPR feed converted to DPM).

#FDMODE = DPM mode (0).

When there are only rotary moves (no linear moves), the following occurs:

#FEED = DPM conversion (see page 3-16).

#FDMODE = DPM mode (0).

When question 436 is set greater than 0 and there are linear and rotary moves, the following occurs:

#FEED = linear equivalent conversion (see page 3-15).

#FDMODE = DPM mode (0).

When question 436 is set to 0 and there are linear and rotary moves, the following occurs:

#FEED = DPM feed (DPR feed converted to DPM).

#FDMODE = DPM mode (0).

☞ If question 435 is set to <0> and question 436 is set to 0, or if question 435 is set to <2> or <3>, the controller automatically anticipates the type of feed word (#FEED) to expect, based on the information in the NC code block. It does not use a feed mode (#FDMODE) setting to interpret the feed word.

436. Rotary axis linear equivalent distance

Explanation: When question 435 is set to <0> or <3>, a linear equivalent of 360° rotation is required for feed rate calculations. For example, if you set this question to 20, then 360° is equivalent to 20 inches of linear motion.

438. Inverse time feedrate code

Explanation: This setting specifies the code (for example, G93) to output to indicate inverse-time feed rate mode (INV). This code is output by the template word #FDMODE when question 435 is set to <1>.

439. Inverse-time feed rate format

Explanation: This setting specifies the format for inverse-time feed rate. (See page 3-2 for information about numeric formats.)

440. Inverse-time feed rate minimum allowed

Explanation: This setting specifies the minimum inverse-time feed rate or velocity accepted by the controller. Feed rates less than this amount will be output as the minimum value.

441. Inverse-time feed rate maximum allowed

Explanation: This setting specifies the maximum inverse-time feed rate or velocity accepted by the controller. Feed rates greater than this amount will be output as the maximum value.

442. Cutter compensation on wrapped elements

Choices:

<0> No

<1> Yes, use other cutter compensation settings

Explanation: This setting indicates whether cutter-radius compensation is available for contouring geometry that is wrapped around the part.

443. Type of Rotary axis code supported for Flat geometry

Choices:

<0> Helical interpolation

<1> Contouring with linear interpolation (polar coordinates)

<2> Contouring with circular interpolation (polar coordinates)

<3> Contouring with linear interpolation (Cartesian coordinates)

<4> Contouring with circular interpolation (Cartesian coordinates)

Explanation: Use setting <0> for machines that allow linear axes to be coded with the rotary axis but cannot interpolate a straight line or an arc. Question 445 is used to break up lines and arcs into short helical segments used to approximate a contour.

Use setting <1> for machines that, given an XZC target coordinate, can generate a straight line vector to the end point. Arcs are broken into line segments using question 91 for deviation tolerance. The cutting-move code used is specified in question 442.

Use setting <2> for machines that can generate an arc given the XC coordinates for the end point and the radius or arc center vectors. Questions 443 and 444 specify the cutting-move code for these arcs.

Setting <3> works like setting <1>, but output is in Cartesian coordinates.

Setting <4> works like setting <2>, but output is in Cartesian coordinates.

☞ These settings are unavailable if question 430 is set to <0>.

444. Rotary axis special cases

Choices:

<0> No special cases

<1> Code lines that do not require rotary axis motion with the @LINE section

<2> Code lines that do not require rotary axis motion and arcs that require only rotary axis motion with the @LINE section

Explanation: With setting <0>, SmartCAM handles all rotary motion by standard linear or circular interpolation (or both) for wrapped and flat geometry.

Use setting <1> for machines that cannot perform feed motion with the rotary axis but can feed in one or more linear axes to cut straight lines after positioning the rotary axis, or for machines that interpolate lines by feeding the linear and rotary axes but require a special linear cutting code for lines that do not require rotary motion. This setting calls the @LINE section in these cases and outputs the #MOV command specified in question 445.

Use setting <2> for machines that are capable only of helical interpolation with the rotary axis. Arcs that require only rotary-axis motion and lines that do not require rotary-axis motion will be coded using the @LINE section, outputting the #MOV command specified in question 85.

445. Special rotary axis Linear cutting move code

Explanation: Enter the code (for example, G101) that your machine tool requires for rotary-axis linear cutting moves. Use this setting when question 443 is set to <1> or <2>.

446. Special rotary axis CW circular cutting move code

Explanation: Enter the code (for example, G102) that your machine tool requires for rotary-axis clockwise linear cutting moves. Use this setting when question 443 is set to <2>.

447. Special rotary axis CCW circular cutting move code

Explanation: Enter the code (for example, G103) that your machine tool requires for rotary-axis counterclockwise linear cutting moves. Use this setting when question 443 is set to <2>.

448. C-axis - Segment length for interpolating lines and arcs into helical XZC motion.

Explanation: Indicate the length of segments that lines and arcs that require XC or ZC motion should be broken into. For example, if you use 0.0100, lines and arcs that require XC or ZC motion will be broken into segments 0.0100 long. Use this setting when question 443 is set to <0> or <1>.

449. Lathe C-axis disengage code (mill-turn)

Explanation: Enter the code (for example, M109) that your machine tool requires to disengage the C-axis drive. This code is output by the template word #CENG (see question 450).

450. Lathe C-axis engage code (mill-turn)

Explanation: Enter the code (for example, M110) that your machine tool requires to engage the C-axis drive. This code is output by the #CENG template word in the @START, @TOOLCHG, and @END sections. SmartCAM outputs the code when the tool changes from a turning tool to a milling tool and vice versa.

451. Rotary axis CW positioning code

Explanation: Enter the code (for example, M15) that your machine uses to control the direction of the C-axis rotation when positioning. This code is output with the template word #CDIR.

Related Settings: See question 304.

452. Rotary axis CCW positioning code

Explanation: Enter the code (for example, M16) that your machine uses to control the direction of the C-axis rotation when positioning. This code is output with the template word #CDIR.

Related Settings: See question 304.

453. Rotary axes, what is the rapid traverse feed rate in degrees/minute

Explanation: Enter the number of degrees per minute that your machine's rotary axis can move. SmartCAM uses this value for cycle-time calculations only.

470. Machine Event toolbox PCB file

Explanation: Specify the name of the .pcb file to be used for the Mach Events toolbox.

471. Code generation Mode

Choices:

<0> JSF mode

<1> JOF mode

Explanation: Set this question based on the mode in which the code should operate. JSF mode will *not* take advantage of the Extended Data. JOF mode will take advantage of the job data, such as fixed cycles and feed rates.

Template File Overview

Introduction

Template files determine the format SmartCAM uses to generate code from CNC process models. Every template file has a `.tmp` extension.

SmartCAM combines information from the model's database (`.pm4`), job operations (`.job`), and machine (`.smf`) files. It outputs that information through the template file. You may also put math and logic statements into a template file to control the final NC code.

SmartCAM organizes template files into functional sections that begin with the @ (at) symbol. These sections contain the following basic types of information SmartCAM uses to output NC code:

- Literal code to be output just as entered.
- Template words for output of program information, and control of that output.
- Conditional brackets (<>) to make some output optional.

In addition, these sections may also contain math and logic functions.

How to Read and Edit the Template File

When you work with a code generator, you may need to modify the template file so that the NC code matches your machine's format and your programming style. Template files have a `.tmp` extension and are stored in subdirectories of your `\SmartCAM Path` directory. The name of each subdirectory begins with one or two letters identifying the application and ends with `smf`. (For example, the `mismf` subdirectory is for Milling `.tmp` files, and the `atismf` subdirectory is for Advanced Turning `.tmp` files).

Template files are ASCII text files and can be edited with Edit Plus or any other text editor. Before editing a template file, make a backup copy of it. After editing a template file, be sure to save your changes. Then enter your SmartCAM application, code the model again, and check the results to see if your changes were successful.

Template File Sections

Template files are organized by section. For example, the **@LINE** (at line) section processes lines or linear move commands.

There are several sections in a `.tmp` file. The purpose of some, such as **@LINE** and **@ARC**, is easy to identify. Others, such as **@STPROF** or **@FXD1**, may be less obvious. A complete listing and description of the basic template sections SmartCAM uses begins on page 5-12.

As SmartCAM generates code, it moves sequentially through the database, working with one element at a time. The operation that an element describes triggers calls to one or more of the `.tmp` file's sections. Then, working with the machine file, it completes the parameters of the section to produce code for that operation.

SmartCAM selects template sections by looking for user commands, element types, tool types, location in the program, and special logic tests. You need to consider only those sections that occur in your model's database when reviewing the `.tmp` file.

You can place a variety of information into a template section, such as math formulas and conditional tests. You can also add setup information or remarks that assist you when you are working with the file.

There are two types of template sections:

- System-defined template sections are those that SmartCAM expects to see. It routinely calls these sections when generating NC code.
- User-defined template sections are those that you create. You must build in ways for SmartCAM to read these sections, since it doesn't look for them automatically.

System-Defined Template Sections

SmartCAM automatically finds and calls system-defined sections according to a predetermined sequence. Most of the logic SmartCAM uses for calling these sections is obvious.

For example, the first section SmartCAM processes is **@START**. The last section SmartCAM processes is **@END**. When SmartCAM generates code for a line in a profile, it processes the **@LINE** section. The logic behind SmartCAM calling some sections is not quite so obvious. For example, SmartCAM uses **@RAP** to process both points (used for positioning) and holes (used for punch hits) in punch applications.

The logic that SmartCAM uses varies with the nature of the database. For example, to go from a tool change (@TOOLCHG) to the start of a profiling move, SmartCAM may process the sections in the following sequence:

@TOOLCHG	(output code to change to new tool)
@STPROF	(rapid move to point above start of profile)
@ZCHKMV	(move to Z-axis check position)
@ZDPTHMV	(feed down to the Z depth of the profiling move)
@LINE	(process a linear cutting move)

If any of these or other system-defined sections are missing in your template file, SmartCAM substitutes a different section or skips the section completely. For example, if @STPROF doesn't exist, it substitutes @RAP. If @LINE doesn't exist, no code is output for lines in the database.

☞ SmartCAM does not alert you with a message when a section is missing. However, SmartCAM will report that the section is missing if you have it generate an error report (see page 7-1).

It is not necessary to understand the exact logic SmartCAM uses to call different template sections. It is more important to know how to set up a test to determine how your code generator is performing operations you are having problems with. In most cases it is better to build a simple model file and test what the code generator is doing than to worry about the internal logic. (See chapter 7 for Information about testing procedures.)

You can control the built-in selection of system-defined sections indirectly with the template words #CALL and #EXIT. #CALL calls a template section and #EXIT exits a section. Insert these words into logic statements in appropriate template sections to alter SmartCAM's normal template processing sequence.

User-Defined Template @ Sections

User-defined template sections have many uses, most of which you never notice when SmartCAM generates code. These sections perform specific functions appropriate to your application.

SmartCAM doesn't automatically call user-defined template sections. You must indicate to SmartCAM when to call a user-defined section. The following are two ways to do that:

- Use the #CALL template word from within a template section. This enables you to call a user-defined template section from another section and base that call on special conditional tests.
- Assign a user command when creating the CNC process model. When SmartCAM encounters a user command in the model's database, it calls a corresponding section and processes its contents. For example, you could insert the following user command into a model to reposition the material:

```
@REPO ( #XSET=50 )
```

This command tells SmartCAM to go to the **@REPO** section, assign a value of 50 to the template word **#XSET**, and process the section.

- ☞ Try to minimize the number of user commands, because they have to appear in the correct sequence with the correct syntax in order to work properly.

Simple Template Sections

The contents of template sections vary, depending on the operations, associated values, and any additional statements. The basic contents of system-defined template sections are fairly easy to understand. They consist of template words that transfer information.

A simple example should help you fully understand how a template section functions. Look at the following example and explanation:

```
@LINE
< #MOV>< X#XPOS>< Y#YPOS>< Z#ZPOS>
```

- **@LINE** signals the start of the section. SmartCAM accesses this section when code for a line needs to be output. The template words in the section act as carriers, transferring values from the model and machine file to the NC code (see chapter 1).
- The template word **#MOV** contains the current move code. Since SmartCAM is generating code for a line, a **G01** is output. The actual code is a setting you specify in question 85 of the machine file.
- The **#XPOS**, **#YPOS**, and **#ZPOS** words contain the model's X, Y, and Z coordinates for the end of the line. The letters X, Y, and Z (and the space in front of each of these characters) are *literals*. SmartCAM outputs them exactly as shown.
- The brackets (<>) on each side of the template words indicate that the output for these template words is conditional. Read < X#XPOS> as Output a space, followed by the letter X, followed by the current X position, only if the X position is different from the last X output by the .tmp file. (For more information about conditionals, see page 5-7.)

SmartCAM considers the end of the line in the template section to be the end of a block of code (except when you use **#EXLN**). A new block number is output if you turn on block numbering in the machine (.smf) file.

Types of Template Words

Template words represent the blanks to be filled in when SmartCAM generates code using the model's part geometry, tool definitions, or machine parameters. A template word must be preceded by a # symbol and must be spelled correctly in uppercase letters. Most template words produce output, but some special template words perform a function or control the output in other ways. The following list identifies template word types:

- *Switches* select entries from the machine `.smf` file.
- *Numerics* hold numeric values.
- *Strings* hold string values.
- *Controls* control the current status, condition, or data to be used.

Numerics, strings, and switches can be either system-assigned or user-assigned. Remember that each word contains a specific piece of information that affects final code output.

Switches

System-Assigned

These template words act like a switch. The code SmartCAM outputs depends on the setting of the template word. For example, four different settings for the `#MOV` template word result in the code output shown in the following list:

Setting	Description	Code Output
<code>#MOV=0</code>	rapid positioning code	G00 from question 80
<code>#MOV=1</code>	linear cutting move	G01 from question 85
<code>#MOV=2</code>	clockwise arc command	G02 from question 92
<code>#MOV=3</code>	counterclockwise arc command	G03 from question 93

The initial value of `#MOV` is an integer value (0, 1, 2, or 3) that is set in the model's database. The code `#MOV` outputs is a text string (G00, G01, G02 or G03) defined in the `.smf` file. SmartCAM reads these string values only when it determines `#MOV`'s setting and is ready to output code.

You can implement a conditional test for the initial values of any switch template word, changing it as necessary. Be sure to work with the numeric value and not the string value that is output. For more information about conditionals, see chapter 7.

User Assigned

You can create user switches by defining them in the `.smf` file. They are called `#C` template words, and are set by `.smf` questions 360–371. You can design these words to control different applications, such as coolant on/off commands. A description of these switches and examples of their use is included in chapter 6.

Numeric Template Words

System Assigned

SmartCAM assigns numeric values to system numeric template words. An example of this is **#XPOS**, which holds the X value for the end of a line, arc, or point of a model. Another example is **#FEED**, which holds the job operations setup feed rate for an operation.

Even though the initial values of these template words are set by SmartCAM, you can alter them with math and logic statements in the `.tmp` file. For example, you can vary the feed rate of an operation when certain conditions are met.

User Assigned

You can assign values to special full-numeric (double-precision) and integer-numeric words. There are 20 of each type available.

Use the full-numeric template words (**#V0–#V19**) to store and make calculations involving axis positioning information. Since there is a limited number of these numeric words, use them only when you need to work with positioning information. (Lathe module reserves **#V0–#V4** for threading.)

Integer template words (**#U0–#U19**) can output only integer values, but they can store decimal values. Use integer words for the following purposes:

- To act like an accumulator (counter) to count how many times a function has been performed.
- To act like a two-way (on/off) switch.

String Template Words

System Assigned

SmartCAM assigns alphanumeric character strings to string template words. There are a variety of string words that SmartCAM sets. For example, **#FILE** is a string template word that transfers the file name of your coded file.

You can test string template words with a logic statement (if they equal a numeric value) or by using the **#IFSTR()** command.

User Assigned

There are 20 user-assigned template words available (**#S0–#S19**). These words can be any valid series of up to 78 alphanumeric characters. You can assign a string word to a numeric word if the string word represents a valid number. You can always assign a numeric word to a string word.

Control Words

A special function occurs when SmartCAM encounters a control word in the `.tmp` file. For example, SmartCAM starts generating block numbers when it encounters the `#ONBLK` control word. Other examples include `#EXC`, which excludes any following tests, and `#EXIT`, which causes the system to exit the current template section.


Using Literals in an @ Section

Literal code displays in the final code exactly the way it is in the template file. Basically, SmartCAM considers anything that is not a template word or characters for a conditional test as a literal. Whether or not the literal actually occurs in the final code depends on its location in the @ section.

For example, the tape rewind character, `%`, might need to be output at the start and end of a program.

You can accomplish this by inserting the `%` in the `@START` and `@END` sections.

If you wish to use one of the special characters SmartCAM uses to label sections (`@`), template words (`#`), or conditionals (`<>`) as a literal, you must precede it with a `\` (backslash). Consequently, if you want to output a `\` as a literal, it must be preceded by another `\`. See chapter 7 for more information about the use of literals.

 The space character is a literal that affects the format of your final code. Most of the SmartCAM sample template files have a space between command codes. This improves readability and helps you distinguish between literal code and template words in each section.

Conditional Brackets (<>)

You can use brackets (`<>`) to set up conditional tests within a template file. Conditional tests result in the content of a template word being output if that content differs from the last time SmartCAM used it. For example, you place the template words that output coordinate information in conditional brackets (`<X#XPOS>`) if your controller doesn't require that coordinate values be output unless they have changed. You can also nest conditionals so the template word outside of the brackets controls the final output to the file.

If you want spaces between your words in lines with conditional statements, be sure the spaces are inside the conditional brackets. If spaces are outside the brackets and none of the conditions is met, a line of just spaces is output.

You can include multiple template words in a single conditional. If a change occurs to any of the words, the content of all the words is output. You can add one of the following control words to a conditional with multiple template words to change the evaluation and output:

`#OR` is the default logic operation for multiple words. If any change occurs the condition tests true.

`#EXC` excludes any following template words from the testing.

#AND requires that *all* of the words which follow must change for the condition to test true.

Examples:

```
< #MOV#OR X#XPOS>
< #DCOMP#EXC D#DOFF>
< X#XPOS#AND Y#YPOS>
```

You can include an **#ELSE** state to output the content of a different template word if no change has occurred in the tested word or words. **#ELSE** must follow the initial conditional test because no conditional test is made for any words inside the **#ELSE**'s brackets.

Example:

```
<T#NTOOL>#ELSE< T0>
```

☞ Use **#IFCHG** to test if a template word is output. See page 6-12.

Logic Control

SmartCAM supports logic operations that enable you to test for certain conditions and trigger different actions based on the results of the test. These logic operations are not limited by the change in the content of a template word, as were the conditionals.

One logic statement is **#IF(test)< >** where (test) is the test operator and <> is the template word or words whose content is set or output. You can insert **#ELSE** after the **#IF()** conditional brackets for an alternate action.

Another logic statement is **#IFSTR(test)< >** where (test) is a string comparison and <> identifies the output.

Example:

```
#IFSTR( #S1=BAR )<G00 X#HOME Y#HOME>
```

You can also add an **#ELSE** statement to the test.

Example:

```
#ELSE<G00 X#POS Y#POS>.
```

The condition test operators include:

=	equal to
< >	not equal to
>	greater than
>=	greater than or equal to
<	less than
<=	less than or equal to

Be sure to use these operators the way they appear above. SmartCAM makes the test by comparing the value on the right with the one on the left, based on the logic operator that separates them.

- ☞ You can nest **#IF** and **#ELSE** conditions. Also, multiple conditional tests may be made. The **#OR** and **#AND** operators work with **#IF**.

Example:

```
#IF( #V2>3 , #OR#V2<-1 )
```

(Note the comma between tests.)

Use **#EXIT** to leave a template @ section early and not use the remaining functions.

Example:

```
#IF( #V9=1 ) <#EXIT>
```

See page 6-45 for more information.

Adding Comments

Adding a // (double forward slash) to the line of a template section causes SmartCAM to ignore any information that follows on that line. This enables you to add comments that document changes or tests you're making. You can add the // to any part of the line. At the start of the next line, SmartCAM begins processing information again.

Mathematical and Trigonometric Functions

You can include mathematical and trigonometric calculations in a template file @ section. These functions enable SmartCAM to produce error-free code in the format you want and greatly expand SmartCAM's capabilities.

Basic Statement

The template word **#EVAL()** precedes most math calculations and assignments in the .tmp file. For example, **#EVAL(#V1=2)** sets the value of **#V1** to 2. It is important to note that **#EVAL()** does not cause any information to be output in the code. It performs a calculation and assigns the results to a template word. A separate statement is necessary to output results of the calculation.

Other examples of how to use **#EVAL()** include the following:

Function	Description
<code>#EVAL(#V2=0)</code>	Sets the template word #V2=0
<code>#EVAL(#V3=1)</code>	Sets the template word #V3=1
<code>THE VALUE OF V3=#V3</code>	Outputs the following: THE VALUE OF V3=1
<code>#EVAL(#V4=2+3)</code>	Sets the template word #V4=5
<code>#EVAL(#PUNCH=1)</code>	Sets the template word #PUNCH=1 . This sets the system to punch active mode. An M75 or other string set in .smf question 223 will be output when #PUNCH is encountered in the .tmp file.
<code>#EVAL(#PUNCH=0)</code>	Sets the template word #PUNCH=0 . This sets the system to punch inhibit mode. An M85 or other string set in .smf question 222 will be output when #PUNCH is encountered in the .tmp file.

Math Functions

SmartCAM supports a variety of math operations. Notice that the words **SQR**, **ABS**, **SGN**, **INT** in the following list are not preceded by the **#** character. They are operators, not template words.

Operator	Meaning	Example
+	add	<code>#EVAL(#V5=#ZPOS+#ZSET)</code> sets #V5 to #ZPOS plus #ZSET
-	subtract	<code>#EVAL(#V5=#ZPOS-#ZSET)</code> sets #V5 to #ZPOS minus #ZSET
*	multiply	<code>#EVAL(#V5=#ARAD*2)</code> sets #V5 to two times the arc radius
/	divide	<code>#EVAL(#V5=#V5/2)</code> sets #V5 to #V5 divided by 2
^	exponential	<code>#EVAL(#V5=#V5^3)</code> raises #V5 to the third power
SQR	square root	<code>#EVAL(#V5=SQR(#LNLEN))</code> sets #V5 to square root of line length
ABS	absolute value	<code>#EVAL(#V5=ABS(#XPOS))</code> sets #V5 to absolute value of #XPOS . If #XPOS=-2.5 , #V5 would equal 2.5
SGN	return sign (+-)	<code>SGN(#XPOS)</code> returns sign (+ or -) of #XPOS . If #XPOS=-2.5 , this would return a minus (-) sign
INT	(integer value)	<code>#EVAL(#V5=INT(#XPOS))</code> sets #V5 to integer value of #XPOS . If #XPOS=2.5 , #V5 would equal 2

Trigonometric Functions Supported by SmartCAM

SmartCAM supports the following list of trigonometric functions. Notice that the # character does not precede the template words (**SIN**, **COS**, **TAN**, **ATAN**).

Operator	Meaning	Example
SIN	sine	#EVAL (#V5=SIN (#LNANG)) sets #V5 to sine of line angle
COS	cosine	#EVAL (#V6=COS (#LNANG)) sets #V6 to cosine of line angle
TAN	tangent	#EVAL (#V6=TAN (#LNANG)) sets #V6 to tangent of line angle
ATAN	arctangent	#EVAL (#V6=ATAN (#YOV/#XOV)) sets #V6 to arctangent of #YOV/#XOV

Order of Operations

SmartCAM uses the following standard math convention for order of operations:

1. Calculate exponential.
2. Multiplication or division left to right.
3. Addition or subtraction left to right.

You can control the order of operations by using parentheses. SmartCAM completes calculations in the innermost parentheses first. In the following example:

```
#EVAL ( #V6 = ( 2 * ( 3.14159 * #ARAD ^ 2 ) ) )
```

SmartCAM performs the calculations in the following order:

1. Raises the arc radius (**#ARAD**) to the second power.
2. Multiplies the resulting value by pi (3.14159).
3. Multiplies the resulting value by 2.
4. Sets **#V6** to the resulting value.

System-Defined Template @ Sections

The following list identifies the system-defined template @ sections SmartCAM uses:

Standard Sections:	Circular Interpolation:
@ARC	@XZARC
@CORNER	@YZARC
@CYCLCHG	Special Mill Sections
@END	@ZCHKMV
@ENDPROF	@ZCLRMV
@LINE	@ZDPTMV
@RAP	Fixed Mill Cycle Sections
@START	@FXD1
@STPROF	@FXD2
@TOOLCHG	@FXD3
@TRAVERSE	@FXD4
Fixed Lathe Cycle Sections	@FXD5
@FXD1	Special Punch Sections:
@FXD2	@ATTCHMT
@FXD3	@PNCHTL
@FXD4	@SHPRF
@FXD5	@TORCH
@FXD6	Lookup table for speed codes:
@FXD7	@SPEEDS
Helical Interpolation:	Change in Work Plane:
@HELIX	@WKSYS
@XZHELIX	
@YZHELIX	

More information about these system-defined template @ sections is available on the following pages in alphabetical order. Your SmartCAM application package calls those @ sections appropriate to the processes. For example, SmartCAM calls @PNCHTL only when you are using a punch press.

@ARC

SmartCAM calls this section when it encounters an arc in the process model.

Example:

```
< #PLANE>< #MOV>< X#XPOS>< Y#YPOS>#EXLN
< I#XCTR>< J#YCTR>< F#FEED>
```

@ATTCHMT

SmartCAM calls this section for punch machines with attachments that do not require a tool change (such as Whitney).

The call for this section is made from the last position before the attachment tool. See also `.smf` question 68.

@CORNER

SmartCAM calls this section to output a corner-rounding command that is a requirement for some machines' cutter compensation function.

You enable SmartCAM to access this section with `.smf` question 129.

@CYCLCHG

This gives the user limited look-ahead capability. SmartCAM accesses this section prior to every other section. `#CUT`, `#ENDPROF`, `#RAP`, and `#STPROF` are set in `@CYCLCHG`

Accessed prior to every system-defined `@` section (`@START`, `@LINE`, `@RAP`, etc.)

This feature can give you look-ahead capability, among other things.

Nothing can be output from this section, but anything can be evaluated (`#EVAL`).

There are four special template words of use in this section that enable you to know what `@` section will be accessed next. They are `#STPROF`, `#ENDPROF`, `#RAP`, and `#CUT`.

These vars will equal either 1 or 0 (True or False), and only one of these can be set to 1 at any time.

@DECLARE

Accessed first for declaration of user-defined variables.

All user-defined variables *must* begin with a `#` followed by a letter. Supported characters are `a-z`, `A-Z`, `0-9`, and the underscore (`_`). User-defined variables should be lowercase (`a-z`) for faster processing.

Supported variable types are integer and decimal; use these instead of `#U` and `#V` vars. They are declared as follows:

```
#DEC #my_decimal_var
#INT #my_integer_var
#INT #spindle_dir// Will get spindle direction
//for tap cycles
```

☞ For more information, see “User-Definable Template Variables” on page 6-48.

@END

This is the last section SmartCAM calls after all the elements in the database have been processed. It usually contains a return to home move, machine stop code, and tape rewind character.

Example:

```
M5 M9< #FXD>
G0 G40
M30
#OFFBLK%
```

@ENDPROF

SmartCAM executes this section when it encounters the end of a profile (a profile is any number of elements that connect in a sequential order).

This section usually contains tool retract, cancel cutter compensation, and end cutting cycle (turn torch off) commands.

Example:

```
< #MOV>< #DCOMP D0>< Z#ZPOS>
```

@FXD1-@FXD7

These sections are fixed cycle sections and execute when SmartCAM processes a machine fixed cycle. SmartCAM determines which @FXD section to call by the Tool Operation (#TLOP) number set in the job operations file and by the operation that corresponds to the element.

The purpose of the @FXD sections varies between machine types (see .smf questions 150-175).

The following is a listing of the @FXD cycles based on machine and operation:

	Mill	Lathe
@FXD1	drill cycle	thread cycle
@FXD2	spot drill with dwell	turning cycle
@FXD3	tapping cycle	facing cycle
@FXD4	bore cycle	tapping cycle
@FXD5	peck drill cycle	peck drill cycle
@FXD6		OD or ID groove cycle
@FXD7		face groove cycle

Details: The @FXD1 section will be used for Drilling, Reaming, and Special. Milling operations don't map to any hole @ sections in the .tmp file. If a hole is assigned a milling operation, the code calls @FXD sections based on the tool type. Code will call the @FXD section if it cannot find the @OP_ section.

```
< #RTNLVL #FXD>< X#XPOS>< Y#YPOS>#EXLN
< Z#ZDPTH>< R#ZCHK>< F#FEED>
```

@OP_DRL - This section could be used for Drilling.

@OP_REAM - This section could be used for Reaming.

@OP_SPEC - This section could be used for Special.

Details: The @FXD2 section will be used for Spot Drilling, Center Drilling, Counter Boring, Spot Facing and Counter Sinking. Use standard G82 from SMF.

```
< #RTNLVL #FXD>< X#XPOS>< Y#YPOS>#EXLN
< Z#ZDPTH>< R#ZCHK>#IF(#DWELL>0)#EXLN
< P#DWELL>< F#FEED>
```

@OP_SPDRL - This section could be used for Spot Drilling.

@OP_CDRL - This section could be used for Center Drilling.

@OP_CBORE - This section could be used for Counter Boring.

@OP_SPFACE - This section could be used for Spot Facing.

@OP_CSINK - This section could be used for Counter Sinking.

For more information, see the @OP_ sections.

@HELIX

SmartCAM calls this section when it encounters a helix in the process model.

For controls that support helix canned cycles as set with .smf question 280.

Example:

```
< #PLANE> #MOV< X#XPOS>< Y#YPOS> Z#ZPOS #EXLN
I#XCTR J#YCTR< F#FEED>
```

@LINE

SmartCAM calls this section when it encounters a line in the process model or needs to output a line move.

Example:

```
< #PLANE>< #MOV>< #DCOMP#EXC D#DOFF>#EXLN
< X#XPOS>< Y#YPOS>< Z#ZPOS>< F#FEED>
```

@OP_BORE

This section is used for Boring. See also @FXD4.

Use G85, G86, G87, OR G88 based on JOF.

#CYCLE can equal 0, 1, 2, or 3 and gets command string from the @BORE_CODES section.

Example:

```
#EVAL(#U19=#TABLE(BORE_CODES,#CYCLE))
#IFCHG(#FXD)<#RESET(#U19)>// Make sure the
//      cycle code gets output after a G80.
< #RTNLVL #U19>< X#XPOS>< Y#YPOS>#EXLN
< Z#ZDPTH>< R#ZCHK>#IF(#CYCLE=3)#EXLN
<< P#DWELL>>< F#FEED>

@BORE_CODES
0,G85
1,G86
2,G87
3,G88
```

@OP_CBORE

This section could be used for Counter Boring. See also @FXD2.

@OP_CDRL

This section could be used for Center Drilling. See also @FXD2.

@OP_CSINK

This section could be used for Counter Sinking. See also @FXD2.

@OP_DRL

This section could be used for Drilling. See also @FXD1.

@OP_PDRL

This section will be used for Peck Drilling. See also @FXD5.

Use G73 or G83 based on JOF.

#CYCLE can equal 0, 1, 2 or 3, and gets command string from the PECK_CODES section.

Example:

```
#EVAL(#U19=#TABLE(PECK_CODES,#CYCLE))
#IFCHG(#FXD)<#RESET(#U19)>// Make sure the
//      cycle code gets output after a G80.
< #RTNLVL #U19>< X#XPOS>< Y#YPOS>#EXLN
< Z#ZDPTH>< R#ZCHK Q#PECK>< F#FEED>
```

@OP_REAM

This section could be used for Reaming. See also @FXD1.

@OP_SPDRL

This section could be used for Spot Drilling. See also @FXD2.

@OP_SPEC

This section could be used for Formhole Making. See also @FXD1.

@OP_SPFACE

This section could be used for Spot Facing. See also @FXD2.

@OP_TAP

This section is used for Tapping. See also @FXD3.

Use G74 or G84 based on JOF.

Section tl(ccw) can equal 0 or 1 and gets command string from the TAP_CODES section.

Example:

```
#EVAL(#spindle_dir=tl(ccw))// Set user-
//      defined variable to JOF data.
#EVAL(#S19=#TABLE(TAP_CODES,#spindle_dir))
#IFCHG(#FXD)<#RESET(#S19)>// Make sure the
//      cycle code gets output after a G80.
< #RTNLVL #S19>< X#XPOS>< Y#YPOS>#EXLN
< Z#ZDPTH>< R#ZCHK>< Q#FEED F#FEED>

@TAP_CODES
0,G84
1,G74
```

@PNCHTL

SmartCAM uses this section when the selection of a torch or attachment is made with an offset number and not a tool change. SmartCAM calls this section after using a torch or attachment prior to calling a punch tool. It is useful for outputting a punch offset number (for example, E00).

@RAP

For most machines, this section is called for processing rapid moves only. This includes moves to the beginning of a profile or to a point. With punch presses, this section generally processes rapid moves, points, and holes (single punch hits).

Example:

```
< #MOV>< X#XPOS>< Y#YPOS>< Z#ZPOS>
```

@SHPRF

SmartCAM uses this section for machines supporting a special shearproof nibble cycle.

@SPEEDS

This section contains a speed table used for lathe controllers without direct RPM coding ability. To enable this section, .smf question 52 must be set to <1> Yes.

@START

SmartCAM executes this section before processing any other sections. The output from this section can include a tape rewind stop character, a command to turn block numbers on, a safe start block, a call for the first tool to be used, and the first rapid move of the program.

Example:

```
%
N1 O1111
#ONBLK G0 G17 G40 G80 G90 E1
M6 T#TOOL (#TDESC)
// #COOLNT can equal 0, 1, 2, 3, or 4 and
// gets command string from the COOLANT section.
#SPNDL #TABLE(COOLANT,#COOLNT) S#SPEED
X#XPOS Y#YPOS
Z#ZPOS H#LOFF
```


@STEPCHG

When there is more than one step assigned to the same tool, a step change occurs. This section is accessed when a step sequence changes, but the actual tool does not.

Anything can be output from this section, and anything can be evaluated (**#EVAL**).

When a step change is encountered, the process will first go to the **@STEPCHG** section and output those values that have changed. Upon leaving the **@STEPCHG** section the process will continue to the next appropriate section.

When only a tool change is encountered, the process will go to the **@TOOLCHG** section of the template and process all required values. **@STEPCHG** will not be processed.

@STPROF

This section is executed at the start of a contour profile (a profile is any number of elements that connect and occur sequentially). It is usually used to turn on cutter compensation and start any cutting cycle commands (for example, to turn a torch on).

Example:

```
< #MOV>< X#XPOS>< Y#YPOS>
```

@TOOLCHG

This section is processed for tool changes.

Example:

```
< #FXD> G40 M5 M9
M6 T#TOOL (#TDESC)
G0 G17 G40 G80 G90 E1
// #COOLNT can equal 0, 1, 2, 3, or 4 and
// gets command string from the COOLANT
// section.
#SPNDL #TABLE(COOLANT,#COOLNT) S#SPEED
X#XPOS Y#YPOS
Z#ZPOS H#LOFF
```

@TORCH

This section is processed when switching from a punching tool to a torch. See `.smf` question 230.

@TPINDX

SmartCAM calls this section when it encounters a plane change.

@TRAVERSE

This section allows the user to include any preparatory commands that are necessary between an **@ENDPROF** section and a **@STPROF** section.

This is similar to **@RAP**, which is called between other sections.

@WAIT

This section is processed when SmartCAM encounters a wait state in 4-axis turning.

@WKSYS

SmartCAM calls this section when it encounters a new plane. See `.smf` questions 66, 68 and 69.

@XZARC

SmartCAM calls this section when it processes an arc on the XZ plane.

Example:

```
< #PLANE>< #MOV>< X#XPOS>< Z#ZPOS>#EXLN
< I#XCTR>< K#ZCTR>< F#FEED>
```

@XZHELIX

Example:

```
< #PLANE> #MOV< X#XPOS> Z#ZPOS< Z#ZPOS> #EXLN
I#XCTR K#ZCTR< F#FEED>
```

@YZHELIX

Example:

```
< #PLANE> #MOV Y#YPOS< Z#ZPOS>< Z#ZPOS> #EXLN
J#YCTR K#ZCTR< F#FEED>
```

@YZARC

SmartCAM calls this section when it processes an arc on the YZ plane.

Example:

```
< #PLANE>< #MOV>< Y#YPOS>< Z#ZPOS>#EXLN
< J#YCTR>< K#ZCTR>< F#FEED>
```

@ZCHKMV

SmartCAM executes this section when it encounters a Z-check move. The Z-check level is set by `.smf` question 83 or by the `#CHKD` template word.

Example:

```
< #MOV>< Z#ZPOS>< #FXD>
```

@ZCLRMV

This section is accessed whenever a Z-clearance move is required.

Example:

```
< #MOV>< Z#ZPOS>< #FXD>
```

@ZDPTMV

SmartCAM executes this section when it encounters a move to Z level (generally used only for milling).

Example:

```
< #MOV>< Z#ZPOS>< F#FEED>
```

Subprogramming Application @ Sections

@ENDEF

This section is executed at the end of a pattern subroutine definition.

@FXDDEF

SmartCAM calls this section to output a drill subroutine's definition.

@GOSUB

SmartCAM calls this section for subroutine support output by the main routine when a pattern element is found.

@SUBDEF

This section is executed at the start of a pattern subroutine definition.

All subprograms are coded to separate files prior to coding the main program. The name of each sub file is the pattern name used inside the model file with a `.sub` extension. To add a `.sub` extension to these files, include the following evaluation sequence in the `@GOSUB` section:

```
#EVAL( #S0=#SNAME" . SUB" )
#INCLUDE( #S0 )
#REPEAT( #SREPT )<#INCLUDE( #S0 )>
```

Can be used to repeat the sub call #SREPT number of times.

☞ The variables SmartCAM loads in the @GOSUB section are the following:

```
Sub Start Point - #XST, #YST, #ZST (new, was #XPOS,...)
Sub Ending Point - #XPOS, #YPOS, #ZPOS
Sub Handle Point - #XCTR, #YCTR, #ZCTR
```

Sample User-Defined Template Sections

The following are examples of user-defined template sections you may want to use in your .tmp file:

@REPO

Use this section to output code to reposition the sheet of material.

The user command looks like this: @REPO(#XSET=50), where @REPO calls the template section, and #XSET is the amount of repositioning.

@STOP

Use this section to output a special machine stop (for example, to clear scrap). You can include a special message to the machine operator.

The following example shows a solution for a punch-specific stop section for material removal.

```
@STOP
#EVAL( #PUNCH=0 )
#EVAL( #MOV=0 )
< #MOV< X#XPOS><><> Y#YPOS><><> #PUNCH>>
#IF( #U9=1 )< ( REMOVE SCRAP )>
#IF( #U9=2 )< ( REMOVE FINISHED PART )>
#IF( #U9=3 )< ( REMOVE PUNCH, DIE, AND STRIPPER )>
#IF( #U0>3 )< ( MACHINE STOPPED, CHECK MACHINE )>
M00
```

Template Word Reference

This chapter contains an alphabetical list of all template words available in SmartCAM, a list of the words organized by category, and information about user-definable template variables.

SmartCAM does not use all of these words for each application. Some template words have different meanings in different applications. For example, #XST in lathe threading is the starting X value of the root thread pass, but for lathe grooving it is the start corner X position, and for the @ARC and @LINE sections it is the start of the element.

☞ Contents of these template words comes from the CNC Process Model database unless otherwise noted.

Alphabetical List of Template Words

#ABSI

Type	Options	Related .smf Questions
Switch	0=absolute 1=incremental	25, 26, 27, 299

Description: Output absolute or incremental G-code and coordinate information depending upon setting.

Example:

Function	Description
#EVAL(#ABSI=1)	Sets system to incremental output.
#ABSI	Outputs incremental positioning code set in .smf #27.

#ADIR
#BDIR
#CDIR

Description: The direction of rotation.

#AITOOL

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Switch	0=Off 1=On	

Description: Set to On when an auto-indexing tool is active.

#AND

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Control		

Description: Logical AND operator, < >

Example:

Function	Description
< #MOV#AND X#XPOS>	If both #MOV and #XPOS have changed, output them both, otherwise don't output anything.
#IF(#V6=#XPOS,#AND#V7=8)<#EXIT>	If #V6 = #XPOS and #V7=8, then exit this section.
#IF(#U0=2,#AND#V9>5)< M00>	If user word #U0 is 2, and user word #V9 is greater than 5, output an M00.

Details: A comma must be placed before #AND inside parentheses.

#ARAD

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric	>0	22, 99, 120

Description: Arc radius value. 3-D start radius for helix. Where the arc has left/right offset, the original arc radius is modified by the cutter radius.

Example: <R#ARAD>

#BDSTR

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
String		17

Description: Used to output block delete string.

#BLK

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric	>0	11 through 16

Description: Current block number.

#BLKDEL

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Switch	0=Off 1=On	17

Description: Turns output of a block delete string on or off.

#C0-#C5

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Switch (User)		360 through 371

Description: Outputs a text string SET in .smf questions #360–371, depending on setting. Useful for controlling coolant codes and spindle ranges.

Example:

```
#IF( #SPEED<=2500 ) <#EVAL( #C1=1 )>
```

See chapter 5 for information about user-assigned template words in SmartCAM.

#CALL()

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Control		

Description: Call another template section and return to this statement afterward. Use the following format:

```
#CALL(section name)
```

where:

#CALL is the call function.

() contain the section to be called, either spelled literally or contained in a string word.

Details: The @ character is not included in the call statement section name.

Example:

```
IF (#TLOP=2) <#CALL(RAP)>
```

goes to **@RAP** section.

#CENG

Description: Outputs the command for C axis engage or disengage as specified in .smf questions 449 and 450.

#CHKD

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		83

Description: Holds the value for tool check distance.

#CLEAR

Description: Absolute Z coordinate of element's clear value.

#COOLNT

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Switch	0=NONE 1=FLOOD 2=MIST 3=THRU 4=AIR	N/A

Origin: Job operation setup

Description: Assigned in the operation field op(coolant). See #TABLE on page 6-24 for information on calling tables.

Example:

```
@COOLNT:
0,M09 //NONE
1,M08 //FLOOD
2,M07 //MIST (if applicable)
3,M08 //THRU (if applicable)
4,M08 //AIR (if applicable)
```


#CSSDIA

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric	>0	

Description: Lathe X-axis diameter value. Controlled by the speed mode set for each operation in the job operation setup. Output conditionally based on the CSS speed mode being active.

#CSSRAD

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric	>0	

Description: Lathe X-axis radius value.

#CTRLLOC

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Switch	0=Off 1=On	100, 101 100, 102

Description: Outputs center-location commands (.smf questions 101 and 102) for arc radius command programming (.smf question 100).

#CUTC

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Switch	0=Off 1=On 2=Off 3=On	110 111 112 113

Description: Turn contour cutting code (on/off). These are switched on and off automatically at the beginning and end of a profiling contour.

#CYCLE

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Integer	Values 0-4	

Origin: Job operation setup

Description: Assigned in the operation field op(cycle).

Details: An extension to the #FXD word to support extended cycles.

#CYTIME

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric	>0	141, 142, 144

Description: Total cycle time so far.

#DATE

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
String		
<i>Origin:</i> System clock		
<i>Description:</i> Current system date.		
<i>Output format:</i> mm/dd/yy		
<i>Example:</i> 1/20/94		

#DCOMP

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Switch	0=None	121, 123
	1=Left	121, 124
	2=Right	121, 125

Description: Outputs cutter compensation code from cutter offset assigned to each element.

#DEC

Description: Used in **@DECLARE** section in order to declare a decimal value (see “User-Definable Template Variables” on page 6-48).

#DOFF

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Integer		
<i>Origin:</i> Job operation setup		
<i>Description:</i> Diameter offset register number set in the DOFF field in the Edit Process Step dialog box. Will be the #TOOL if unassigned.		

#DWELL

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		50

Origin: Job operation setup

Description: Dwell time for dwell or spot drill fixed cycle set in the Operation tab.

Details: You can calculate this in template file before output. The following example calculates and outputs a dwell time equal to two spindle rotations.

Example: `#EVAL(#DWELL=(1/#SPEED)*2)T#DWELL`

#EANG

Type	Options	Related .smf Questions
Numeric	-360–360	24

Description: End angle of arc, hole pattern, or helix.

#EBLK

Description: Sub end block number

#ELSE

Type	Options	Related .smf Questions
Control		

Description: Logical ELSE operator. IF first condition FALSE, output following <> conditional. Note that this word can be used with the standard conditional test or the #IF() word.

Example:

Function	Description
<#TOOL>#ELSE<00>	If #TOOL's value has changed, then output it, otherwise output 00.
< T#NTOOL>#ELSE< T0>	If the value in #NTOOL (next tool) has changed, output T followed by that value, otherwise output T0. (This situation occurs only during the last tool in the data base).
#IF(#U0=2) < M00>#ELSE <M02>	If user word #U0 is 2, output an M00; otherwise, output M02.
#IF(#NTOOL>10) <#EVAL (#U1=1)>	If the next tool number used is greater than 10,
#ELSE<#EVAL(#U1=0) >	THEN set #U1=1, ELSE set #U1=0

#ENAME

Type	Options	Related .smf Questions
String		120

Description: Element name for current element. This is only available for elements which are not offset or are coded to the part profile.

#EVAL()

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Control		

Description: Evaluates math and string assignments. The parentheses must enclose any math or string assignment. #EVAL() only performs the calculation; it does not output any information. You must provide a separate statement to output the results of the calculation.

#EXC

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Control		

Description: Exclude any following template word from the implied conditional test whether its content has changed or not.

Example:

```
@LINE
<#DCOMP#EXC D#DOFF>
```

#EXIT

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Control		

Description: Exits a section early. Generally used with the #IF() conditional.

#EXLN

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Control		

Description: Extends the line you are typing in the template file to the next line. SmartCAM combines both lines into the same line of NC code.

Example:

```
< #MOV><>< X#XPOS><>< Y#YPOS><>< F#FEED><><#C1>#EXLN
< #PUNCH>#IF(#TLOP=3)< M00>
```

#FDADJ

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Switch	0=Off 1=On	47, 48 47, 49

Description: Adjust feed rate for periphery of arcs. By making assignments to this template word, SmartCAM automatically adjusts the feed rate if you enable the .smf settings.

#FDMODE

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Switch	0=Primary 1=Secondary 2=Inverse time	44, 45, 438

Description: Output primary or secondary feed data from the Feed Mode field in the Edit Process Step dialog box. This is set by the feed type given for each tool in the job operation setup.

#FDMULT

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		41

Description: Feed rate multiplier for Z depth moves.

Details: Initial setting is set by .smf question 41.

#FEED

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric	Default = op (feedupr)	40, 41

Origin: Job operation setup

Description: Feed rate of current operation. Initially set in the Operations tab. You can adjust the feed rate automatically using the template words #FDADJ and #FDMULT.

Details: Will use appropriate feedupr for xy milling or z plunge/ramp moves. Q41 does not apply in the JOF mode.

Example: <F#FEED>

#FILE

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
String		

Description: Name of code output file.

Example:

```
@START
#FILE
%
#ONBLK G40 G17 G80 G90
```

#FMT()

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Control		

Description: Used to force the numeric format of a template word.

Example:

```
#FMT( #V4 , T2 . 3 )
```

where

#FMT is the template word that forces the format.

#V4 is the word the format code controls.

T2 . 3 is the format code.

Details: Automatic inch/metric conversion is not performed on words formatted with #FMT.

#FTHRD

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		187

Origin: Process model thread element pitch.

Description: Feed (lead) for thread.

Details: Lathe only. Used by @FXD1 section.

#FXD

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Switch	Mill:	Mill: 160, 471
	0=Cancel	161
	1=Drill cycle	162
	2=Spot drill cycle with dwell	163
	3=Tap cycle	164
	4=Bore cycle	165
	5=Peck drill cycle	166
	Lathe:	Lathe: 180
	1=Thread cycle	186
	2=Turn cycle	181
3=Facing cycle	182	

Origin: Job operation setup

Description: Used to output the fixed cycle or cancel fixed cycle codes. The fixed cycle codes are specified in the corresponding SMF Questions.

Details: The numeric values that are assigned to the #FXD word are determined as follows.

Milling: If SMF question #471 is set to 0 (JSF mode) then the numeric value is set equal to the #TLOP operation number. If the #TLOP is equal to 1 and the hole element has a pecking turned on, then the #FXD value is set to 5.

If question #471 is set to 1 than the numeric value is based on the operation type as shown below:

- 1=Drilling, Reaming, Special Hole
- 2=Spot Drilling, Center Drilling, Counterboring, Spot Facing, Counter Sinking
- 3=Tapping
- 4=Boring
- 5=Peck Drilling

Turning:

- 1=Thread tool and thread element
- 2=Turning tool and horizontal element
- 3=Facing tool and vertical element

#IF()

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Control		

Description: Logical IF testing in the following format:

```
#IF(this is true)<then do this>#ELSE<do something else>
```

where

#IF is the logical IF operator, () contains the condition to be tested, and <> contains the operation to perform.

The #ELSE operator is optional. See also discussion of #ELSE.

Example:

Function	Description
#IF(#U0=1)<#EXIT>	IF the value of the user word #U0 is 1, exit this section.
#IF(#TLOP=2)<#CALL(TORCH)>	IF the value of #TLOP is 2, branch to the @TORCH template section, execute the commands in that section, and return.
#IF(#U0>2)<M02>#ELSE<M01>	If #U0 is > 2, then output M02, otherwise output M01.

The condition to be tested can also include the logical **#AND** and **#OR**. See the following examples, and also see **#AND** and **#OR** in this chapter.

Function	Description
<code>#IF(#V3=5,#AND #U1=0)<#EXIT></code>	IF #V3 is equal to 5, AND #U1=0 , THEN exit this section now.
<code>#IF(#V3=5,#OR #V3=2)<#EXIT></code>	IF #V3 is equal to 5, OR if #V3=2 , THEN exit this section now.

☞ **#IF** statements can be nested, as in the following example:

```
@TOOLCHG
#IF(#U0=0)<#IF(#TLOP=2)<(REMOVE SLUG)
M18#EVAL(#U0=1)>>
```

This statement performs the same test and output as the above example, but only if the value of **#U1** is 0. In addition, it sets **#U0** to 1 to ensure that the next time this statement is encountered no output will occur.

#IFCHG()

Type	Options	Related .smf Questions
Control		

Origin: Job operation setup

Description: Provides the user with a means to test if a variable has changed or would be output.

Example:

```
#IFCHG(#V1)<output body of code>#ELSE<#EXIT>
#IFCHG(#CYCLE)<#EVAL(#DRLCYCLE=#TABLE#EXLN
(DRILL_CODES,#CYCLE))>
#IFCHG(#CYCLE,#AND #FXD)<>
#IFCHG(#CYCLE,#OR #FXD)<>
```

☞ If a statements requires more than one line, you can use the **#EXLN** word to continue to the next line.

#IFSTR()

Type	Options	Related .smf Questions
Control		

Description: Logical **IF** test for string variables. Same operation as **#IF()** except you use strings in the test instead of numeric comparisons.

#INC

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		

Description: Angular or linear increment between holes for @BHC, @LAA or @GRID; any increment for @LINE and @ARC (for punch, increment between hits for @LINE and @ARC); for 3-D milling, pitch in depth axis for each full revolution of helix.

#INC2

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		22

Description: Increment in secondary direction for @GRID; for 3-D, radial pitch for each full revolution of helix.

#INC3

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		

Description: Rise per radian for helix.

#INCLUDE()

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Control		

Description: Incorporate an external file into code output and automatically number the blocks if #ONBLK is on. May be a name or a string word such as #S1 or #SNAME.

#INDXA

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		Punch: 248 3-D: 291, 298–302, 304

Description: For punch, index angle for auto-index punch stations; for 3-D, index angle for A-axis rotary motion.

#INDXB

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		Punch: 248 3-D: 291, 298–304

Description: For punch, tool angle from job operation setup for non-auto-indexing tools; for 3-D, index angle for B-axis rotary motion.

#INDXC

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		3-D: 291

Description: Index angles for C-axis rotary tables.

#INO**#JNO****#KNO**

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		290, 299

Description: Components of tool plane normal vector.

#INT

Description: Used in **@DECLARE** section to declare a variable as an integer (see “User-Definable Template Variables” on page 6-48).

#IPECK

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric	Default =0	Overrides 152

Origin: Job operation setup

Description: Otherwise known as initial peck; assigned in the operation field.

#LEVEL

Description: Returns the absolute Z-level coordinates of an element.

#LNANG

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		

Description: Angle of current line or rapid move.

#LNLEN

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		

Description: Length of line element. Distance between current and previous point or hole.

Details: Not available for **@LAA** or **@GRID**. You must use **#UPDATE** to update **#XPOS** and **#YPOS**.

#LOFF

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		

Description: Length offset register number assigned in the Edit Process Step dialog box in the LOFF field.

Details: Will be the #TOOL if unassigned.

#LTOOL

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		

Description: Number of last tool coded.

#MOV

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Switch	0=Rapid	80
	1=Linear	85
	2=CW arc	92
	3=CCW arc	93
	4=Linear	243
	5=CW	244
	6=CCW	245
	45=CW helix	285
	46=CCW helix	286

Description:

Outputs rapid, line, or arc code.

3-D: cw/ccw helix command.

#NBLMD

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Switch	0=Punch	240
	1=Nibble	241

Description: Output punch or nibble code depending on .smf question 235.

#NEXTPT

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Control		

Description: Causes SmartCAM to skip to next element in database. Use this to incorporate more than one model element in the current section. Use #NEXTPT followed by the additional code to be processed using the second element. Also use it to ensure that the current element is updated after the @START and @TOOLCHG sections are processed. In this case, place #NEXTPT as the last item in the section. It is also useful for skipping over the coordinate information attached to a user command, so you use only the user command text.

Example:

```
@REPO
X#XSET
#NEXTPT
```

#NHOL

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		

Description: Number of holes for @BHC, @LAA, or @GRID primary direction; for punch, number of hits for @LINE or @ARC.

#NHOL2

<i>Type</i>	<i>Option</i>	<i>Related .smf Questions</i>
Numeric		22

Description: Number of holes in the secondary direction for @GRID.

#NTOOL

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		

Description: Number of next tool to be coded.

Details: Useful for tool ready commands.

#OFFBLK

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Control		10, 11, 14

Description: Turns off block numbering.

Example:

```
@END
G49 G90 X120 Y8.0
M30#OFFBLK
%
```

Details: Should not be embedded in conditionals.

#ONBLK

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Control		10, 11, 14

Description: Turns on block numbering.

Example:

```
@START
%#ONBLK
G49 G40 G90 G17
```

#OPTYPE

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Integer	Default = 0 101 Rough Milling 102 Finish Milling 103 Face Milling 104 Surface Milling 105 Edge Milling 106 107 Copy Milling 201 Spot Drilling 202 Drilling 203 Peck Drilling 204 Reaming 205 Boring 206 Tapping 207 Counter Boring 208 Counter Sinking 209 Spot Facing 210 Formhole Making 211 Center Drilling	471

Origin: Job operation setup

Description: The #OPTYPE word is set to a unique value for each operation that is available.

Details: Milling only. Assigned based on the Operation tab in the Edit Process Step dialog box recorded within the JOF file.

#OR

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Control		

Description: Logical OR operator

Example:

Function

< #MOV#OR X#XPOS>

Description

If either #MOV or #XPOS has changed output them both; otherwise don't output anything.

#IF(#U0=#LTOOL, #OR#U0<1) <#EVAL(#V1=1)>

If #U0 = #LTOOL or it is less than 1, then set #V1 = 1.

#IF(#U0=2, #OR#V9>5) <M00>

If user word #U0 is 2, or user word #V9 is greater than 5, output an M00.

☞ When inside the () a comma must be placed before #OR.

#OVANG

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		130

Origin: SmartCAM

Description: This is the angle of the offset vector in degrees. It is used to calculate #XOV, #YOV and #ZOV.

#PCAN

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Control		

Description: Output punch off for previously active mode. Used with @ATTCHMT and @PNCHTL when switching between primary and secondary punching operations.

#PECK

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric	>0	150

Description: Peck drill increment for peck drill cycle (@FXD5).

Details: Overrides .smf question 150. Assigned in the Operation tab.

#PLANE

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Switch	0	270, 271
	1	273
	2	275

Description: Orthogonal plane specification command.

#PRMRY

Description: Stores the value of the primary turret. (Turning only.)

#PSETLEN

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric	Default = 0	N/A

Origin: Job operation setup

Details: Assigned in the length preset field on the Tool tab of the Edit Process Step dialog box..

#PTOP

Description: The absolute Z coordinates of the element's profile top plus #CHKD.

#PUNCH

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Switch	-3=Off	225
	-2=On	226
	-1=Inhibit	227
	0=Off	222
	1=On	223
	2=Inhibit	224

Description: Output punch on and off codes. SmartCAM generally switches #PUNCH on for holes and nibbling moves and off for points, tool changes, and at end of program.

#QANG

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		24

Description: Q inclination value, taper angle perpendicular from primary move. Positive values are to the right and negative values to the left.

Details: Wire EDM only.

#RAD2

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		

Description: Secondary radius, and radius of arc on the secondary profile.

Details: Wire EDM only.

#RANG

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		24

Description: R inclination value, angle parallel from primary move. Positive values are ahead of primary end point and negative values are behind.

Details: Wire EDM only.

#REPEAT()

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Control		

Description: Repeat the information in the conditional a specified number of times. A value of 0 will skip the conditional entirely. Use the following format:

```
#REPEAT(count)<operation>
```

where

#REPEAT is the repeat function.

() contains the count as a number, numeric word, or expression.

<> contains the operation to perform count times.

Example:

```
#REPEAT(12)<#CALL(REPO)>
```

calls the section @REPO 12 times.

```
#REPEAT(#U2)<0 >
```

outputs #U2 0's.

#REPO

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		

Description: The amount of X axis movement for repositioning operations on a punch press or profiler. Set by the REPO.MCL system macro file or user command.

#RESET()

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Control		60

Description: Resets the old value of a template word. This forces the word to be output the next time SmartCAM encounters it, whether it has changed or not. It's format is as follows:

```
#RESET( #word1 , #word2 , etc. )
```

where

#RESET is the reset function.

() contain the list of template words to reset.

Example:

```
#RESET( #PUNCH )
#RESET( #XPOS , #ZPOS )
```

#RFEED

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		40, 81

Description: For machines that use a rapid feed string (Whitney Punch Press) for rapid moves instead of a separate rapid move code. See a Whitney code generator for an example of this word's use.

#ROT1

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		

Description: Angular orientation of a sub call.

#ROT2

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		

Description: Angular increment of a rotated repeated sub.

#RTNLVL

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric	0=Z Check	173, 174
	1=Z Clear	173, 175

Description: Return level for fixed cycles. Outputs Z check or Z clear depending on switch value.

#S0-#S19

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
String		

Origin:

Application (User Command)
 job operation setup (can be placed in the notes and comments field)
 Template (.tmp) file

Description: String words are used to store and output test. Use #IFSTR() to text equality.*Example:* #EVAL(#S1=REMOVE PUNCH, DIE AND STRIPPER)

☞ When assigning a string word in a user command, no other word assignment may occur. The evaluator will assume whatever follows is part of the string.

#SAFBLK

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Control		12, 15

Description: Output safe start block number.*Example:*

```
@STOP
#SAFBLK
```

#SBLK

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		337

Description: Sub start block number**#SBTYP**

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric	0 = None 1 = Regular 2 = Drill subs	

Description: System flag identifying subroutine type.**#SLDWN**

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Switch	0=Slowdown off 1=Slowdown on	Punch: 131-133

Description: When a sharp corner slow down condition exists, #SLDWN is set to 1. The values that determine a sharp corner are set with .smf question 132 and 133.

#SNAME

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
String		

Description: Sub name (Does not include the .sub file extension.)

#SPEED

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		52

Origin: Job operation setup

Description: Speed format set for current tool. If .smf question 52 is set to 1, the speed is taken from the Edit Process Step dialog box. This speed table is a template section, called @SPEEDS. See also @SPEEDS or .smf question 52 for more information. Speed can be adjusted in the application using a user command. Overrides .smf question 150 in JOF mode. Assigned in the Operation tab.

#SPMODE

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Switch	0=RPM 1=CSS	57 58

Origin: Job operation setup

Description: Output RPM or CSS speed code. This is initially set by the speed mode you specify in the Edit Process Step dialog box.

#SPNDL

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Switch	0=Off 1=On CW 2=On CCW	54 55 56

Origin: Job operation setup

Description: Output appropriate spindle code. This is initially set in the Tool tab in the Edit Process Step dialog box. You can also set this using math and logic statements. Though you can set this with a use command, we recommend that you automate your template file whenever possible using math and logic statements.

#SPOFF

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Control		54

Description: This command will output the spindle-off code if the spindle changes direction between tools.

#SREPT

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		330

Description: Number of times to repeat a sub call after the first time. Note that #SREPT will be 0 for the subprogram to be executed only one time.

#STA**#STB****#STC**

Description: The start angle.

#STANG

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		24

Description: Starting angle for an arc or hole pattern (@BHC).

3-D: start angle of helix.

#SYNCH

Description: Merge block codes state flag.

#SYSTIME

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		

Description: Current system time, placed at the beginning of code. Set from the system clock like #DATE.

#TABLE()

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Control		

Description: Returns information from a table within the template file. Useful for calling multiple speed tables. Use the following format:

```
#TABLE(table, value)
```

where

#TABLE is the table function.

() contains the table section name and the value to look up.

Example:

```
#TABLE (NAME , #V1 )
```

where NAME is the template section name and #V1 is the value to look up.

Details: If you are calling a single speed table call it @SPEEDS and use .smf question 52 to automatically call the table instead of using this question. Using this template word and .smf question 52 to call the table @SPEEDS results in the speed being run through the table twice resulting in improper speed code.

Example:

```
#EVAL ( #S0=#TABLE (TABLE , #V1 ) )
```

#TANG

Type	Options	Related .smf Questions
Numeric		24

Origin: SmartCAM

Description:

Total angle for an arc or hole pattern.
3-D: total angle of helix.

#TCODE

Description: Timing codes (Turning).

#TDESC

Type	Options	Related .smf Questions
String		

Origin: Job operation setup

Description: Tool description string as it displays in the related Tool tab in the Edit Process Step dialog box .

Example: 0.5 diameter end mill.

#THDPTCH

Type	Options	Related .smf Questions
Numeric	Default = 0	43

Origin: Job operation setup

Description: Thread (tap) pitch

Details: Assigned in the Tool tab.

#TIME

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		

Description: Value to be added to the total cycle time. Note that this word is not intended as an output word. Any value assigned to this word will be added to the total cycle time. Note that time is measured in decimal minutes.

Example:

1. To add #DWELL time to #CYTIME.
2. To add miscellaneous function time (parts catcher, pallet loader, etc.).

#TINDX

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric	0–4	

Description: A flag distinguishing tool plane transition types:

<0> = No tool plane change.

<1> = Transition between parallel tool planes with the same origin.

<2> = Transition between parallel tool planes with different origins (3-D offset).

<3> = Transition between nonparallel planes with the same origin (index move).

<4> = Transition between nonparallel tool planes with different origins (index move and 3-D offset).

Details: Use this variable with template file logic to determine when it is appropriate to exit template sections early or to call another template section. For example, if you wanted to use the @WKSYS section when a tool-plane change took place without an indexing move, you could add the following test to the @TPINDX section:

```
IF<INDX=2>CALL@WKSYS
IF<INDX=2>#EXIT
```

#TLCHG

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Control		62

Description: If .smf question 62 is set, tool- and operation-related words are not set with new values until #TLCHG is processed in the .tmp file.

Example:

```
@TOOLCHG
#IF( #LTOOL<>#TOOL ) <#TLCHG>
```

#TLCMT

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
String		

Origin: Job operation setup

Description: Tool comment string from Comment field on the Tool tab in the Edit Process Step dialog box .

Example: 2-flute R.H. HSS End Mill

#TLDIA

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		126–128

Origin: Job operation setup

Description: Tool diameter set on the Tool tab in the Edit Process Step dialog box.

#TLEN

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		

Origin: Job operation setup

Description: Tool (cut, flute, etc) length set on the Tool tab in the Edit Process Step dialog box. Identifies the cut length of tools when assigned in the Tool tab.

#TLGRAPHIC

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
String	Default = 0	N/A

Origin: Job operation setup

Details: Assigned in the Tool tab.

#TLID

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
String		

Origin: Job operation setup

Description: Tool ID string from the Tool tab in the Edit Process Step dialog box.

Example: B-1632-L

Details: Obsolete template word. Replaced by #TLGRAPHIC

#TLOP

Type	Options	Related .smf Questions
Numeric	Mill: 0=Face mill 0=Draft mill 1=Drill 1=Reamer 2=End mill 2=Ball mill 2=Counterbore 2=Countersink 2=Other 2=Spot drill 3=Tap 4=Bore Lathe: 1=Face 1=Turn 1=Bore 1=Other 2=Face groove 2=OD groove 2=ID groove 3=OD thread 3=ID thread 4=Drill 5=Tap Punch: 1=Round 1=Square 1=Rectangle 1=Obround 1=Diamond 1=Other 2=Torch 3=Attachment	

Origin: Job operation setup

Description: The #TLOP word is set to a number based on tool categories. The number is used to determine value of the #FXD word and which @FXD section will be used when canned cycles are output.

Details: Assigned based on the Tool tab in the Edit Process Step dialog box recorded within the JOF file.

#TLPAGEID

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
String	Default = 0	N/A

Origin: Job operation setup

Details: Assigned in the step field.

#TLSTA

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Integer	Default = 0	N/A

Origin: Job operation setup

Details: Assigned in the step fields.

#TLTIME

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		138–142, 144

Origin: Calculated by system.

Description: Cycle time for the previous tool.

Details: Active @TOOLCHG and @END.

#TLTYPE

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric	Mill: 1=Spot drill 2=Twist drill 3=Reamer 4=Tap 5=End mill 6=Ball mill 7=Face mill 8=Boring tool 9=Counterbore 10=Countersink 11=Form hole tool 12=Center drill 13=Form mill Turn: 1=Face 2=Turn 3=Bore 4=Face groove 5=OD groove 6=ID groove 7=OD thread	

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
	8=ID thread 9=Drill 10=Tap 11=Other	
	Fab: 1=Round 2=Square 3=Rectangle 4=Obround 5=Diamond 6=Torch 7=Attachment 8=Other	

Origin: Job operation setup

Description: The #TLTYPE word is set to a unique value for each tool that is available. These numbers are useful for helping to determine the type of tool being used.

Details: Assigned based on the Tool tab in the Edit Process Step dialog box recorded within the JOF file.

#TLWD

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		

Origin: Job operation setup

Description: Diameter of current tool. Same as #TLDIA. Usually used with non-round punch tools along with #TLEN.

#TOFF

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		

Origin: Job operation setup

Description: Combined tool and offset number (offset number follows tool number). Use for lathes that require the combination the tool number and offset number.

#TOOL

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		

Origin: Job operation setup

Description: Current tool number from the Tool tab. This is the same as the tool number shown in the application.

#U0-#U19

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric (User Integer)		
<i>Origin:</i>		
Application (User command)		
job operation setup (can be assigned in the Notes section)		
Template (.tmp) file (use #EVAL)		

Description: These words default to integer output format. They are usually used as binary switches (on/off) and as counters.

Example:

```
@TOOLCHG
#IF (#TLOP=2) <#EVAL( #U1=0 ) >
@RAP
#IF( #U1=0 ) <#EXIT>
```

#UOV

Description: X displacement component.

#VOV

Description: Y displacement component.

#WOV

Description: Z displacement component.

#UPDATE()

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Control		5

Description: Updates the internal old value which is checked for conditional output. This causes the system to assume that the value or position was output. With position words such as #XPOS, this updates the last position used to calculate incremental moves and arc centers. In certain types of code where #XPOS, #YPOS are not output directly, use one of the following update methods:

- Machine file item 5. Update template words after each block. A side effect of this option is that template words may be unintentionally updated and not output when expected.
- #UPDATE: If machine file item 5 is set to no, template words are not updated unless they are actually output to a file. #UPDATE allows the words to be used as if they were output to a file and actually were not.

Its format is as follows:

```
#UPDATE( #word1 , #word2 , etc. )
```

where:

#UPDATE is the update function.

() contains the list of template words to update.

Example:

```
#EVAL( #V5=( #XPOS*#XPOS*3.14159 ) )
C#V5 Y#YPOS
#UPDATE( #XPOS )
```

#V0–#V19

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		

Origin:

Application (User command)

job operation setup (can be assigned in the Notes section)

Template (.tmp) file

Description: Full double precision numeric words. Used for storing and retrieving coordinate information.

Example:

```
EVAL( #V4=#XPOS+#XSET )
X#V4
```

#WINCL

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Switch	0=Cancel	262
	1=Incl left	263
	2=Incl right	264

Description: Specifies wire-inclination state as viewed along the tool path. Outputs the strings in .smf questions 262 through 264.

#WKPLN

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
String		

Description: Tool plane name assigned to a work plane.

#WKSCHG

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Switch	0=Off 1=Plane change/ tool change combination	

Description: Set to 1 (On) when WKSYS combines with TOOLCHG or when the first position SmartCAM codes with @START is not assigned to the XY plane (ZX plane for Turning). Use this word for testing when SmartCAM encounters a new work plane and tool change.

#WRAD

Description: Holds the radius value of wrapped elements and is used for feed rate calculations.

#XCTR

Details: X pattern handle position, always absolute. For Subroutines.

Description: Arc center X/Y/Z position. Handle point for SUBS.
Lathe: also start X/Z value for root pass during treading (@FXD1).
3-D: helix center position X#XCTR.

#YCTR


Details: Y pattern handle position, always absolute. For subroutines.

Description: Arc center X/Y/Z position. Handle point for SUBS.
Lathe: also start X/Z value for root pass during treading (@FXD1).
3-D: helix center position X#XCTR.

#ZCTR

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		96, 97 Lathe: 98

Description: Arc center X/Y/Z position. Handle point for SUBS.
Lathe: XZ thread pass start for thread cycles root start before first pass.
3-D: helix center position X#XCTR.

 The .smf questions determine how these template words are defined.

#XFO
#YFO
#ZFO

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		

Description: Distance from the world coordinate origin point to a new work plane's origin point.

#XHOME
#YHOME
#ZHOME

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		69

Description: The coordinates of the first point in any application file are stored by these words. They can be called at any time. These are sometimes useful for moves to return to home or tool change position.

Example:

```
@END
X#XHOME Y#YHOME
M30
```

#XOV
#YOV
#ZOV

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		130

Description:

- Profiling on all machine types:
These are the X, Y and Z components of an offset vector. .smf question 130 controls whether this vector is normal or tangent to the move.
- Lathe Thread:
#ZOV:Delta Z for final (root) pass of lathe thread.
#XOV:Delta X for final (root) pass of lathe thread.
- Lathe Groove:
#ZOV:Width (depth for face groove).
#XOV:Depth (width for face groove).
- WEDM:
#XOV:the U axis incremental distance from primary X end point to secondary X endpoint.
#YOV:the V axis incremental distance from primary to secondary Y.

#XPASS

Details: X axis scale factor for a sub call

#YPASS

Details: Y axis scale factor for a sub call

#ZPASS

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		Lathe: 185

Details: Z axis scale factor for a sub call

- Lathe thread(@FXD1 cycle) .smf question 185 affects this word.
#XPASS: ending X value for root pass.
#ZPASS: ending Z value for root pass.
- Lathe groove(@FXD6 or 7 cycle).
#XPASS: bottom (of X side for face groove).
#ZPASS: side (or Z bottom for face groove).
- Lathe peck drill(@FXD5 cycle).
#ZPASS: bottom or end or tapping move to depth.
- Lathe tap(@FXD4 cycle).
#ZPASS: bottom or end of tapping move to depth.
- Lathe turn or facing(@FXD2 or 3 cycle).
#X/ZPASS: end X/Z value of pass.

#XPOFF**#YPOFF****#ZPOFF**

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		34; Lathe: 66 <4>

Description: X/Y/Z position offset values to be added or subtracted to all moves. Usually used for punch presses that reset all coordinates after repositioning.

#XPOS**#YPOS****#ZPOS**

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		Mill: 35 Lathe: 37, 38

Description: End X/Y/Z position value of current element from the application.

#XPSET**#YPSET****#ZPSET**

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		303

Description: Values for the absolute preset for a tool in new plane when.

#XSET**#YSET****#ZSET**

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		66, 67

Origin: Job operation setup, model file

Description:

Lathe: X/Z absolute preset position for new tool based on the previous work plane.
3-D: Machine (world) origin to new tool plane origin in rotated position.

#XST**#YST****#ZST**

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		238

Description:

Start X/Y/Z position for current move.
Lathe Groove: start corner X/Z position.

#ZCHK

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		83, 170, 171

Description: Z-check position value.

Details: Used with mill fixed cycles only.

#ZDPTH

<i>Type</i>	<i>Options</i>	<i>Related .smf Questions</i>
Numeric		172

Description: Current Z-depth position value.

Details: For fixed cycles only.

Lists of Template Words by Category

This section contains a list of template words divided into categories, which are arranged alphabetically. To find information about a specific template word, first consider what type of information it is (such as tool change information or fixed cycle information), and then look through the alphabetical listing of categories to find the one you want.

Each list has a template word in the left column and a definition of how the template word is used in the right column.

Arcs

#ARAD	Arc radius
#CTRLLOC	Center location
#EANG	End angle of arc (all angles in degrees)
#FDADJ	Feed adjust
#INDXA	Index angle
#LNLEN	Line length
#MOV	Move preparatory code
#OVANG	Offset vector angle
#PLANE	Arc plane designation
#STANG	Start angle of arc
#TANG	Total angle of arc
#XCTR	Center position of arc
#XOV	Offset vectors
#XPOS	End position of arc
#XST	Start position of arc
#YCTR	Center position of arc
#YOV	Offset vectors
#YPOS	End position of arc
#YST	Start position of arc
#ZCTR	Center position of arc
#ZOV	Offset vectors
#ZPOS	End position of arc
#ZST	Start position of arc

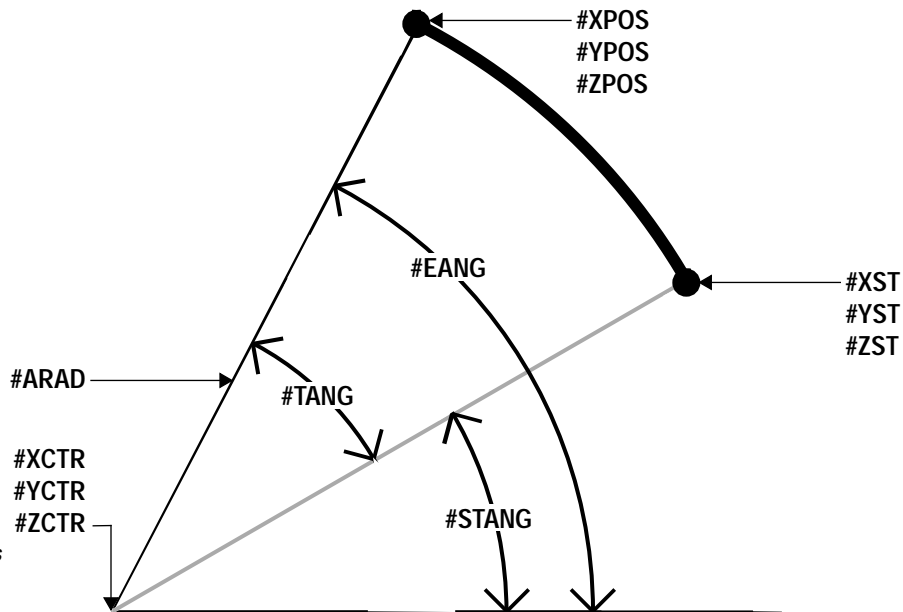


Figure 6-1
Use these
template words
for information
about arcs.

Block Number Control

#BDSTR	Delete block string variable
#BLK	Current block number
#BLKDEL	Turns block delete on or off
#OFFBLK	Turns block numbers off
#ONBLK	Turns block numbers on
#SAFBLK*	Safe start block number

Control Functions

#CALL()	Call another template section
#EVAL()	Perform expression contained in ()
#EXLN	Extend template line to next line
#FMT()	Force special format for template word
#INCLUDE()	Include lines from an external file for subroutines.
#NEXTPT	Advance to the next point in the shape
#REPEAT(<>	Repeat information in <> number of times specified in ()
#RESET()	Force next occurrence of word(s) in () to be output
#TABLE()	Return value from table
#TLCHG	Trigger update of selected template words
#UPDATE()	Update word(s) in () as if they were output

Cycle Time Calculation

#CYTIME	Total cycle time so far
#TIME	Value to add to total cycle time
#TLTIME	Cycle time for previous tool

Feed Control

#FDMODE	Feed mode, primary or secondary
#FDMULT	Feed rate multiplier for Z depth moves
#FEED	Feed rate

Fixed Cycle, Lathe Groove (@FXD6; OD/ID groove) (@FXD7; face groove)

See .smf questions 200–204

#V1	Corner radius
#V2	Maximum width of cut
#XOV	Depth (Width for face groove)
#XPASS	Bottom (or X side for face groove)
#XST	Start corner X position
#ZOV	Width (Depth for face groove)
#ZPASS	Side (or Z bottom for face groove)
#ZST	Start corner Z position

Fixed Cycle, Lathe Peck Drill (@FXD5)

#FEED	Feed from the job operation setup
#PECK	Peck depth, set in .smf question 150 or updated with user command
#ZPASS	Bottom or end of peck cycle
#ZPOS	Origin of peck cycle

Fixed Cycle, Lathe Tapping (@FXD4)

#FEED	Feed from the job operation setup set for tap lead
#ZPASS	Bottom or end of tapping cycle
#ZPOS	Origin of tapping cycle

Fixed Cycle, Lathe Threading (@FXD1)

See `.smf` questions 185–195. The meanings of some thread template words depend upon the setting of `.smf` question 185.

Q. 185=0 For thread cycles (such as G33) where rapid moves must be included.

#FTHRD	Thread lead
#V0	Value of thread crest X dimension
#V1	Current depth of thread. Absolute value, updated for each pass
#V2	Thread depth remaining. Absolute value, updated for each pass
#V3	Incremental depth move to each pass depth for individual passes or to first pass depth for fixed cycle.
#V4	Number of passes at first pass depth that will fit.
#XCTR	Starting X value for LAST thread pass of cycle
#XOV	Change in X (delta X) between start and end of LAST thread pass
#XPASS	Ending X value for LAST thread pass of cycle
#ZCTR	Starting Z value for LAST thread pass of cycle
#ZOV	Change in Z (delta Z) between start and end of LAST thread pass
#ZPASS	Ending Z value for LAST thread pass of cycle
#FXD	Used to output the thread code from <code>.smf</code> question 186
#XPOS	Ending X value for each thread pass of cycle
#XST	Starting X value for each thread pass of cycle
#ZPOS	Ending Z value for each thread pass of cycle
#ZST	Starting Z value for each thread pass of cycle

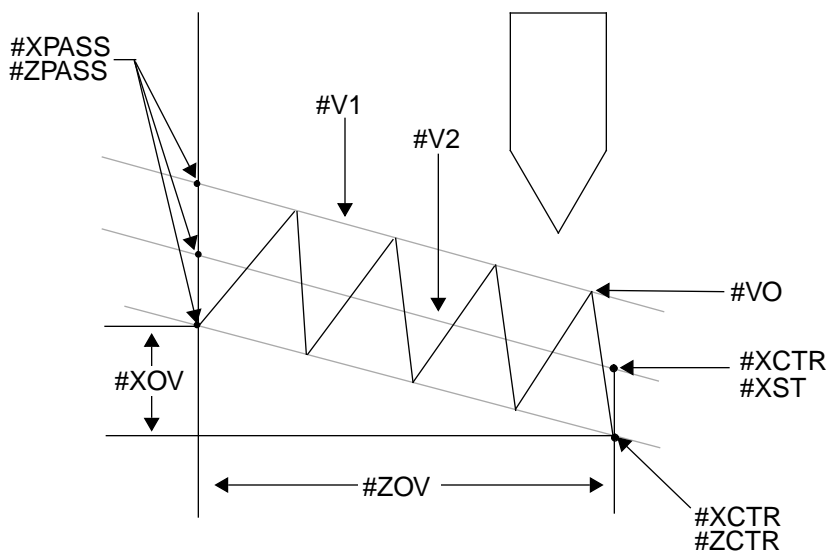


Figure 6-2
Use these
template words
for thread
(@FXD1) cycles.

Q.185=1 For thread cycles (such as G38) where each thread pass must be output, but where rapid moves do not need to be output with each pass.

#FTHRD	Thread lead
#FXD	Used to output the thread from .smf question 186
#VO	Value of thread crest X dimension
#V1	Current depth of thread. Absolute value, updated for each pass
#V2	Thread depth remaining. Absolute value, updated for each pass
#V3	Incremental depth move to each pass depth for individual passes or to first pass depth for fixed cycle.
#V4	Number of passes at first pass depth that will fit.
#XCTR	Starting X value for EACH thread pass of cycle
#XOV	Change in X (delta X) between start and end of LAST thread pass
#XPASS	Ending X value for EACH thread pass of cycle
#XST	Start X of thread. Actual value, updated for each pass
#ZCTR	NOT ACTIVE
#ZOV	Change in Z (delta Z) between start and end of LAST thread pass
#ZPASS	Ending Z value for EACH thread pass of cycle
#ZST	NOT ACTIVE

Q.185=2 For thread cycles (such as G76) where the entire thread cycle is generated based on a single set of parameters.

#V0	Value of thread crest X dimension
#V1	Absolute value of Depth of LAST thread pass
#V2	Absolute value of Depth of FIRST thread pass
#V3	Incremental depth move to each pass depth for individual passes or to first pass depth for fixed cycle.
#V4	Number of passes at first pass depth that will fit.
#FTHRD	Thread Lead
#FXD	Used to output the thread code from .smf question 185
#XCTR	Starting X value for LAST thread pass of cycle
#XOV	Change in X (delta X) between start and end of LAST thread pass
#XPASS	Ending X value for LAST thread pass of cycle
#XST	Start X of FIRST thread pass (includes rapid clear value)
#ZCTR	Starting Z value for LAST thread pass of cycle
#ZOV	Change in Z (delta Z) between start and end of LAST thread pass
#ZPASS	Ending Z value for LAST thread pass of cycle
#ZST	NOT ACTIVE

Fixed Cycle, Lathe Turning (@FXD2)

See .smf questions 180–184.

#XPASS	Ending X value of pass
#XPOS	Start Rapid Clear X position for first pass and any subsequent passes if desired
#ZPASS	Ending Z value of pass
#ZPOS	Start Rapid Clear Z position for first pass and any subsequent passes if desired

Lines

#LNANG	Line angle (in degrees)
#LNLEN	Line length
#OVANG	Offset vector angle
#XOV	Offset vectors

#XPOS	End position of line
#XST	Start position of line
#YOV	Offset vectors
#YPOS	End position of line
#YST	Start position of line
#ZOV	Offset vectors
#ZPOS	End position of line
#ZST	Start position of line

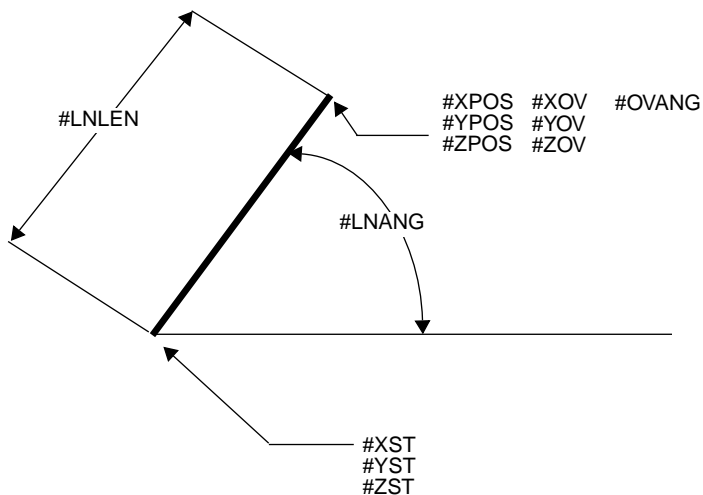



Figure 6-3
Use these
template words
for information
about lines.

Logic Functions

#AND	Logical AND
#CALL()	Calls a template section
#ELSE	Logical ELSE
#EXC	Exclude any following test for condition
#EXIT	Exit a template section
#IF()	Logical IF
#IFCHG()	Test for conditional output
#IFSTR()	Logical IF for string comparisons
#OR	Logical OR
#REPEAT()	Repeat template information specified number of times

Mathematical Functions

 The # sign is not required as a prefix

ABS	Returns the absolute value of the variable in parentheses
#EVAL()	Evaluates the expression in parentheses
INT	Returns the integer value of the variable in parentheses
SGN	Returns the sign of the variable in parentheses (1 or -1)
SQR	Returns the square root of the variable in parentheses

Spindle Control

#SPEED	Spindle speed
#SPMODE	Speed mode for RPM or CSS
#SPNDL	Spindle On direction

String Words

#BDSTR	String for block delete
#DATE	Current date
#ENAME	Name of current element
#FILE	Filename
#S0-#S19	User-assignable string variables
#TDESC	Tool description string from the job operation setup
#TLCMT	Tool comment string from the job operation setup
#TLID	Tool ID string from job operation setup
#TLTYP	Tool type number of active tool
#WKPLN	Name of tool plane

Tool Control

#ABSI	Absolute or incremental preparatory code
#FDADJ	Adjust feed for periphery of arcs
#MOV	Move preparatory code
#PUNCH	Punch ON/OFF code

Tool Information

#DCOMP	Cutter compensation code
#DOFF	Diameter offset register
#LOFF	Length offset register
#LTOOL	Last tool used
#NTOOL	Next tool used
#TDESC	Tool description string
#TLCHG	Control update of certain template variables
#TLCMT	Tool comment string
#TLDIA	Tool diameter (same as #TLWD)
#TLEN	Tool length
#TLID	Tool identification string
#TLOP	Tool operation number
#TLTIME	Cycle time for previous tool

#TLWD	Tool width
#TOFF	Combined tool and offset number
#TOOL	Current tool number
#XPOS	End position value of current element from the model
#XSET	Preset position for new tool based on the previous work plane
#YPOS	End position value of current element from the model
#YSET	Preset position for new tool based on the previous work plane
#ZPOS	End position value of current element from the model
#ZSET	Preset position for new tool based on the previous work plane

Trigonometric Functions

ATAN	Arc tangent
COS	Cosine
SIN	Sine
TAN	Tangent

User-Assigned Words

#C0-#C5	Switches (set in <code>.smf</code> questions 360–371)
#S0-#S19	String variables
#U0-#U19	Integer variables
#V0-#V19	Full double precision numeric variables

Work Coordinate System

#WKSCHG	Switch for new work plane and tool change
#XFO	New work plane's origin
#XHOME	First point of database
#XPSET	Absolute tool preset when #WKSCHG is 1
#XSET	Absolute tool preset
#YFO	New work plane's origin
#YHOME	First point of database
#YPSET	Absolute tool preset when #WKSCHG is 1
#YSET	Absolute tool preset
#ZFO	New work plane's origin
#ZHOME	First point of database
#ZPSET	Absolute tool preset when #WKSCHG is 1
#ZSET	Absolute tool preset

User-Definable Template Variables

All user-defined template variables (UDTVs) must be declared in the **@DECLARE** section of the template file. Although this section can appear anywhere in the `.tmp` file, it is usually best to make it the first section. Any blank lines, comments, and declaration statements in the declare section will be processed as UDTV information.

Supported variable types are decimal and integer. These variable types are indicated by the template words **#DEC** and **#INT**, respectively, and are declared using the following formats:

```
#DEC #dname
```

```
#INT #iname
```

where *dname* is the name of the decimal variable and *iname* is the name of the integer variable.

Each variable name can be up to 15 characters and must start with a letter. It can include upper- or lowercase letters A–Z, numerals 0–9, and the underscore character (`_`). Use of lowercase letters is recommended for clarity and better performance.

#DEC variables will use the default position format (set in `.smf` question 22) and **#INT** variables will use the default integer format (set in `.smf` question 9).

UDTVs are global in scope (available anywhere in the template file) and are initialized to 0 (zero) for each code run. Template-file syntax for UDTV is the same as that for system variables. Conditional behavior is also the same. The number of variables that can be used is limited only by available memory.

Testing and Verification

Introduction

This chapter explains how to test and verify components of a code generator. It focuses on specific syntax of the machine and template files and helps you write better documentation. The topics this chapter covers include:

- Working with template files
- Using literals
- Testing logic statements
- Solving problems

As you work with a code generator, you need to follow some basic guidelines. The following considerations are important for successful code generator testing:

- Knowledge of how SmartCAM generates code
- A logical methodology
- The ability to devise a test and correctly interpret the results
- Persistence

Creating Automatic Error Reports

When you set Report to Yes in the [Cd_err] section of the `smartcam.ini` file, SmartCAM will create an error file whenever it encounters any coding errors. By reading the error messages in this file and their corresponding definitions, you can easily determine where and why your code is not operating properly.

The error report file has the same name and path as the output code file, but it has an `.err` extension; for example:

Code file = `path\testpart`

Error file = `path\testpart.err`

If it detects an error, SmartCAM places a corresponding error message in the error file and displays a special notice, "Error report file has been generated," in the graphic view upon completion of the operation. The information that the error message contains is based on the following format:

Report#-Row, Col-Err#-Message

where:

Report# = sequential entry in the report file for cross-referencing to the code file.

Row, Col = row and column in the template file where the error was detected. A 0 indicates that it is not a `.tmp` file error.

Err# = number of the error.

Message = error message from the error message registration. See your application manual for information about specific error messages.

The following table identifies the lines in the [Cd_err] section of the `smartcam.ini` file:

Line	How to Use It
ErrorReference	Set to Yes to place a reference of the form <i>x</i> , where <i>x</i> is the report number or the instance the error was detected.
Conditionals	Set to Yes to output all conditionals and their status to the error file.
Sections	Set to Yes to output names of template sections that are called during code generation.
Report	Set to Yes to have SmartCAM create an error report whenever it generates code.

Isolating a Problem

If your code generator is working incorrectly, you need to find the file or files responsible for the errors. First, check your CNC Process Model to be sure it is constructed correctly. Code-related errors often result from wrong model values, not from the code generator. Use Show Path to verify the tool path graphically before generating code.

If the model is correct, check the setting of the `.smf` question that relates to the incorrect code. Remember that an `.smf` setting can affect other questions and the meaning of template words.

Modifying an .smf File

If you need to make changes to the machine file, use the Machine Define utility (see chapter 2). Make the necessary changes and save the file under the same name. (Make a backup copy of the file before you start, in case you need to restore the original file.)

Invalid Machine File Error

If you get an “Invalid machine file” error message when you try to generate code for a model, the code generator may have been written for a different machine type, an older version of the software, or both. Check the machine file with Edit Plus (see the [SmartCAM Edit Plus User Guide](#)) or another ASCII text editor. The first line identifies the version of the machine file, and the second line identifies which machine type the file was written for. You can use the Machine Define utility to convert a machine file to a different version (see chapter 2).

- ☞ Version numbers of machine files do not correspond with version numbers of SmartCAM applications.

Working with Template Files

Common Mistakes

Spelling and syntax errors are common mistakes that result in the .tmp file calling @ sections in ways you don't expect. Always check the @ sections in the .tmp file to make sure the spelling is correct. If SmartCAM tries to call a section that does not exist (or is spelled incorrectly), it skips that section without displaying a message.

- ☞ SmartCAM will report that the section is missing if you have it generate an error report (see page 7-1).

Adding Comments

Adding a double forward slash (//) to the line of a template section causes SmartCAM to ignore any information that follows on that line. This enables you to add comments that document changes or tests you're making. You can add the double slash anywhere in the line. SmartCAM begins processing information again at the start of the next line.

Using Literals

- ☞ You do not need to use literals if you have SmartCAM automatically create an error file during code generation (see page 7-1).

Marking Sections in Code

If the model file, .smf file, and spelling of .tmp sections are correct, the next step is to locate the .tmp section that is outputting incorrect code. You can assist your search by using literals to mark the NC code with the @ section SmartCAM uses. For example, if you place the literal (**line) just below the section name @LINE, as in the following example:

```
@LINE
(**line)
#MOV X#XPOS Y#YPOS Z#ZPOS F#FEED
```

SmartCAM will output the literal text to the NC code each time it accesses the @LINE section. The block of code would look like this:

```
(**line)
G01 X1.125 Y2.25 Z1.0 F20
```

The literal (**line) clearly shows where code output by the @LINE section begins. This helps to quickly isolate the problem area.

Turning Section Literals On and Off

You can use logic statements to toggle section markers off and on. Do this by adding a simple #IF statement to the beginning of all template sections, as in the following example:

```
@LINE
#IF(#U9=1)<**line>
#MOV X#XPOS Y#YPOS Z#ZPOS F#FEED
@ARC
#IF(#U9=1)<**arc>
#MOV X#XPOS Y#YPOS Z#ZPOS I#XCTR J#YCTR K#ZCTR F#FEED
```

These logic statements check to see if the variable #U9 equals 1 (#U9=1). If #U9 equals 1, then each statement outputs the appropriate section marker. This marks the sections in code and helps you identify where problems in code originate.

Once you locate the section producing incorrect code, find out why this section is not acting as expected. This may require creating a test to find out which sections SmartCAM calls and when. Once you correct the section causing the problem, turn off the markers by setting #U9=0.

The following example is a simple test to see how a .tmp file is functioning. Note that the contents of the output code file do not match the test template file line for line.

Test Template File	Output Code File
@START	*****START
%	N100 G49 G40 G90 G17
#EVAL(#U9=1)	N105 G00 X2.0 Y0.9875 T3 M06
#IF(#U9=1)< ***** START>	N110 ***** RAPID


```

#ONBLK G49 G40 G90 G17          N115 Z0.5
#MOV X#XPOS Y#YPOS T#TOOL M06  N120 ***** LINE
                                N125 G41 D33 G01 Y0.9375 Z-0.375 F12.0
@TOOLCHG                        N130 ***** LINE
#IF(#U9=1)                      N135 X3.6386
< ***** TOOLCHG>            N140 ***** ARC
#MOV X#XPOS Y#YPOS T#TOOL M06  N145 G03 X3.8005 Y1.0303 I0.0 J0.1875
                                N150 ***** LINE
@END                              N155 G01 X5.0436 Y3.154
#IF(#U9=1)< ***** END>      N160 ***** ARC
M30                              N165 G03 X4.666 Y3.8125
#OFFBLK%                         I-0.3776 J0.221
                                N170 ***** END
                                N175 M30
                                %

@RAP
#IF(#U9=1)< ***** RAPID>
< #ABSI>< #FXD>< #MOV>< X#XPOS>< Y#YPOS>< Z#ZPOS>
@LINE
#IF(#U9 = 1)< ***** LINE>
< #DCOMP#EXC D#DOFF>< #MOV>< X#XPOS>< Y#YPOS><
Z#ZPOS>< F#FEED>
@ARC#IF(#U9 = 1)< ***** ARC>
< #MOV>< X#XPOS>< Y#YPOS>< I#XCTR>< J#YCTR>< F#FEED>
@

```

Using Literals within a Section

If you are not sure where each block of code comes from, you can place a literal in front of each line in the template file to mark the output code. (This is similar to using section markers, as described earlier in this chapter.) The following example shows how this might look in a file.

```

@TOOLCHG
a < #FXD> M09
b #MOV G91 G28 Z0
c G90 X#XPOS Y#YPOS

```

```
d G49 T#TOOL M06
e #SPNDL S#SPEED F#FEED
f G43 H#LOFF Z#ZPOS M08
```

A sample of output follows:

```
a G80 M09
b G00 G91 G28 Z0
c G90 X3.500 Y7.500
d G49 T3 M06
e M03 S1000 F10
f G43 H3 Z.5 M08
```

☞ Use lowercase letters, because they are easier to see in the code.

These literals in the NC code printout show where every line of code comes from. In cases where you have narrowed the problem down to a particular template section, this enables you to find the faulty template file line.

Testing Logic Statements

You can also insert literals to determine whether `#IF` or `#CALL` statements are working correctly. Use lowercase letters to mark processing within a line, as shown in the following example:

```
@ARC
***arc
#IF(#TOOL>20)<#CALL(ARCALC)a>
#MOV X#XPOS Y#YPOS Z#ZPOS I#XCTR J#YCTR K#ZCTR F#FEED
```

In this example, if (a) displays in a separate block, you know `#TOOL` is for a tool size bigger than 20, and the `#IF` statement is executed.

Make sure any test you devise does not interfere with the way the template file functions. For example, if you output a template word such as `#XPOS` to check its value, SmartCAM updates the word and then skips to the next coordinate in the database. If that same template word is then used in a calculation or is output elsewhere in the template section, the result may be incorrect, showing information derived from the next coordinate.

To avoid this, set the variable as user assigned and output the value of the user-assigned variable, as shown in the following example:

```
@LINE
#EVAL(#V4=#XPOS)
**xpos=#V4
#EVAL(#V4=#V4*3.14159^2)
#MOV X#V4 Y#YPOS
#UPDATE(#XPOS)
```

SmartCAM outputs the results of that variable but does not update the template word (`#XPOS`) before it is used in another calculation. This is accomplished by the last line of the example.

Common Code Generator Problems and Solutions

Problem: *The arc center/end position is not rounding correctly.*

Cause: The format for this information is not set high enough in the machine file.

Solution: Change the numeric format in the machine file (for example, change the format from D2.3 to D2.7).

Problem: *There are blank lines in the code.*

Causes:

1. There are spaces at the end of one or more lines in the template file.
2. There are blank spaces between conditional template words.

Solutions:

1. You can use Edit Plus to find spaces at the end of a line. Open the template file in Edit Plus and press the **END** key on the keyboard to move to the end of a line. If the cursor goes past the last visible character in any line, move the cursor to the right of the last visible character and press **F4** to remove the spaces.
2. Remove spaces between conditional template words. For example, change

```
<#MOV> <#XPOS> <Y#YPOS>  
to
```

```
<#MOV><#XPOS><Y#YPOS>
```

The spaces will cause blank lines if there is no output from the conditionals.

Problem: *The code generator is calling the wrong section.*

Cause: A template section with a similar but longer name may be listed ahead of the intended section. For example, if @ENDPROF is listed before @END, when the system searches for the @END section, it will execute @ENDPROF instead.

Solution: When a section name is identical to the first part of a longer section name, place the shorter section name ahead of the other one in the template file. For example, place @END before @ENDPROF.

Problem: *Data does not change.*

Causes:

1. The correct template word is not used or is misspelled.
2. Machine file question 5 is set so that the template word is updated but never output.

Solutions:

1. Check the definition and spelling of the template word.
2. Check the setting of question 5.

Problem: The error message “invalid machine file” displays during code generation.

- Causes:**
1. You are trying to use a nonsupported version of a machine file.
 2. You are trying to use a machine file for a different machine type.
 3. Your .smf file has been corrupted.

- Solutions:**
1. Convert the machine file to the correct version (see page 2-2).
 2. Use a different machine file.
 3. Replace the corrupted file with a backup copy.

Problem: Inch/metric conversion is not working for some template words.

- Causes:**
1. #FMT() in the template file prevents conversion of the template word.
 2. #V0 through #V9 use the format set in the machine file for positioning moves.

- Solutions:**
1. Do not use #FMT().
 2. Use #FMT() to specify the format and avoid automatic conversion.

Problem: Lathe tool change is not occurring.

Cause: Lathe requires a tool-change point for the @TOOLCHG section to be called.

Solution: Make sure you have a tool-change point for all tools in your model.

Problem: A logic function is not working.

- Causes:**
1. The logic function has incorrect syntax.
 2. A line in the template file is corrupted.

- Solutions:**
1. Check your logic syntax.
 2. Run a test to make sure the line is being called. As a last resort, delete and retype the entire line in the template file. Sometimes characters you cannot see in Edit Plus are actually present in the file. These blind characters will sometimes cause the logic function to fail. (You may be able to see these characters in another text editor.)

Problem: Metric code is needed, but code output is inches.

Cause: Question 20 in the machine file is set incorrectly. Whether or not data will be converted depends upon the input units specified in the job operations setup.

Solution: Set .smf question 20 to 2.

Problem: No data is output for a template word.

- Causes:**
1. A numeric format for the word is not assigned in the machine file.
 2. No data for the word is assigned in the model, the job operations setup, or the machine file.

- Solutions:**
1. Assign an appropriate numeric format in the machine file.
 2. Check the model and the .jof and .smf files.

Problem: *There are spaces within blocks of code.*

Cause: Spaces in the template file act like literals.

Solution: Remove all spaces in the template file. For example, change < X #XPOS> to <X#XPOS>.

Problem: *The speed table is giving wrong information.*

- Causes:**
1. You have not set up the table properly.
 2. You are trying to use machine file question 52 and the template word #TABLE. This causes the system to convert twice.

- Solutions:**
1. See #TABLE or machine file question 52.
 2. Use question 52 if your table is called @SPEEDS. Otherwise, use #TABLE. Don't use both.

Problem: *A template section is not being executed.*

- Causes:**
1. The template section name is not spelled correctly.
 2. The system is not calling that section.
 3. For custom sections, the #CALL or user command format is incorrect.

- Solutions:**
1. Check the spelling of the template section name. SmartCAM will skip over nonexistent (or misspelled) sections with no error message.
 2. Check the manual and build a simple model test file to help you understand which sections are supposed to be called.
 3. Check the format of your #CALL or user command sections.

Problem: *The value 32000 or 100000 is output for a template word.*

Cause: SmartCAM could not find a value for the template word.

Solution: Check your model file, job operations setup, and math assignments in the template file related to the template word.

Index

character 1-9, 5-4
% (tape rewind) character 5-7
() (parentheses) 5-11
, (comma) 5-9
// (double slashes) 5-9
 in template files 7-3
<> (brackets) 1-6, 5-7
@ character 1-6, 5-1
\ (backslash) 5-7

#FXD 6-10, 6-41, 6-42, 6-43
 #IF() 5-8, 6-11, 6-45
 #IFCHG() 6-12, 6-45
 #IFSTR() 5-6, 6-12, 6-45
 #INC 6-13
 #INC2 6-13
 #INC3 6-13
 #INCLUDE() 3-4, 6-13, 6-39
 #INDXA 3-13, 3-14, 6-13, 6-37
 #INDXB 3-13, 3-14, 6-13
 #INDXC 3-13, 3-14, 6-14
 #INO 3-13, 3-14, 6-14
 #INT 6-14, 6-48
 #IPECK 6-14
 #JNO 3-13, 3-14, 6-14
 #KNO 3-13, 3-14, 6-14
 #LEVEL 6-14
 #LNANG 6-14, 6-44
 #LNLEN 6-14, 6-37, 6-44
 #LOFF 6-15, 6-46
 #LTOOL 6-15, 6-46
 #MOV 5-5, 6-15, 6-37, 6-46
 #NBLMD 6-15
 #NEXTPT 6-16, 6-39
 #NHOL 6-16
 #NHOL2 6-16
 #NTOOL 6-16, 6-46
 #OFFBLK 6-16, 6-38
 #ONBLK 6-17, 6-38
 #OPTYPE 6-17
 #OR 5-7, 6-18, 6-45
 #OVANG 6-18, 6-37, 6-44
 #PCAN 6-18
 #PECK 6-18, 6-40
 #PLANE 6-19, 6-37
 #PRMRY 6-19
 #PSETLEN 6-19
 #PTOP 6-19
 #PUNCH 6-19, 6-46
 #QANG 6-19
 #RAD2 6-20
 #RANG 6-20
 #REPEAT 3-4
 #REPEAT() 6-20, 6-39, 6-45
 #REPO 6-20
 #RESET() 6-21, 6-39
 #RFEED 6-21
 #ROT1 3-4, 6-21
 #ROT2 3-4, 6-21
 #RTNLVL 6-21
 #S0-#S19 5-6, 6-22, 6-46, 6-47
 #SAFBLK 6-22, 6-38
 #SBLK 3-4, 6-22
 #SBTYP 3-4, 6-22
 #SLDWN 6-22
 #SNAME 3-4, 6-23
 #SPEED 6-23, 6-45
 #SPMODE 6-23, 6-45
 #SPNDL 6-23, 6-45
 #SPOFF 6-23
 #SREPT 3-4, 6-24
 #STA 3-13, 6-24
 #STANG 6-24, 6-37
 #STB 3-13, 6-24
 #STC 3-13, 6-24
 #SYNCH 6-24
 #SYSTEME 6-24
 #TABLE() 6-24, 6-39
 #TANG 6-25, 6-37
 #TCODE 6-25
 #TDESC 6-25, 6-46
 #THDPTCH 6-25
 #TIME 6-26, 6-39
 #TINDX 3-13, 3-14, 6-26
 #TLCHG 6-26, 6-39, 6-46
 #TLCMT 6-27, 6-46
 #TLDIA 6-27, 6-46
 #TLEN 6-27, 6-46
 #TLGRAPHIC 6-27
 #TLID 6-27, 6-46
 #TLOP 6-28, 6-46
 #TLPAGEID 6-29

#TLSTA 6-29
 #TLTIME 6-29, 6-39, 6-46
 #TLTYP 6-46
 #TLTYPE 6-29
 #TLWD 6-30, 6-47
 #TOFF 6-30, 6-47
 #TOOL 6-30, 6-47
 #U0-#U19 5-6, 6-31, 6-47
 #UOV 6-31
 #UPDATE() 6-31, 6-39
 #V0 6-41, 6-42, 6-43
 #V0-#V19 5-6, 6-32, 6-47
 #V1 6-40, 6-41, 6-42, 6-43
 #V2 6-40, 6-41, 6-42, 6-43
 #V3 6-41, 6-42, 6-43
 #V4 6-41, 6-42, 6-43
 #VOV 6-31
 #WINCL 6-32
 #WKPLN 3-13, 3-14, 6-32, 6-46
 #WKSCHG 3-13, 3-14, 6-33, 6-48
 #WOV 6-31
 #WRAD 6-33
 #XCTR 3-4, 6-33, 6-37, 6-41, 6-42, 6-43
 #XFO 3-12, 3-14, 6-34, 6-48
 #XHOME 3-14, 6-34, 6-48
 #XOV 6-34, 6-37, 6-40, 6-41, 6-42, 6-43, 6-44
 #XPASS 6-35, 6-40, 6-41, 6-42, 6-43
 #XPOFF 6-35
 #XPOS 6-35, 6-37, 6-41, 6-43, 6-44, 6-47
 #XPSET 3-12, 3-14, 6-36, 6-48
 #XPST 3-14
 #XSET 3-13, 3-14, 6-36, 6-47, 6-48
 #XST 3-12, 6-36, 6-37, 6-40, 6-41, 6-42, 6-43,
 6-44
 #YCTR 3-4, 6-33, 6-37
 #YFO 3-12, 3-14, 6-34, 6-48
 #YHOME 3-14, 6-34, 6-48
 #YOV 6-34, 6-37, 6-44
 #YPASS 6-35
 #YPOFF 6-35
 #YPOS 6-35, 6-37, 6-44, 6-47
 #YPSET 3-12, 3-14, 6-36, 6-48
 #YPST 3-14
 #YSET 3-13, 3-14, 6-36, 6-47, 6-48
 #YST 3-12, 6-36, 6-37, 6-44
 #ZCHK 6-36
 #ZCTR 3-4, 6-33, 6-37, 6-41, 6-42, 6-43
 #ZDPTH 6-36
 #ZFO 3-12, 3-14, 6-34, 6-48
 #ZHOME 3-14, 6-34, 6-48
 #ZOV 6-34, 6-37, 6-40, 6-41, 6-43, 6-44
 #ZPASS 6-35, 6-40, 6-41, 6-43, 6-44
 #ZPOFF 6-35
 #ZPOS 6-35, 6-37, 6-40, 6-41, 6-44, 6-47
 #ZPSET 3-12, 3-14, 6-36, 6-48
 #ZPST 3-14
 #ZSET 3-13, 3-14, 6-36, 6-47, 6-48
 #ZST 3-12, 6-36, 6-37, 6-40, 6-41, 6-43, 6-44
 @ARC 5-12
 @ATTCHMT 5-13
 @CORNER 5-13
 @CYCLCHG 5-13
 @DECLARE 5-13, 6-48
 @END 5-14
 @ENDDEF 3-3, 5-21
 @ENDPROF 5-14
 @FXD1-@FXD7 3-3, 5-14
 @FXDDEF 3-3, 5-21
 @GOSUB 3-4, 5-21
 @HELIX 5-15
 @LINE 5-15
 @OP_BORE 5-16
 @OP_CBORE 5-16
 @OP_CDRL 5-16
 @OP_CSINK 5-16
 @OP_DRL 5-16
 @OP_PDRL 5-16
 @OP_REAM 5-17, 5-20
 @OP_SPDRL 5-17
 @OP_SPEC 5-17

- @OP_SPFACE 5-17
 - @OP_TAP 5-17
 - @PNCHTL 5-18
 - @RAP 3-8, 3-9, 5-18
 - @REPO 5-22
 - @SHPRF 5-18
 - @SPEEDS 5-18
 - @START 3-13, 3-14, 5-18
 - @STEPCHG 5-19
 - @STOP 5-22
 - @STPROF 5-19
 - @SUBDEF 3-4, 5-21
 - @TOOLCHG 3-7, 3-8, 3-9, 3-13, 3-14, 5-19
 - @TORCH 5-19
 - @TPINDX 3-7, 3-8, 3-9, 3-13, 3-14, 5-19
 - @TRAVERSE 5-20
 - @WAIT 5-20
 - @WKSYS 3-13, 5-20
 - @XZARC 5-20
 - @YZARC 5-20
 - @ZCHKMV 5-21
 - @ZCLRMV 5-21
 - @ZDPTHMV 5-21
 - 3-D arcs, machine file questions for 4-47–4-50
 - 4- and 5-axis positioning, machine file questions for 4-50–4-54
 - 4-axis, machine file questions for
 - lathe 4-37–4-39
 - wire EDM 4-46–4-47
 - 4th-axis positioning 3-9
 - machine file questions for 3-11
 - 5th-axis dual rotary positioning 3-10
- A**
- A-axis rotation 3-11
 - ABS 5-10, 6-45
 - #ABSI 6-1, 6-46
 - action buttons (Machine Define)
 - All Categories 2-12
 - Next 2-4
 - Previous 2-4
 - #ADIR 3-13, 6-2
 - Advanced 3-D Machining 3-10
 - #AITOOL 6-2
 - All Categories action button 2-12
 - #AND 5-8, 6-2, 6-45
 - applications, location of files 1-4
 - #ARAD 6-2, 6-37
 - @ARC 5-12
 - arcs
 - machine file questions for
 - 3-D 4-47–4-50
 - circular profile moves 4-17–4-20
 - arcs, template words for 6-37
 - ASCII text editor 1-7
 - ATAN 5-11, 6-47
 - @ATTCHMT 5-13
 - axis orientation, machine file questions for 4-5–4-8
 - axis rotation 3-11
- B**
- backslash (\) 5-7
 - B-axis rotation 3-11
 - #BDIR 3-13, 6-2
 - #BDSTR 6-3, 6-38, 6-46
 - #BLK 6-3, 6-38
 - #BLKDEL 6-3, 6-38
 - block number control
 - machine file questions for 4-4–4-5
 - template words for 6-38
 - block numbering 5-4
 - brackets (<>) 1-6, 5-7
 - buttons see action buttons
 - buttons, resizing (Machine Define) 2-4
- C**
- #C0–#C5 6-3, 6-47
 - CAD systems, converting information from 1-8
 - #CALL 5-3
 - #CALL() 6-3, 6-39, 6-45

- CAM Connection 1-8
- categories
 - see machine file questions *and* template word categories
 - selecting in Machine Define 2-11
- Category List (Machine Define) 2-11
- C-axis rotation 3-11
- [Cd_err] section of `smartcam.ini` file 7-1
- #CDIR 3-13, 6-2
- #CENG 6-4
- changing the printer setup 2-9
- characters
 - # 1-9, 5-4
 - % (tape rewind) 5-7
 - () (parentheses) 5-11
 - , (comma) 5-9
 - // (double slashes) 5-9
 - <> (brackets) 1-6, 5-7
 - @ 1-6, 5-1
 - \ (backslash) 5-7
 - space 5-7
- #CHKD 6-4
- circular arc profile moves, machine file questions for 4-17–4-20
- #CLEAR 6-4
- closing Machine Define 2-10
- code filtering for mesh polylines 3-6
- code formats 1-14
- code generation
 - and SmartCAM 1-7–1-14
 - files required for 1-3
- code generators
 - getting from CAMAX Manufacturing Technologies 1-1
 - modifying 1-14–1-21
- comma (,) 5-9
- command buttons see action buttons
- commands, user 5-3
 - machine file questions for 4-56–4-57
- comments
 - adding to template sections 5-9
 - in template files 7-3
- Communicate 1-7
- compensation (cutter), machine file questions for 4-21–4-25
- condition test operators 5-8
- conditional brackets (<>) 1-6, 5-7
- Conditionals (line in `smartcam.ini` file) 7-2
- contour machining, rotary 3-10
- contouring, machine file questions for
 - profile 4-20–4-21
 - rotary 4-60–4-65
- control functions, template words for 6-39
- control template words 5-7
- control, logic 5-8
- control-menu box (Machine Define) 2-4
- conversion, linear equivalent feed rate 3-15–3-17
- converting
 - information from CAD systems 1-8
- #COOLNT 6-4
- coordinate system (work), template words for 6-48
- @CORNER 5-13
- COS 5-11, 6-47
- #CSSDIA 6-5
- #CSSRAD 6-5
- #CTRLOC 6-5, 6-37
- #CUTC 6-5
- cutter compensation, machine file questions for 4-21–4-25
- @CYCLCHG 5-13
- #CYCLE 6-5
- cycle time
 - calculation, template words for 6-39
 - machine file questions for 4-25–4-26
- cycles, machine file questions for
 - lathe 4-31
 - lathe groove 4-36–4-37

cycles, machine file questions for (*cont'd*)
 mill 4-28–4-31
 mill peck drill 4-26–4-27
 punch 4-40–4-45
 #CYTIME 6-5, 6-39

D

database, process model 1-8
 #DATE 6-6, 6-46
 #DCOMP 6-6, 6-46
 #DEC 6-6, 6-48
 @DECLARE 5-13, 6-48
 default printer
 changing settings for 2-9
 selecting 2-9
 dialog boxes (Machine Define)
 Open SMF 2-6
 Print 2-9
 Print Setup 2-9
 Save As 2-8
 Select Categories 2-11
 direction of axis rotation 3-11
 directories
 for machine (.smf) files 1-5
 for process model (.pm4) files 1-4
 #DOFF 6-6, 6-46
 double slashes (//)
 in template files 7-3
 double-precision template words 5-6
 drill cycles (peck), machine file questions
 for 4-26–4-27
 drill subroutines 3-5
 dual rotary positioning, 5th-axis 3-10
 #DWELL 6-6
 dwell, machine file questions for 4-9–4-11

E

#EANG 6-7, 6-37
 #EBLK 3-4, 6-7
 Edit Plus 1-7

editing machine files 2-14
 EDM (4-axis wire), machine file questions
 for 4-46–4-47
 EDM machine files 3-1
 Element Data 1-8
 elements of Machine Define window 2-3
 #ELSE 5-8, 6-7, 6-45
 #ENAME 6-7, 6-46
 @END 5-14
 @ENDDEF 3-3, 5-21
 @ENDPROF 5-14
 error messages in Machine Define 2-16
 error reports, creating 7-1
 ErrorReference (line in smartcam.ini file) 7-2
 errors
 creating error reports 7-1
 in code generation 7-7–7-9
 in template files 7-3
 Invalid Machine File 7-3
 #EVAL() 5-9, 6-8, 6-39, 6-45
 examples of numeric formats 3-3
 #EXC 5-7, 6-8, 6-45
 #EXIT 5-9, 6-8, 6-45
 Exit 2-10
 exiting Machine Define 2-10
 #EXLN 6-8, 6-12, 6-39
 explanation box (Machine Define) 2-5

F

#FDADJ 6-8, 6-37, 6-46
 #FDMODE 6-9, 6-39
 #FDMULT 6-9, 6-39
 #FEED 6-9, 6-39, 6-40
 feed control, template words for 6-39
 feed rate conversion 3-15–3-17
 feeds, machine file questions for 4-9–4-11
 fifth-axis dual rotary positioning 3-10
 #FILE 6-9, 6-46
 File Manager, starting Machine Define from 2-3

- File menu (Machine Define) 2-6–2-10
 - Exit 2-10
 - Open SMF 2-6
 - Print 2-9
 - Print Setup 2-9
 - Save 2-7
 - Save As SMF V4 2-7
 - Save As SMF V5 2-7
 - file versions 2-2, 7-3
 - files
 - required for code generation 1-3
 - test 1-16
 - see also* job operations setup files, machine files, process model files, *and* template files
 - filtering code for mesh polylines 3-6
 - finding specific questions in a machine file 2-13
 - fixed cycles, template words for
 - lathe groove 6-40
 - lathe peck drill 6-40
 - lathe tapping 6-40
 - lathe threading 6-41
 - lathe turning 6-43
 - #FMT() 6-10, 6-39
 - formats
 - for code 1-14
 - numeric 3-2–3-3
 - examples 3-3
 - forward slashes
 - (//) in template files 7-3
 - four- and five-axis positioning, machine file questions for 4-50–4-54
 - four-axis, machine file questions for
 - lathe 4-37–4-39
 - wire EDM 4-46–4-47
 - fourth-axis positioning 3-9
 - machine file questions for 3-11
 - #FTHRD 6-10, 6-41, 6-42, 6-43
 - full-numeric template words 5-6
 - functions
 - mathematical and trigonometric 5-9–5-11
 - operators 5-11
 - order of operations 5-11
 - #FXD 6-10, 6-41, 6-42, 6-43
 - @FXD1–@FXD7 3-3, 5-14
 - @FXDDEF 3-3, 5-21
- ## G
- @GOSUB 3-4, 5-21
 - Goto Question 2-13
 - groove cycle (lathe), machine file questions for 4-36–4-37
- ## H
- @HELIX 5-15
 - helixes, machine file questions for 4-47–4-50
 - Help menu (Machine Define) 2-4
- ## I
- icon for Machine Define 2-3
 - #IF() 5-8, 6-11, 6-45
 - #IFCHG() 6-12, 6-45
 - #IFSTR() 5-6, 6-12, 6-45
 - #INC 6-13
 - #INC2 6-13
 - #INC3 6-13
 - #INCLUDE() 6-13, 6-39, 3-4
 - #INDXA 3-13, 3-14, 6-13, 6-37
 - #INDXB 3-13, 3-14, 6-13
 - #INDXC 3-13, 3-14, 6-14
 - #INO 3-13, 3-14, 6-14
 - installing Machine Define 2-2
 - #INT 6-14, 6-48
 - INT 5-10, 6-45
 - integer template words 5-6
 - Invalid Machine File error 7-3
 - #IPECK 6-14
 - Item List (Machine Define) 2-11

J

#JNO 3-13, 3-14, 6-14
 job operations setup (.jof) files 1-13
 location of 1-5
 .jof files *see* job operations setup files

K

keyboard, using in Machine Define 2-15
 keystrokes in Machine Define 2-15
 #KNO 3-13, 3-14, 6-14

L

lathe fixed cycles, template words for
 groove 6-40
 peck drill 6-40
 tapping 6-40
 threading 6-41
 turning 6-43
 lathe machine files 3-1
 lathe, machine file questions for
 4-axis 4-37–4-39
 cycles 4-31
 groove cycles 4-36–4-37
 threading 4-31–4-36
 leaving Machine Define 2-10
 #LEVEL 6-14
 levels of modification 1-2
 @LINE 5-15
 linear equivalent feed rate conversion 3-15–3-17
 linear processing, defined 1-7
 linear profile moves, machine file questions
 for 4-16
 lines, template words for 6-44
 lists (Machine Define)
 Category List 2-11
 Item List 2-11
 literals
 special characters 5-7
 using in template sections 5-7
 literals, using in template files 7-3–7-6

#LNANG 6-14, 6-44
 #LNLEN 6-14, 6-37, 6-44
 location of files
 job operations setup (.jof) files 1-5
 machine (.smf) files 1-5
 process model (.pm4) files 1-4
 template (.tmp) files 1-6
 #LOFF 6-15, 6-46
 logic control 5-8
 logic functions, template words for 6-45
 #LTOOL 6-15, 6-46

M

machine (.smf) files 1-11, 2-1, 3-1–3-17
 components of 2-4
 editing 2-14
 going to a specific question in 2-13
 location of 1-5
 opening 2-6
 printing 2-9
 searching for text in 2-12
 selecting categories for viewing in Machine
 Define 2-11
 types of 3-1
 versions 2-2, 7-3
 machine communications 1-7
 Machine Define 2-1–2-16
 buttons *see* action buttons
 control-menu box 2-4
 editing files in 2-14
 error messages 2-16
 exiting 2-10
 files for 2-1
 icon for 2-3
 installing 2-2
 menu bar 2-4
 scroll bar 2-4
 see also menus
 selecting categories in 2-11
 starting 2-2–2-3

- Machine Define (*cont'd*)
 - using the keyboard 2-15
 - window elements 2-3
 - machine file questions
 - 3-D arcs and helices 4-47–4-50
 - 4- and 5-axis positioning 3-11, 4-50–4-54
 - 4-axis wire EDM 4-46–4-47
 - block number control 4-4–4-5
 - circular arc profile moves 4-17–4-20
 - cutter compensation 4-21–4-25
 - cycle time 4-25–4-26
 - feeds, speeds, and dwell 4-9–4-11
 - general 4-2–4-4
 - lathe 4-axis 4-37–4-39
 - lathe cycles 4-31
 - lathe groove cycle 4-36–4-37
 - lathe threading 4-31–4-36
 - linear profile moves 4-16
 - mill and lathe peck drill cycle 4-26–4-27
 - mill cycles 4-28–4-31
 - profile contouring 4-20–4-21
 - punch cycles 4-40–4-45
 - rapid positioning 4-15–4-16
 - rotary contouring 3-12, 4-60–4-65
 - subprogramming 4-54–4-55
 - Tape-to-Shape 4-57–4-59
 - tool change 4-12–4-15
 - units and axis orientation 4-5–4-8
 - user commands 4-56–4-57
 - machine file reference 4-1–4-65
 - machining, multiaxis and rotary, template variables for 3-12
 - mathematical functions 5-9–5-11
 - operators 5-10
 - order of operations 5-11
 - template words for 6-45
 - see also* trigonometric functions
 - menu bar (Machine Define), location of 2-4
 - menus (Machine Define)
 - File 2-6–2-10
 - Help 2-4
 - Search 2-12–2-14
 - View 2-10–2-12
 - mesh polylines, code filtering for 3-6
 - Microsoft Windows
 - starting Machine Define from 2-2
 - window elements 2-3
 - mill cycles, machine file questions for 4-28–4-31
 - peck drill 4-26–4-27
 - mill machine files 3-1
 - missing template sections 7-3
 - modification, levels of 1-2
 - modifying code generators 1-14–1-21
 - #MOV 5-5, 6-15, 6-37, 6-46
 - moves, machine file questions for
 - circular arc profile 4-17–4-20
 - linear profile 4-16
 - multiaxis machining
 - template variables for 3-12
- N**
- #NBLMD 6-15
 - Next action button 2-4
 - #NEXTPT 6-16, 6-39
 - #NHOL 6-16
 - #NHOL2 6-16
 - #NTOOL 6-16, 6-46
 - numbering, block 5-4
 - numeric formats 3-2–3-3
 - examples 3-3
 - numeric template words 5-6
- O**
- #OFFBLK 6-16, 6-38
 - #ONBLK 6-17, 6-38
 - @OP_BORE 5-16

- @OP_CBORE 5-16
 - @OP_CDRL 5-16
 - @OP_CSINK 5-16
 - @OP_DRL 5-16
 - @OP_PDRL 5-16
 - @OP_REAM 5-17, 5-20
 - @OP_SPDRL 5-17
 - @OP_SPEC 5-17
 - @OP_SPFACE 5-17
 - @OP_TAP 5-17
 - Open SMF 2-6
 - operations (math), order of 5-11
 - operators, condition test 5-8
 - #OPTYPE 6-17
 - #OR 5-7, 6-18, 6-45
 - orientation of printer paper 2-10
 - #OVANG 6-18, 6-37, 6-44
- P**
- paper (printer)
 - orientation 2-10
 - size 2-10
 - source 2-10
 - parentheses, using 5-11
 - path, tool 1-9
 - #PCAN 6-18
 - #PECK 6-18, 6-40
 - peck drill cycles, machine file questions
 - for 4-26–4-27
 - #PLANE 6-19, 6-37
 - .pm4 files *see* process model files
 - @PNCHTL 5-18
 - polylines (mesh), code filtering for 3-6
 - positioning
 - 4- and 5-axis, machine file questions for 4-50–4-54
 - dual 5th-axis rotary 3-10
 - rapid, machine file questions for 4-15–4-16
 - postprocessing 1-2
 - Previous action button 2-4
 - Print (Machine Define) 2-9
 - Print Setup 2-9
 - printer setup, changing 2-9
 - printers
 - changing settings for 2-9
 - selecting a default printer 2-9
 - printing a machine file 2-9
 - #PRMRY 6-19
 - problems
 - in code generation 7-7–7-9
 - in Machine Define 2-16
 - isolating 7-2
 - process model (.pm4) files, location of 1-4
 - profile contouring, machine file questions
 - for 4-20–4-21
 - profile moves, machine file questions for
 - circular arc 4-17–4-20
 - linear 4-16
 - Program Manager, starting Machine Define
 - from 2-2
 - #PSETLEN 6-19
 - #PTOP 6-19
 - #PUNCH 6-19, 6-46
 - punch cycles, machine file questions for 4-40–4-45
 - punch machine files 3-1
- Q**
- #QANG 6-19
 - question list box (Machine Define) 2-5
 - questions, machine file *see* machine file questions
- R**
- #RAD2 6-20
 - #RANG 6-20
 - @RAP 3-8, 3-9, 5-18
 - rapid positioning, machine file questions
 - for 4-15–4-16
 - #REPEAT 3-4

#REPEAT() 6-20, 6-39, 6-45
#REPO 6-20
@REPO 5-22
 Report (line in `smartcam.ini` file) 7-2
#RESET() 6-21, 6-39
 resizing buttons (Machine Define) 2-4
#RFEED 6-21
#ROT1 3-4, 6-21
#ROT2 3-4, 6-21
 rotary contouring, machine file questions
 for 4-60–4-65
 rotary machining 3-9–3-14
 machine file questions for 3-12
 rotation direction 3-11
 template variables for 3-12
#RTNLVL 6-21

S

#S0–#S19 5-6, 6-22, 6-46, 6-47
#SAFBLK 6-22, 6-38
 Save 2-7
 Save As dialog box (Machine Define) 2-8
 Save As SMF V4 2-7
 Save As SMF V5 2-7
#SBLK 3-4, 6-22
#SBTYP 3-4, 6-22
 scroll bar, location of (Machine Define) 2-4
 Search menu (Machine Define) 2-12–2-14
 Goto Question 2-13
 Text 2-12
 searching for text in a machine file 2-12
 Sections (line in `smartcam.ini` file) 7-2
 sections, template 1-6
 Select Categories 2-11, 2-12
 selection box (Machine Define) 2-5
 settings, printer 2-9
 setup, printer 2-9
SGN 5-10, 6-45
 Show Path 1-8
@SHPRF 5-18

SIN 5-11, 6-47
 size of printer paper 2-10
 slashes
 backslash 5-7
 double 5-9
#SLDWN 6-22
 SmartCAM
 Advanced 3-D Machining 3-10
 code generation in 1-7–1-14
 Communicate 1-7
 Edit Plus 1-7
 location of application files 1-4
 `smartcam.ini` file 7-1
 .smf files *see* machine files
#SNAME 3-4, 6-23
 source of printer paper 2-10
 space character 5-7
#SPEED 6-23, 6-45
@SPEEDS 5-18
 speeds, machine file questions for 4-9–4-11
 spelling errors in template files 7-3
 spindle control, template words for 6-45
#SPMODE 6-23, 6-45
#SPNDL 6-23, 6-45
#SPOFF 6-23
SQR 5-10, 6-45
#SREPT 6-24, 3-4
#STA 3-13, 6-24
#STANG 6-24, 6-37
@START 3-13, 3-14, 5-18
 starting Machine Define 2-2–2-3
#STB 3-13, 6-24
#STC 3-13, 6-24
@STEPCHG 5-19
@STOP 5-22
@STPROF 5-19
 string template words 5-6, 6-46
@SUBDEF 3-4, 5-21
 subprogramming
 application sections 5-21
 machine file questions for 4-54–4-55

- subroutines 3-3–3-6
 - drill 3-5
- switches 5-5
- symbols *see* characters
- syntax errors in template files 7-3
- system-defined template sections 5-2–5-3, 5-12–5-21
- #SYSTIME 6-24

- T**
- #TABLE() 6-24, 6-39
- TAN 5-11, 6-47
- #TANG 6-25, 6-37
- tape rewind character (%) 5-7
- Tape-to-Shape, machine file questions for 4-57–4-59
- #TCODE 6-25
- #TDESC 6-25, 6-46
- template (.tmp) files 1-10, 1-13, 5-1–5-22
 - // (double slashes) in 7-3
 - comments in 7-3
 - errors in 7-3
 - location of 1-6, 5-1
 - using literals in 7-3–7-6
- template sections 1-6, 5-1, 5-2
 - comments in 5-9
 - missing 5-3, 7-3
 - subprogramming application 5-21
 - system-defined 5-2–5-3, 5-12–5-21
 - types 5-2
 - user-defined 5-3, 5-22
- template variables
 - for multiaxis and rotary machining 3-12
 - user-definable 6-48
- template word categories
 - arcs 6-37
 - block number control 6-38
 - control functions 6-39
 - cycle time calculation 6-39
 - feed control 6-39
 - template word categories (*cont'd*)
 - fixed cycles
 - lathe groove 6-40
 - lathe peck drill 6-40
 - lathe tapping 6-40
 - lathe threading 6-41
 - lathe turning 6-43
 - lines 6-44
 - logic functions 6-45
 - mathematical functions 6-45
 - spindle control 6-45
 - string words 6-46
 - tool control 6-46
 - tool information 6-46
 - trigonometric functions 6-47
 - user-assigned words 6-47
 - work coordinate system 6-48
- template word reference 6-1–6-48
- template words 1-9
 - double-precision 5-6
 - full numeric 5-6
 - integer 5-6
 - numeric 5-6
 - switches 5-5
 - types 5-4–5-7
- test files 1-16
- test operators for conditions 5-8
- Text 2-12
- text editor 1-7
- text, searching for 2-12
- #THDPTCH 6-25
- threading (lathe), machine file questions
 - for 4-31–4-36
- three-dimensional arcs, machine file questions
 - for 4-47–4-50
- #TIME 6-26, 6-39
- time (cycle), machine file questions for 4-25–4-26
- #TINDX 3-13, 3-14, 6-26
- #TLCHG 6-26, 6-39, 6-46
- #TLCMT 6-27, 6-46

#TLDIA 6-27, 6-46
#TLEN 6-27, 6-46
#TLGRAPHIC 6-27
#TLID 6-27, 6-46
#TLOP 6-28, 6-46
#TLPAGEID 6-29
#TLSTA 6-29
#TLTIME 6-29, 6-39, 6-46
#TLTYP 6-46
#TLTYPE 6-29
#TLWD 6-30, 6-47
 .tmp files *see* template files
#TOFF 6-30, 6-47
#TOOL 6-30, 6-47
 tool change
 code for 3-6–3-9
 machine file questions for 4-12–4-15
 tool control, template words for 6-46
 tool information, template words for 6-46
 tool path 1-9
@TOOLCHG 3-7, 3-8, 3-9, 3-13, 3-14, 5-19
 tool-plane change
 code for 3-6–3-9
@TORCH 5-19
@TPINDX 3-7, 3-8, 3-9, 3-13, 3-14, 5-19
@TRAVERSE 5-20
 trigonometric functions
 operators 5-11
 template words for 6-47
 types of template sections 5-2
 types of template words 5-4–5-7

U

#U0–#U19 5-6, 6-31, 6-47
 UDTVs (user-definable template variables) 6-48
 units, machine file questions for 4-5–4-8
#UOV 6-31
#UPDATE() 6-31, 6-39
 user commands 5-3
 machine file questions for 4-56–4-57

user-assigned words, template words for 6-47
 user-definable template variables 6-48
 user-defined template sections 5-3, 5-22
 using literals in template files 7-3–7-6
 using the keyboard in Machine Define 2-15
 utilities
 Communicate 1-7
 Edit Plus 1-7
 see also Machine Define

V

#V0 6-41, 6-42, 6-43
#V0–#V19 5-6, 6-32, 6-47
#V1 6-40, 6-41, 6-42, 6-43
#V2 6-40, 6-41, 6-42, 6-43
#V3 6-41, 6-42, 6-43
#V4 6-41, 6-42, 6-43
 variables, template
 for multiaxis and rotary machining 3-12
 user-definable 6-48
 versions of machine files 2-2, 7-3
 View menu (Machine Define) 2-10–2-12
 Category List 2-11
 Item List 2-11
 Select Categories 2-11, 2-12
#VOV 6-31

W

@WAIT 5-20
#WINCL 6-32
 Windows
 starting Machine Define from 2-2
 window elements 2-3
 wire EDM (4-axis), machine file questions
 for 4-46–4-47
#WKPLN 3-13, 3-14, 6-32, 6-46
#WKSCHG 3-13, 3-14, 6-33, 6-48
@WKSYS 3-13, 5-20

words, template 1-9
 string 6-46
 user-assigned 6-47
 work coordinate system, template words for 6-48
 #WOV 6-31
 #WRAD 6-33

X

#XCTR 3-4, 6-33, 6-37, 6-41, 6-42, 6-43
 #XFO 3-12, 3-14, 6-34, 6-48
 #XHOME 3-14, 6-34, 6-48
 #XOV 6-34, 6-37, 6-40, 6-41, 6-42, 6-43, 6-44
 #XPASS 6-35, 6-40, 6-41, 6-42, 6-43
 #XPOFF 6-35
 #XPOS 6-35, 6-37, 6-41, 6-43, 6-44, 6-47
 #XPSET 3-12, 3-14, 6-36, 6-48
 #XPST 3-14
 #XSET 3-13, 3-14, 6-36, 6-47, 6-48
 #XST 3-12, 6-36, 6-37, 6-40, 6-41, 6-42, 6-43,
 6-44
 @XZARC 5-20

Y

#YCTR 3-4, 6-33, 6-37
 #YFO 3-12, 3-14, 6-34, 6-48
 #YHOME 3-14, 6-34, 6-48
 #YOV 6-34, 6-37, 6-44

#YPASS 6-35
 #YPOFF 6-35
 #YPOS 6-35, 6-37, 6-44, 6-47
 #YPSET 3-12, 3-14, 6-36, 6-48
 #YPST 3-14
 #YSET 3-13, 3-14, 6-36, 6-47, 6-48
 #YST 3-12, 6-36, 6-37, 6-44
 @YZARC 5-20

Z

#ZCHK 6-36
 @ZCHKMV 5-21
 @ZCLRMV 5-21
 #ZCTR 3-4, 6-33, 6-37, 6-41, 6-42, 6-43
 #ZDPTH 6-36
 @ZDPTHMV 5-21
 #ZFO 3-12, 3-14, 6-34, 6-48
 #ZHOME 3-14, 6-34, 6-48
 #ZOV 6-34, 6-37, 6-40, 6-41, 6-43, 6-44
 #ZPASS 6-35, 6-40, 6-41, 6-43, 6-44
 #ZPOFF 6-35
 #ZPOS 6-35, 6-37, 6-40, 6-41, 6-44, 6-47
 #ZPSET 3-12, 3-14, 6-36, 6-48
 #ZPST 3-14
 #ZSET 3-13, 3-14, 6-36, 6-47, 6-48
 #ZST 3-12, 6-36, 6-37, 6-40, 6-41, 6-43, 6-44