Simple introduction to CNC machining

Attached are some handouts that should help you in creating CNC programs to machine your parts. The first one gives a good introduction to milling machines and some basic cut motions. The second one covers how to select the cutting speeds and feeds so that you achieve a good finish on your parts and so that you don't break a tool or damage the machine. When calculating the speed, keep in mind that our machine has a max spindle speed of 5000 RPM (which actually is relatively fast for milling machines), so depending on your tool size it may be necessary to run a little slower than the optimum. When choosing feed rates, use a value of .005 inches per tooth for aluminum, and .010 inches per tooth for plastic or wax. The third handout shows the coordinate systems on a CNC milling machine, and discusses accuracy. Our machine has an accuracy to 4 decimal places, although if you specify more it will round it off for you. The final two handouts are from our machine's user manual, and they describe the instruction codes and also how to verify programs in the software simulation mode.

Here's the basic procedure for creating programs:

1. Choose the shape and size of stock that you will machine your part from. Determine how you will clamp your stock so that it can be held in a way that won't interfere with the cutting tools.

2. Pick an origin for your part. This will be the point that all other dimensions are referenced from. Typically you will set the origin to be the front left-hand corner of the part, on the top surface. This is the point that the machine will use to make all its moves in relation to.

3. Make a detailed drawing of your part, showing the dimensions of each cutout, and the positions of the cutout relative to the part origin. You will need to dimension both the lateral offsets (X-Y positions) and the vertical depth of cuts (Z positions).

4. Choose what size tool you will use for each of the cutouts. If you need to make a .250" slot, then the natural choice is a 1/4" end mill. If that slot has a rounded bottom, then you will want a 1/4" ball-end mill. By using a tool size that matches the cut size you will have to program fewer movements of the tool. Here are some of the tools we have available:

End Mills:	Ball-end Mills:
1/16"	1/8"
3/32"	1/4"
1/8"	3/8"
3/16"	1/2"
1/4"	3/4"
5/16"	
3/8"	
1/2"	
3/4"	

5. Write the G-codes to move each tool to the positions you want. Each G-code movement will be a position for the tip of the tool to move to. You will have to calculate where to position the tip in order to leave behind the edges you want. Don't cut too deep in a single pass: you should limit the cuts to be no more than 1/3 as deep as the diameter of the tool. Deeper slots will require multiple passes. Group the G-codes by tool, so that you do everything you need with one tool before changing to the next tool. That will save you the time of changing tools more than necessary when you run your program. Don't forget to raise the tool when you move from one area of your part to the next area, unless you are making a cut!

Keep in mind that G-codes are simply the same sort of movement instructions as a machinist would perform on a manual milling machine; you are just telling the milling machine where to move for each step, referenced to your part's origin.

I've installed the Benchman software on the right-hand PC in HBH 2202, so you can load your program and verify it using the instructions in the handout. Please verify your programs before you bring them to run on the milling machine.

Mike Vande Weghe
Research Engineer
Institute for Complex Engineered Systems
Carnegie Mellon University, HbH 2207
412-268-6846 / vandeweg@cmu.edu

1.4.4 Vertical-milling Machine (Vertical Miller)

A wide variety of operations involving the machining of horizontal, vertical, and inclined surfaces can be performed on a vertical-milling machine. As the name of the machine implies, the spindle is vertical. In the knee-type machine illustrated in Fig. 1.32 the workpiece can be fed either

1. Along the vertical axis (Z' motion) by raising or lowering the knee
2. Along a horizontal axis (Y' motion) by moving the saddle along the knee
3. Along a horizontal axis (X' motion) by moving the table across the saddle

In larger vertical-milling machines the saddle is mounted directly on the bed, and relative motion between the tool and workpiece along the vertical axis is

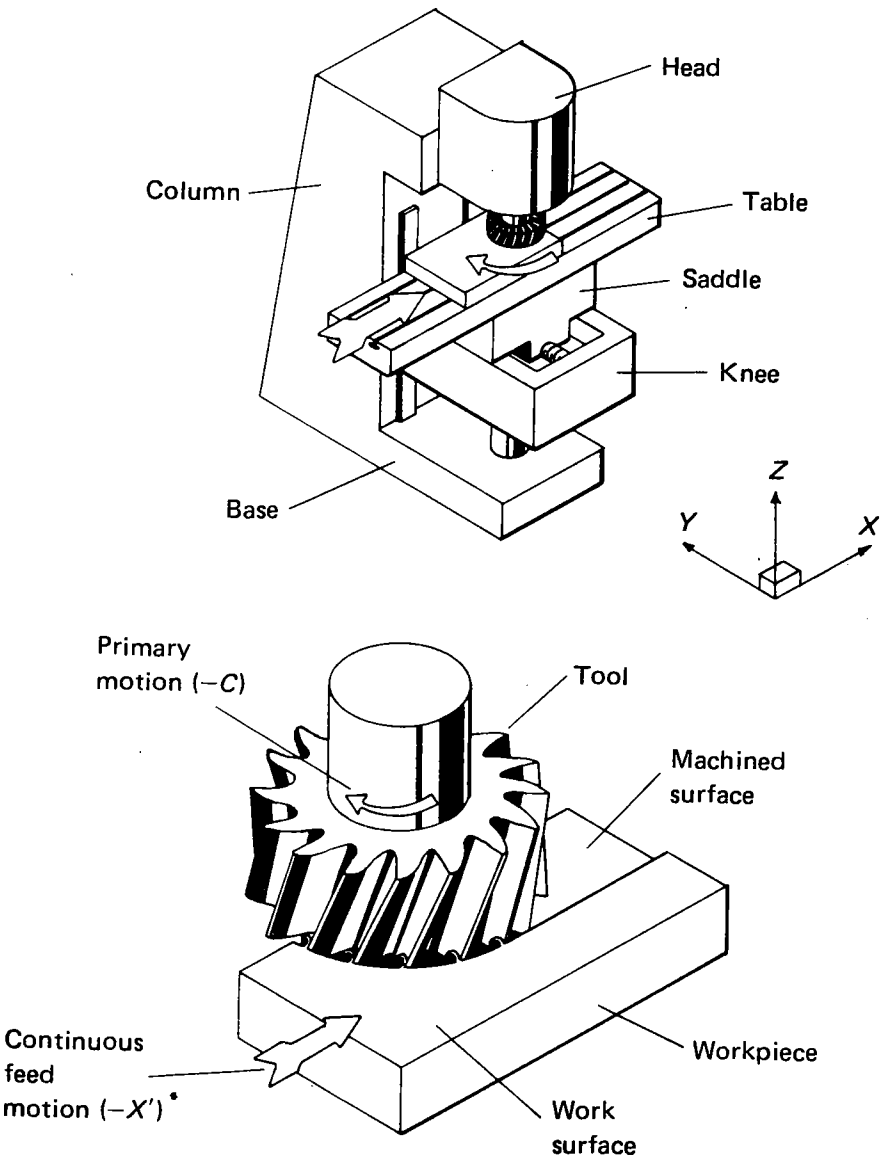


FIG. 1.32 Face milling on a knee-type milling machine.

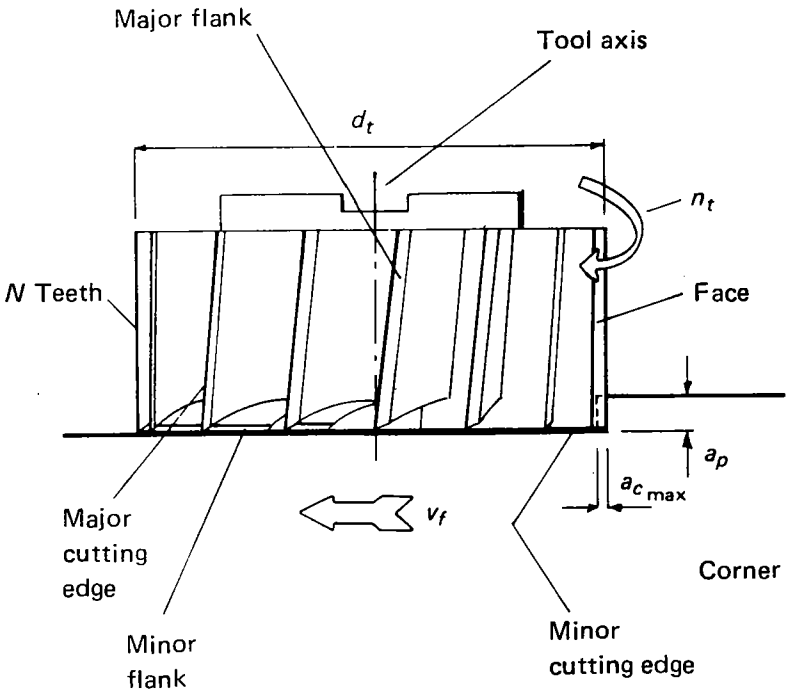


FIG. 1.33 Geometry of face milling, where $a_{c_{max}} = v_f/Nn_t$.

achieved by motion of the head up or down the column (Z motion); these machines are called bed-type, vertical-milling machines.

A typical face-milling operation, where a horizontal flat surface is being machined, is shown in Fig. 1.32. The cutter employed, known as a *face-milling cutter*, is shown in Fig. 1.33, which also illustrates the geometry of the operation.

The feed f is the distance the cutter advances across the workpiece during one revolution. Thus

$$f = \frac{v_f}{n_t} \tag{1.30}$$

where v_f is the feed speed, and n_t is the rotational speed of the cutter.

If the tool axis passes over the workpiece, the undeformed chip thickness increases to a maximum value and then decreases during the time each tooth is engaged with the workpiece; its maximum value, $a_{c_{max}}$, is equal to the feed engagement, which is equal to f/N, where N is the number of teeth on the cutter. Thus

$$a_{c_{max}} = \frac{v_f}{Nn_t} \tag{1.31}$$

In estimating the machining time t_m allowance should again be made for the additional relative motion between the cutting tool and workpiece. As can be

seen in Fig. 1.34, the total motion when the path of the tool axis passes over the workpiece is given by $(l_w + d_t)$ and therefore the machining time is given by

$$t_m = (l_w + d_t)v_f \tag{1.32}$$

where l_w is the length of the workpiece, and d_t is the diameter of the cutter.

When the path of the tool axis does not pass over the workpiece,

$$t_m = \frac{l_w + [2\sqrt{a_e(d_t - a_e)}]}{v_f} \tag{1.33}$$

where a_e is the working engagement. In this case the operation is similar to slab milling with a large working engagement, and the maximum value of the undeformed chip thickness will be given by Eq. (1.26).

The metal-removal rate Z_w in both cases is given by Eq. (1.29).

A variety of vertical-milling machine operations are illustrated in Fig. 1.35. It can be seen that, in one pass of the tool, several combinations of machined surfaces can be produced.

Milling cutters for vertical-milling machines generally have either a bore or a straight shank. Those having a bore are called *shell end-mills* and are secured to an arbor (Fig. 1.36) held in a socket in the machine spindle with a draw bar. Those having a straight shank are either gripped in a chuck or held in the spindle by a screw bearing on a flat surface, machined into the shank.

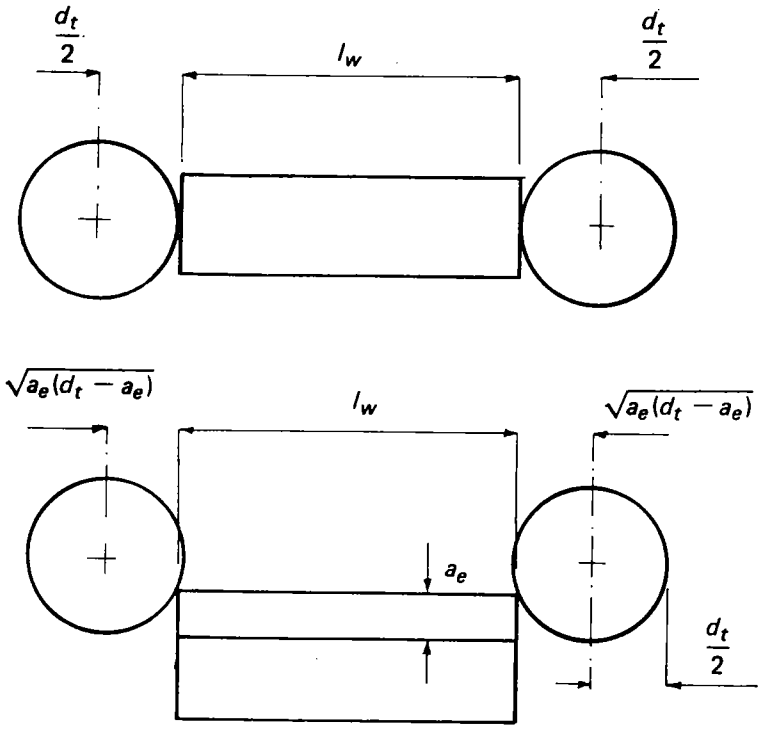


FIG. 1.34 Relative motion between the face-milling cutter and the workpiece during machining time.

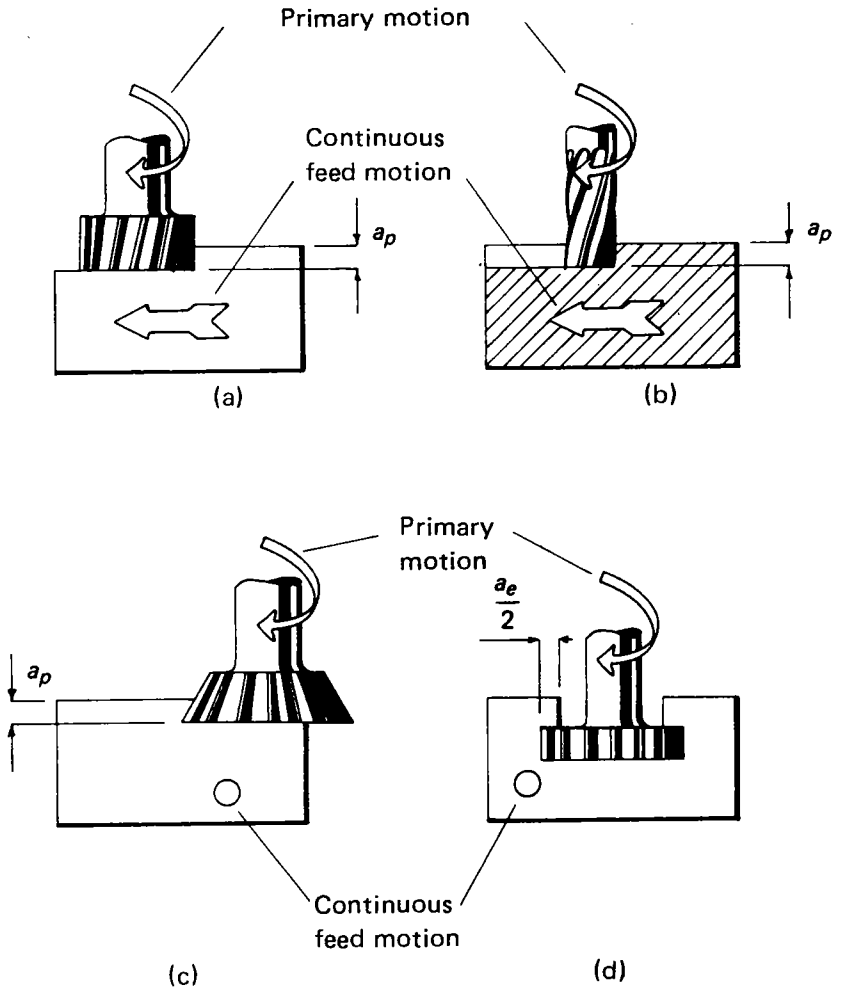


FIG. 1.35 Some vertical-milling-machine operations. (*a*) Horizontal surface; (*b*) slot; (*c*) dovetail; (*d*) T slot.

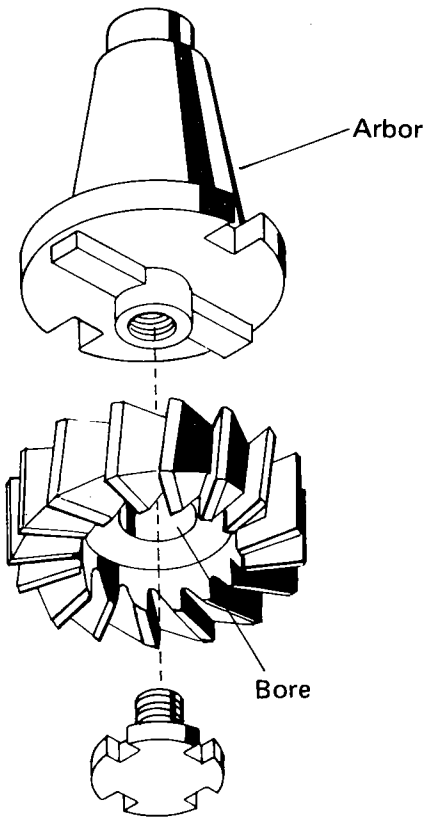


FIG. 1.36 Tool holding for a shell end mill.

Unit 76. Cutting Speeds and Feeds

To operate a milling machine successfully, you should learn how to figure cutting speeds and feeds. While there are formulas to help, you will find that there are many different things that affect cutting speed and feed. Judgment must be used in the final adjustment of the machine. The experienced machinist merely uses the result of the formula as a guide. You must remember that milling as a production process is primarily concerned with the rapid removal of metal. The amount of metal removed in a given length of time is the important thing in figuring correct cutting speeds and feed. As a beginner, it is usually best to start with slower speeds and feeds and then to increase them to the maximum limits. Of course, when you use a small hand miller with a hand feed, rapid production is a matter of skill and experience.

Cutting Speed

Cutting speed is defined as the distance one tooth of the cutter moves in one minute as measured in feet on the circumference. This is called the surface feet per minute (sfpm). In general, cutting speed is slower for harder materials and faster for softer ones. Cutting speed is not the same as machine speed (rpm). The spindle of the milling machine operates at certain given revolutions per minute. If you place a 4-inch-diameter cutter on the spindle, in one complete revolution a tooth will travel about 12 inches (4 × 3.1416). If a 1-inch-diameter cutter is employed, it will travel only about 3 inches (1 × 3.1416). The cutting speed in feet per minute is affected by many things. The two most important are the kind of material being machined and the kind of material in the milling cutter. The cutting speeds shown in Table 76-1 represent averages that will give a fast removal of metal and at the same time assure a long life to the cutter. Too fast a speed will dull or burn the cutter. The cutting speed for finishing cuts can be as much as 40 to 80 per cent higher than for roughing cuts. To find the machine speed to use or the revolutions per minute at which the spindle must rotate, the following should be used:

$$\text{rpm} = \frac{4 \times \text{sfpm (surface feet per minute)}}{d \text{ (diameter of cutter in inches)}}$$

Table 76-1. CUTTING SPEEDS*

Material to be milled	Material in cutter: Carbon tool steel	High-speed steel	Super high-speed steel	Stellite	Tantalum carbide	Tungsten carbide
	Cutter speed in feet per minute					
Aluminum	250–500	500–1000	.....	800–1500	.....	1000–2000
Brass, soft	40–80	70–175	.....	150–200	.....	350–600
Bronze:						
Hard	30–60	65–130	.....	100–160	.....	200–425
Very hard		30–50	.....	50–70	.....	125–200
Cast iron:						
Soft	30–40	50–80	60–115	90–130	.....	250–325
Hard		30–50	40–70	60–90	.....	150–200
Chilled			30–50	40–60	.....	100–200
Malleable iron	35–50	70–100	80–125	115–150	.....	250–370
Steel:						
Soft	30–45	60–90	70–100	.....	150–250	
Medium	30–40	50–80	60–90	.....	125–200	
Hard		30–50	40–70	.....	100–150	

* Learn what cutters will stand. Start with slow speeds and step up. For hand millers it is a good idea to use the lowest cutting speed shown in the table.

MACHINE TOOL METALWORKING
JOHN L. FEIRER AND EARL E. TATRO
McGRAW-HILL 1961

employed, it will
(1 × 3.1416). The
inute is affected by
important are the
hined and the kind
utter. The cutting
represent averages
al of metal and at
life to the cutter.
r burn the cutter.
ing cuts can be as
t higher than for
machine speed to
minute at which
following should

eet per minute)
ter in inches)

m e	Tungsten carbide
	1000–2000
	350–600
	200–425
	125–200
	250–325
	150–200
	100–200
	250–370
0	
0	
0	

vest cutting speed shown

Suppose you want to machine a piece of mild steel (SAE 1025) with a high-speed steel cutter that is 6 inches in diameter. The cutting-speed range for a high-speed steel cutter used on mild steel is between 79 and 97. Let's select a cutting speed on the lower side of about 80 sfpm. Then,

$$\text{rpm} = \frac{4 \times 80}{n} \quad \text{or} \quad \frac{320}{6} \quad \text{or} \quad 53.333$$

or about 53 or 54. The machine speed, or rpm, is adjusted in one of the following ways:

1. Most small bench and floor hand millers are belt-driven, just like a drill press. To change the simple speed you must change the position of the belt on the pulleys. A plate fastened to the side of the machine will tell you the machine speed when the belt is in different positions. Sometimes there is a back gear that can be engaged to give about eight possible machine speeds.

2. On larger milling machines, several methods can be followed for adjusting for speed:

   a. Some machines have two or three levers on the left side of the machine that can be moved to obtain the desired cutting speed.
   b. Other machines have one lever or knob and a speed-selector dial. When the lever or knob is turned, the machine speed will be indicated on the dial.

Feed

Feed is the rate at which the workpiece advances under the cutter. It is really the most important single factor in determining how fast metal can be removed from the workpiece. Actually, feed, plus depth of cut, plus width of cut, determines how many cubic inches of metal are removed from the workpiece in any given length of time. There are three ways feed is controlled:

1. *Manual feed.* On many small hand millers there is no power feed. Therefore you must depend on your own judgment about how fast to move the workpiece under the cutter. In general a slower feed is best for heavy roughing cuts and a faster feed for light, finishing cuts. Usually the tendency is to go too slow rather than too fast. A slow feed causes excessive wear on the cutter because a slow speed produces more rubbing than cutting action.

2. *Inches per revolution of the spindle or cutter.* On some machines the feed is directly related to the speed. As the speed increases, the feed increases. This feed arrangement is found on cone-driven machines. Some of the smaller hand millers have this feed arrangement.

3. *Inches per minute* (ipm). Most larger milling machines use a feed rate shown in inches per minute. The feed rate is independent of the speed. In other words, the feed is set at the desired amount in inches per minute. A change in the machine speed does not affect this feed.

The actual feed should be in terms of the following:

   a. The kind of material in the workpiece
   b. The kind of cutter
   c. The kind of material in the cutter
   d. The power available at the spindle (horsepower)
   e. The way the workpiece is held (how rigid the setup is)
   f. The shape and kind of workpiece (how rigid it is)

As a general rule, feeds should be as coarse as possible to obtain the desired finish. At the same time they must be fine enough to secure a long cutter life. The feed is generally reduced a little for the finish cut. A method commonly used to find the estimated feed rate in inches per minute is as follows:

$$F \text{ (feed rate)} = f \text{ (feed per tooth)} \times T \text{ (number of teeth)} \times N \text{ (rpm of cutter)}$$

Table 76-2. RECOMMENDED FEED PER TOOTH FOR HIGH-SPEED STEEL CUTTERS

Material	Face mills	Spiral mills	Slotting and side mills	End mills	Form cutters	Saws
Aluminum. Soft bronze	0.022	0.017	0.013	0.011	0.006	0.005
Medium bronze. Cast iron, soft	0.018	0.014	0.011	0.009	0.005	0.004
Malleable iron. Cast iron, medium	0.015	0.012	0.009	0.008	0.005	0.004
SAE X-1112 steel. Cast iron, hard	0.013	0.010	0.008	0.006	0.004	0.003
SAE 1020 steel. SAE X-1335 steel	0.011	0.009	0.007	0.005	0.004	0.003
SAE 1045 steel. Cast steel	0.009	0.007	0.006	0.005	0.003	0.003
Alloy steel:	0.008	0.006	0.005	0.004	0.003	0.002
Medium	0.007	0.005	0.004	0.004	0.002	0.002
Tough	0.005	0.004	0.003	0.003	0.002	0.0015
Hard	0.006	0.005	0.004	0.003	0.002	0.0015
Alloy tool steel						

Table 76-2 gives the average value for f for high-speed steel cutters of different kinds when used on various materials. You will notice that this formula takes into consideration only the first three factors that should be considered in feed selection. In production milling, the feed is determined in a different way. A chart developed by manufacturers of milling machines shows the maximum rate at which metal can be removed in cubic inches per minute. This rate is determined by the rated horsepower of the machine and the kind of metal being machined. For example, with a 15-horsepower machine, soft steel can be removed at a rate of 7 cubic inches per minute. If the depth of cut and the width of cut are known, the feed (F) in inches per minute can be found as follows:

$$F = \frac{\text{maximum metal removal in cubic inches per minute}}{\text{depth of cut} \times \text{width of cut}}$$

When the feed (F) in inches per minute has been found, then the feed (f) per tooth can be found as follows:

$$f = \frac{F \text{ (feed in inches per minute)}}{\text{cutter rpm} \times \text{number of teeth on cutter}}$$

This feed per tooth (f) in inches can be compared with the suggested feed per tooth for milling different materials, as shown in Table 76-2, to discover if the feed is too fast or too slow. The experienced machinist then uses his judgment for adjusting the correct feed. In production milling, the feed rate is the important factor in getting the maximum amount of milling completed in a minimum length of time.

Unit 77. Work-holding Devices

There are many different devices for holding the workpiece to be machined. Many are similar to those used on the drill press and shaper. The following are important attachments for use on the milling machine:

1. The *plain vise* is the handiest and most common work-holding device. The vise can be fastened to the table with the jaws either parallel or at right angles to the T slots.
2. The *swivel vise* is the same as the plain

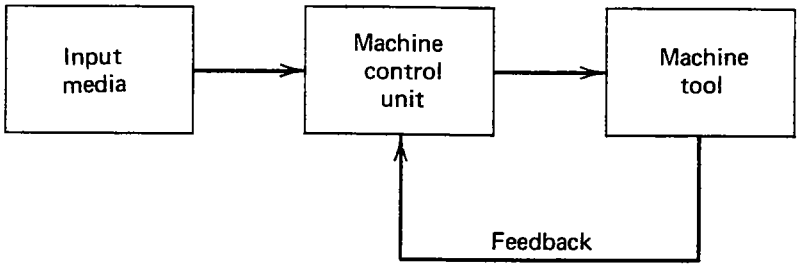


Figure 1.2 A simplified schematic of an NC system.

1.3 The Numerically Controlled Machine

We have defined NC as a method of automatic control that uses symbolically coded instructions to cause the machine to perform a specific series of operations. The most important of these instructions is that of *position.*

1.3.1 Positioning

Although modern NC systems perform many functions, the most important controlled operation is static or dynamic *positioning* of the cutting tool with the use of a system of coordinates that is general enough to define any geometric motion.

The right-handed Cartesian coordinate system, illustrated in Figure 1.3, provides a simple method for the definition of any point in three-dimensional space. While all NC machines make use of a coordinate system, some require only two axis (x and y) motion, and others require three-dimensional linear and angular axes. For the present purposes the 3-D Cartesian system is used.

Referring to Figure 1.3, the point P_1 is defined by the coordinate set (1, 2, 2), where each planar subdivision represents one unit. Likewise, point P_2 is defined by the coordinate set (3, 3, 4). The number of subdivisions within the three-dimensional space may be increased so that any point within the predefined boundaries may be described by a countable number of units.

Unlike a pure Cartesian system in which an infinite number of axial subdivisions is assumed, an NC coordinate system considers only a finite number of subdivisions. A later chapter will show that each *command* which is output by the control unit to the machine corresponds to one unit of motion. An NC machine is only as accurate as this smallest predefined subdivision. To illustrate this concept, consider a one-axis NC device which generates motion along the x-axis bounded by the values $x = -0.1$ m and $x = +0.1$ m. If the axis has 1000 predefined subdivisions, the machine could position to 0.2 mm. If 10,000 subdivisions were used, accuracy would increase tenfold to 0.02 mm. In these illustrations each command would correspond to 0.2 mm or 0.02 mm of motion.

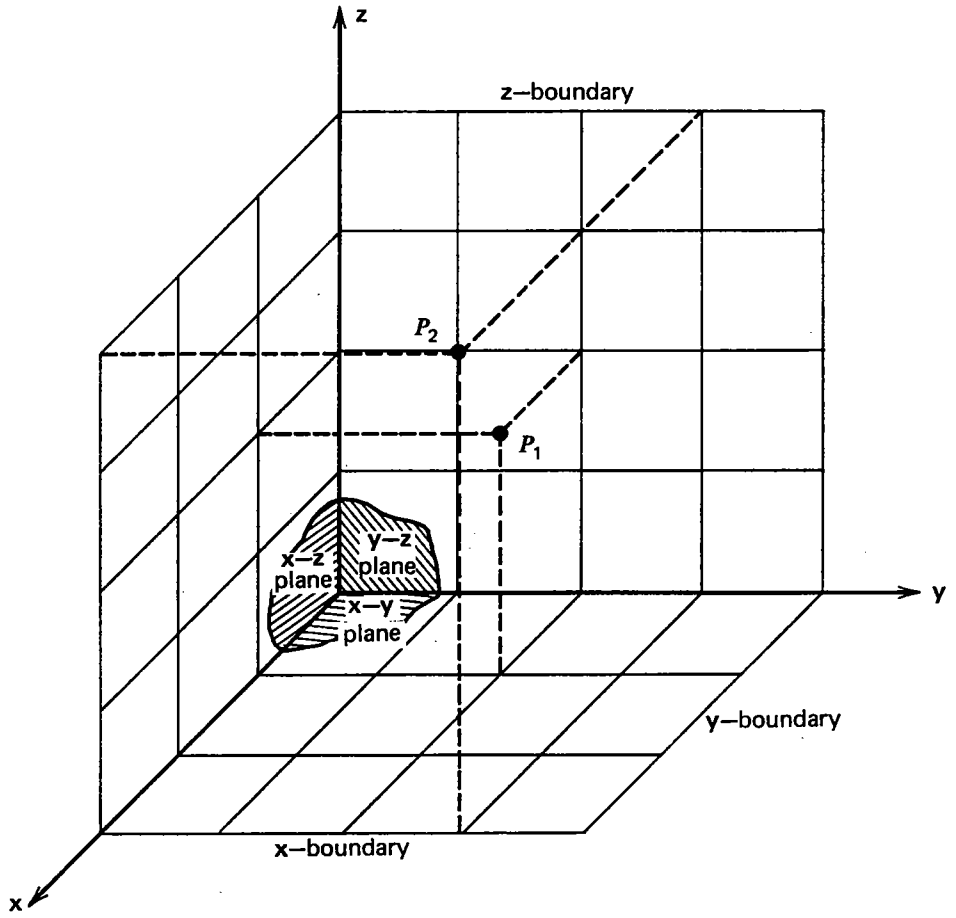


Figure 1.3 A bounded cartesian reference system.

An NC machine tool uses the coordinate system as a framework for positioning the cutting tool with respect to the workpiece. To accomplish this, points along the component profile* are defined by x, y, z coordinates. These coordinates are then fed in sequence to the NC controller which generates the appropriate positioning commands. A typical machine-axis configuration is illustrated in Figure 1.4. Any combination of spindle motion and/or work-holding table motion enables the proper position to be attained.

Positioning can be accomplished using two distinct methods. The first method, called *absolute positioning*, fixes the reference system and enables the actual x, y, z coordinates to be specified with reference to a fixed origin. Using an absolute positioning system, the points P_1 and P_2 would be specified as (x_1, y_1, z_1) followed by (x_2, y_2, z_2), regardless of the cutter position before the command is issued.

The second method of positioning uses *incremental* movement to obtain the same result. In an incremental device, the reference system is relative to the

* In many cases *offset* points are required (see Chapter Seven).

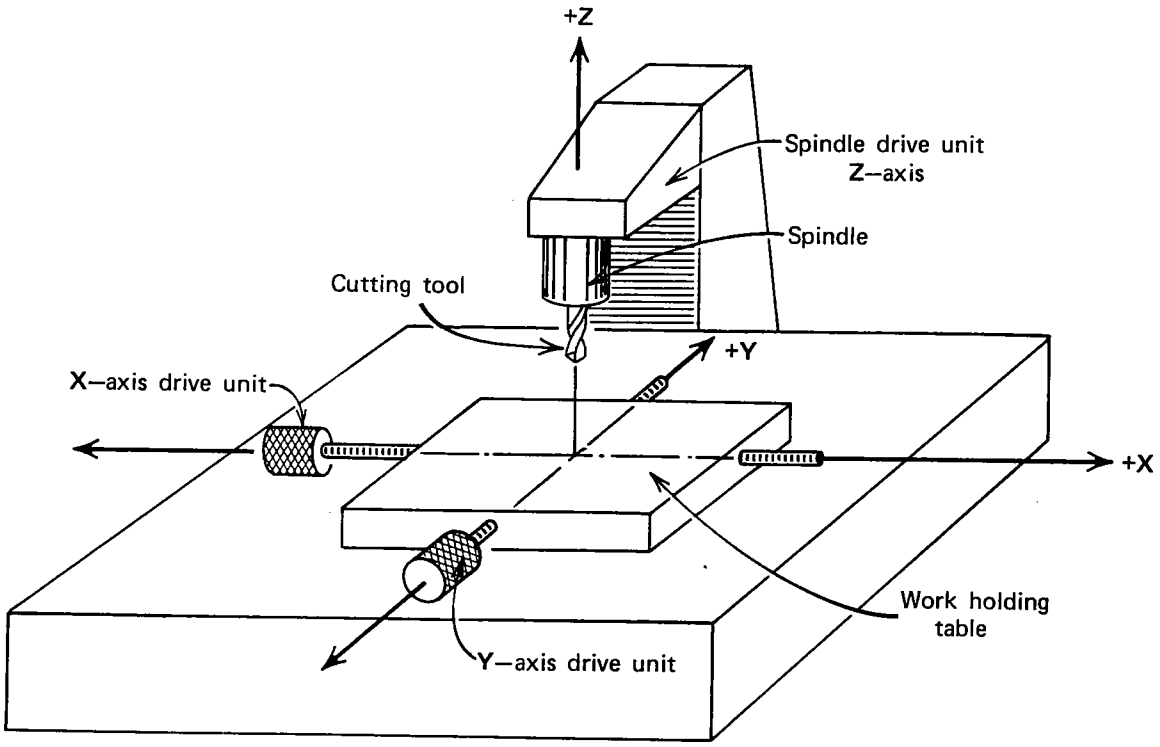


Figure 1.4 A typical NC machine axis system.

last position. For example, assume that the tool's current position is point $P_0(x_0, y_0, z_0)$. To specify a new position at P_1, the incremental distances, rather than the absolute coordinate values, are given. Let $\Delta x_{10} = (x_1 - x_0)$, $\Delta y_{10} = (y_1 - y_0)$, $\Delta z_{10} = (z_1 - z_0)$. Then, the new position is obtained by $(\Delta x_{10}, \Delta y_{10}, \Delta z_{10})$, where the reference system is assumed to have its origin at P_0. Similarly, a move from P_1 to P_2 is obtained by $(\Delta x_{21}, \Delta y_{21}, \Delta z_{21})$, where $\Delta x_{21} = (x_2 - x_1)$, etc. The incremental system therefore uses the change in x, y, z dimensions to specify position, whereas the absolute system uses coordinate values.

1.3.2 Control System

In our discussion of positioning, the path taken between points was disregarded. The path which the cutting tool follows as it traverses from point to point depends upon the type of control system used. Three basic path control systems are found in general usage.

The *point-to-point* system (also called a *positioning system*) effectively disregards the path between points. Each axis of motion is controlled independently so that the path steps from the start position to the next position as shown in Figure 1.5. The path shown is not unique as some point-to-point systems first satisfy the x command and then the y, whereas others reverse the

The Elements of an NC Part Program

Part programs generally incorporate two types of instructions: those which define the tool path (such as X, Y and Z axis coordinates), and those which specify machine operations (such as turning the spindle on or off). Each instruction is coded in a form the computer can understand.

An NC program is composed of *blocks* (lines) of code. The maximum number of blocks per program is limited by the memory (RAM) on your computer. You can, if necessary, chain programs together to form very large part programs.

Each block contains a string of *words*. An NC word is a code made up of an alphabetic character (called an *address character*) and a number (called a *parameter*). There are many categories of address characters used in NC part programs for the machining center (see Categories of NC Code).

Each block of NC code specifies the movement of the cutting tool on the machining center and a variety of conditions that support it. For example, a block of NC code might read:

```
N0G90G01X.5Y1.5Z0F1
```

If the machine is currently set for inch units, the individual words in this block translate as:

N0 This is the block sequence number for the program. Block 0 is the first block in the program.

G90 This indicates absolute coordinates are used to define tool position.

G01 This specifies linear interpolation.

X.5 This specifies the X axis destination position as 0.5".

Y1.5 This specifies the Y axis destination position as 1.5".

Z0 This specifies the Z axis destination position as 0". The cutting tool will move to the absolute coordinate position (0.5,1.5,0).

F1 This specifies a feed rate of 1 inch per minute, the speed at which the tool will advance to the specified coordinate points.

Categories of NC Code

There are many categories of NC code used for programming. The following is a list of the NC codes (designated by the address character) supported by the BENCHMAN 4000.

Code:	Function:
%	Incremental Arc Centers (Fanuc).
$	Absolute Arc Centers (LMC).
\	Skip.
/	Optional skip.
D	Compensation offset value.
F	Feed rate in inches per minute; with G04, the number of seconds to dwell.
G	Preparatory codes.
H	Input selection number; Tool length offset.
I	Arc center, X axis dimension (circular interpolation).
J	Arc center, Y axis dimension (circular interpolation).
K	Arc center, Z axis dimension (circular interpolation).
L	Loop counter; Program cycle (repeat) counter for blocks and subprograms; Specify homing tolerance.
M	Miscellaneous codes.
N	Block number (user reference only).
O	Subprogram starting block number.
P	Subprogram reference number (with M98); Uniform scale multiplier (with G51).
Q	Peck depth for pecking canned cycle.
R	Arc radius for circular interpolation (with G02 or G03); Starting reference point for peck drilling (with canned cycle codes).
S	Spindle speed.
T	Tool specification.
U	Incremental X motion dimension for absolute dimensioning.
V	Incremental Y motion dimension for absolute dimensioning.
W	Incremental Z motion dimension for absolute dimensioning.
X	X axis motion coordinate.
Y	Y axis motion coordinate.
Z	Z axis motion coordinate.
;	Comments.

Incremental Arc Center (%)

The incremental arc center code selects the Fanuc mode for programming arc coordinates. This mode is selected for the entire NC program as well as for any chained programs.

In the Fanuc mode, arc centers are always incremental, regardless of whether the system is in G90 (absolute) or G91 (incremental) mode. In contrast, arc center specifications in EIA-274 mode follow the selected programming mode, absolute or incremental.

You can specify the default arc center mode in the Run Settings dialog box.

This character must stand alone on the first line of the NC program in which it appears.

Absolute Arc Centers ($)

The absolute arc center code selects the EIA-274 mode of programming arc coordinates. This mode is selected for the entire NC program as well as for any chained programs.

In the EIA-274 mode, the mode of programming arc centers follows the selected programming mode; absolute (G90) or incremental (G91). In contrast, arc center specifications in Fanuc mode are always incremental, regardless of whether the whether the system is in absolute or incremental mode.

You can specify the default arc center mode in the Run Settings dialog box.

This character must stand alone on the first line of the NC program in which it appears.

Skip (\) and Optional Skip (/)

Note: The optional skip (/) code works only when the Optional Skip parameter from the Run Settings dialog box is on.

The Skip and Optional Skip codes allow you to skip particular lines of code in your program.

To use the Skip code (\):

Place the code at the beginning of the line you wish to skip. When you run the NC program, the specified line will be skipped.

To use the Skip code (\) with a parameter:

Use the Skip code with a parameter to instruct the Control Program to execute the line of code every nth pass. Place the code at the beginning of the line you wish to skip.

The syntax is: \n, where n is the number of passes between executions.

For example, if you want to execute a block of code every 5 passes, place \5 as the first code at the beginning of the block.

To use the Optional Skip code (/):

1. Place the code at the beginning of the line you wish to skip.
2. Select the Optional Skip option from the Run Settings dialog box or the Operator Panel.

When you run the NC program, the specified line will be skipped. If you do not select the Optional Skip option in the Run Settings dialog box, the skip code is ignored and the line is executed normally.

To use the Optional Skip code (/) with a parameter:

Use the Optional Skip code with a parameter to instruct the Control Program to execute the line of code every nth pass. Place the code at the beginning of the line you wish to optionally skip.

The syntax is: /n, where n is the number of passes between executions.

For example, if you want to execute a block of code every 5 passes, place /5 as the first code at the beginning of the block.

Compensation Offset Value (D Code)

The D code is used to select a value from the Control Program Offset Table. For example, D1 selects entry number 1 from the Offset Table.

Use the D code with:

- Cutter compensation codes to specify the tool radius.
- Tool offset adjust codes to specify a consistent increase or decrease in the commanded movement.

Use the Offsets command under the Setup Menu to view and manage the Offsets Table.

Feed Rate (F Code)

Use the F code to:

- Specify the rate of speed at which the tool moves (feed rate) in inches per minute (ipm). For example, F3 equals 3 ipm.

  The feed rate should be set to a low value (up to 50 ipm) for cutting operations. Feed rate values are in millimeters per minute (mpm) when using metric units. The Control Program limits the programmed feed rate so it doesn't exceed the maximum allowed by the machining center.

- Specify the number of seconds to dwell when used with the G04 code.

Preparatory Codes (G Codes)

G codes take effect before a motion is specified. They contain information such as the type of cut to be made, whether absolute or incremental dimensioning is being used, whether to pause for operator intervention, and so on.

More than one G code from different groups can appear in each NC block. However, you may not place more than one G code from the same group in the same block.

The G codes supported by the Control Program fall into the following groups:

- The Interpolation Group
- The Units Group
- The Plane Selection Group
- The Wait Group
- The Canned Cycle Group
- The Programming Mode Group
- The Preset Position Group
- The Compensation Functions Group
- The Coordinate System Group
- The Polar Programming Group

Note: More than one G code from different groups can appear in each NC block. *However, you may not place more than one G code from the same group in the same block.*

The Interpolation Group

The interpolation group allows you to specify the type of motion for interpolation. These G codes are retained until superseded in the NC program by another code from the interpolation group.

The supported interpolation G codes are:

G00 Rapid traverse
G01 Linear interpolation (default)
G02 Circular interpolation (clockwise)
G03 Circular interpolation (counterclockwise)

The Units Group

By default, an NC program is interpreted using the units of measure (inch or metric) specified using the Units command on the Setup Menu.

The codes in the Units group, G70 (inch) and G71 (metric), are used to override the Units command, for the entire program, or for a single line.

If the code is placed at the beginning of the program before any tool motions are made, that unit of measure is assumed for the entire program. If the code appears in a block of code, the unit of measure is in effect for that block and all following blocks. You can use these codes to switch between inch and metric modes throughout your program at your convenience.

The Fanuc equivalents, G20 (inch) and G21 (metric), can also be used.

The Plane Selection Group

This group of codes allows you to select different planes for circular interpolation. G17 is the Control Program default.

The supported Plane Selection Group codes are:

G17 Select the X, Y plane for circular interpolation. The arc center coordinates are given by I for the X axis and J for the Y axis.
G18 Select the X, Z plane for circular interpolation. The arc center coordinates are given by I for the X axis and K for the Z axis.
G19 Select the Y, Z plane for circular interpolation. The arc center coordinates are given by J for the Y axis and K for the Z axis.

The Wait Group

Wait Group codes apply only to the block in which they appear. The program does not continue until the wait conditions are satisfied.

The supported Wait Group codes are:

G04 Dwell (wait): Stop motion on all axes for the number of seconds specified by the F code, then continue the program. Because the F code is used to specify the number of seconds, you cannot also specify a new feed rate in the same block.
Example: G04F10; Wait for 10 seconds

G05 Pause: Used for operator intervention. Stop motion on all axes until the operator manually resumes program execution.

G25 Wait until TTL input #1 goes low before executing the operations on this block. Used for external device synchronization.

G26 Wait until TTL input #1 goes high before executing the operations on this block. Used for external device synchronization.

G31 Linear move to specified coordinate; used with H code to specify both the input number and the High or Low condition for stop (designated by the input operator, + or -). The move occurs until an input is triggered or until a coordinate is reached. The move stops short if specified input goes High (if H is positive) or Low (if H is negative). The default is input 1 High.
You can have the control program go to a specified block (N Code number) if the input meets the required condition. Use a P code to specify the destination, as with the M98 code.
For example, G31X5Y5H-2P50000 instructs:
Move (using the current programming mode) to the X and Y given.
If input 2 goes low during the move, jump to block number 50000.
If input 2 doesn't go low, continue with the next block in the program.

G35 Wait until TTL input #2 goes low before executing the operations on this block. Used for external device synchronization.

G36 Wait until TTL input #2 goes high before executing the operations on this block. Used for external device synchronization.

G131 Specifically for digitizing with the probe. The user specifies a Z position and a feedrate. The probe moves from its current position to the specified Z position at the specified feedrate (or current feedrate if not specified on the same block). If the probe input is tripped before reaching the specified Z position, a valid point is captured. In either event (point or no point), when the probe stops moving down, it rapids back to the initial Z position.

The Canned Cycle Group

Canned cycle codes allow you to perform a number of tool motions by specifying just one code. Canned Cycle codes are typically used for repetitive operations to reduce the amount of data required in an NC program. Canned cycle codes are retained until superseded in the program by another canned cycle code.

The supported Canned Cycle codes are:

G80 Canned cycle cancel
G81 Canned cycle drilling
G82 Canned cycle straight drilling with dwell
G83 Canned cycle peck drilling
G85 Canned cycle boring
G86 Canned cycle boring with spindle off (dwell optional)
G89 Canned cycle boring with dwell

Refer to Section G for more information on these functions.

The Programming Mode Group

Programming mode G codes select the programming mode, absolute (G90) or incremental (G91). These codes remain in effect until superseded by each other. The default code on program start up is G90.

With absolute programming, all X, Y and Z coordinates are relative to origin of the current coordinate system. With incremental programming, each motion to a new coordinate is relative to the previous coordinate.

The supported Programming Mode codes are:

G90 Absolute programming mode
G91 Incremental programming mode

The Preset Position Group

The preset position G codes move the tool to a predetermined position, or affect how future motions will be interpreted.

The supported Preset Position codes are:

G27 Check reference point: This code moves the machine to its home position and compares the reported position against zero to see if any position has been lost. The difference between the reported position and zero is compared to a tolerance value specified using the Setup Program. Use the L code in this block to override the tolerance value in the Setup Program.

G28 Set reference point: This code moves the machine to its home position and sets the machine position to 0,0,0. The G28 code performs an automatic calibration of the axes.

G92 Set position: This code works like the Set Position command under the Setup Menu. The X, Y and Z coordinates following a G92 code define the new current position of the tool.

G98 Rapid move to initial tool position after canned cycle complete.

G99 Rapid move to point R (surface of material or other reference point) after canned cycle complete.

The Compensation Functions Group

Use the cutter compensation NC codes to automatically compensate for the variations in a cutting tool's radius and length. Refer to Section H for more information on using cutter compensation.

The supported Compensation codes are:

G39 Corner offset circular interpolation.

G40 Cancel cutter compensation.

G41 Left cutter compensation: Enables cutter compensation to the left of programmed tool path.

G42 Right cutter compensation: Enables cutter compensation to the right of programmed tool path.

G43 Tool length offset: Shifts Z axis in a positive direction by a value in the Offset Table, specified by an H code.

G44 Tool length offset: Shifts Z axis in a negative direction by a value in the Offset Table, specified by an H code.

G45 Tool offset adjust: Increases the movement amount by the value stored in the offset value memory.

G46 Tool offset adjust: Decreases the movement amount by the value stored in the offset value memory.

G47 Tool offset adjust: Increases the movement amount by twice the value stored in the offset value memory.

G40 Tool offset adjust: Decreases the movement amount by twice the value stored in the offset value memory.

G49 Cancels tool length offset.

G50 Cancels scaling.

G51 Invokes scaling.

G68 Invokes rotation.

G69 Cancels rotation.

The Coordinate System Group

Use the coordinate system codes to establish multiple coordinate systems on one work piece to create multiple parts.

For instance, you can run a part program using a typical coordinate system (with the point of origin on the surface of the front left corner of the workpiece), then select another coordinate system which has its origin at a different point on the surface of the workpiece. For an overview of coordinate systems, see page H-27.

There are seven coordinate system codes. One of these codes (G53) is used to rapid to specified machine coordinates. The other six codes allow you to make up to six individual parts on the same workpiece by specifying different work coordinate systems for each part.

The coordinate system codes are G54 through G59, referring to coordinate systems 1 through 6 respectively. These coordinate systems may be set through the Coordinate Systems command on the Setup Menu.

The Polar Programming Group

The polar programming codes allow you to perform polar programming operations, based on polar coordinates. The polar coordinates are defined by X (radius) and Y (angle in degrees) when programming for the X, Y plane. Refer to Section H of this Guide for more information on using polar programming.

The supported Polar Programming codes are:

G15 Polar programming ON
G16 Polar programming OFF

Input Selection Number/Tool Length Offset (H Code)

The H code has multiple uses. It can be used to specify inputs, input state changes, outputs, and offset amounts.

Use the H code in conjunction with:

- The wait codes G25 and G26, to specify the input number. If the H code is not used with these G codes, input 1 is assumed.
- The wait code G31, to specify input change to high or low. If the H code is not used with this G code, input 1 High is assumed.
- The tool length offset codes G43 and G44, to specify the amount of Z axis shift. (The Offset Table you use for Tool Length Offset H values is the same table you use for Cutter Compensation and Tool Offset Adjust D values.)
- The transmit codes M25 and M26 for interfacing with robots or other external devices, to specify the output number. If the H code is not used with these M codes, output 4 is assumed.

X Axis Coordinate of Center Point (I Code)

In absolute programming mode (G90), the I code specifies the X axis coordinate of the center point of a circle when using circular interpolation. In incremental mode (G91), the I code specifies the X axis distance from the start point of motion to the center point of the circle for circular interpolation.

If no I code is specified, the system uses the current X axis location as the X axis center of the arc.

In Fanuc mode, all arc centers are incremental.

The I code is also used with the G51 code to specify the scale factor for the X axis when performing scaling functions, including scaling each axis and mirror scaling. Refer to Reference Guide Section H for more information on using scaling.

Y Axis Coordinate of Center Point (J Code)

In absolute programming mode (G90), the J code specifies the Y axis coordinate of the center point of a circle when using circular interpolation. In incremental mode (G91), the J code specifies the Y axis distance from the start point of motion to the center point of the circle for circular interpolation.

If no J code is specified, the system uses the current Y axis location as the Y axis center of the arc.

In Fanuc mode, all arc centers are incremental.

The J code is also used with the G51 code to specify the scale factor for the Y axis when performing scaling functions, including scaling each axis and mirror scaling. Refer to Reference Guide: Section I for more information on using scaling.

Z Axis Coordinate of Center Point (K Code)

In absolute programming mode (G90), the K code specifies the Z axis coordinate of the center point of a circle when using circular interpolation. In incremental mode (G91), the K code specifies the Z axis distance from the start point of motion to the center point of the circle for circular interpolation.

If no K code is specified, the system uses the current Z axis location as the center of the arc.

In Fanuc mode, all arc centers are incremental.

The K code is also used with the G51 code to specify the scale factor for the Z axis when performing scaling functions, including scaling each axis and mirror scaling. Refer to Reference Guide: Section I for more information on using scaling.

Angle of Arc Resolution, Loop Counter (L Code)

The L code specifies the angle of arc resolution in circular interpolation programming. With BENCHMAN, it is only used if a helical motion is executed, or if you specifically enable approximated arcs in that program (M111 command).

Use the L code with:

- The M98 code as a loop counter for subprograms.
- The M47 code as a program cycle counter, to repeat a program a finite number of times.
- The G27 code to specify tolerance with homing commands (this is an LMC-specific NC language extension). The difference between the current position and 0 is compared to a tolerance value specified using the Setup Program; use the L code to override this tolerance value.

Miscellaneous Codes (M Codes)

Note:

All M codes used to turn on a device, such as the spindle, execute at the beginning of the tool motion for that block of NC code.

All M codes used to turn off a device execute after the tool motion for that block is completed.

To avoid confusion, it is sometimes easier to place M codes in a separate block from the motion commands.

M codes control a variety of functions while a part program is running. M codes should be placed on separate blocks to avoid confusion over whether an M code is activated during or after a motion command.

The supported M codes are:

M00 Pause: Allows you to place a pause in your code. Acts like a G05 pause.

M01 Optional Stop: Allows you to place an optional pause in your code. Place an M01 in the block of code where you would like to pause. There are switches to activate or deactivate the Optional Stop code in the Run Settings dialog box and on the Operator Panel. With Optional Stop on, the M01 works like a G05 pause. With Optional Stop off, the M01 code is ignored, and the other codes on the block are executed as usual.

M02 End of Program: Takes effect after all motion has stopped; turns off drive motors, spindle and accessories.

M03 Spindle Motor On Forward: Activated concurrently with motion specified in the program block; remains in effect until superseded by M04 or M05.

M04 Spindle Motor On Reverse: Activated concurrently with motion specified in the program block; remains in effect until superseded by M03 or M05.

M05 Spindle Motor Off: Activated after the motion specified in the program block; remains in effect until superseded by M03 or M04.

M06 Tool Change: Pauses all operations, turns off spindle, retracts spindle for tool change.

M08 Coolant On: Turns on coolant; remains in effect until superseded by M09.

M09 Coolant Off: Turns off coolant; remains in effect until superseded by M08.

M10 Air Vise On: Unclamps Air Vise; remains in effect until superseded by M11.

M11 Air Vise Off: Clamps Air Vise; remains in effect until superseded by M10.

M20 Chain to Next Program: This code is used to chain several NC files together. It appears at the end of a part program and is followed on the next line by the file name of another program which is executed when all motion stops. Here's an example of a part program chain to another program:

```
N37 Z.2
N38 M20
PROGRAM2.NC ;CHAIN TO PROGRAM TWO
```

If the two programs you are chaining are not in the same directory on your computer, you must specify the full path name for the next program file.

M22 Output current position to file. Typically used in digitizing.

M25 Set TTL output #1 Off: Used for external device synchronization.

M26 Set TTL output #1 On: Used for external device synchronization.

M30 End of program: Same as M02.

M35 Set TTL output #2 Off: Used for external device synchronization.

M36 Set TTL output #2 On: Used for external device synchronization.

M47 Rewind: Restarts the currently running program; takes effect after all motion comes to a stop. Typically used with an L code to repeat a program a set number of times.

M98 Call to subprogram. Use the P code to specify the subprogram starting block number. Use the L code to specify the number of times the subroutine is executed. You can nest subprogram calls to a depth of 20.

M99 Return from Subprogram; Goto

M105 Operator Message (LMC)

M122 Output current position to file. Almost identical to M22, except that if a macro (@X@Y@Z) is used to insert a coordinate, the position of the digitized point will be used, rather than the current machine position.

M22: Output Current Position to File

The M22 code is used to write information to a file while a program is running. Typically, this code is used when digitizing to write the current X, Y, and Z machine coordinates to a file. The proper format for using this code is: M22(*filename*)DataToWriteToFile. The first time the Control Program encounters an M22 code, it opens the specified file. You must enclose the name of the file in parentheses for the Control Program to recognize it. If you do not specify any DataToWriteToFile text, the default data is output. This default is the current position, equivalent to specifying 'X@X Y@Y Z@Z'. Notice that the @X,@Y,@Z 'macros' are replaced by the actual machine position when the data is written. Each M22 code automatically adds a line feed to the end of its output so the next M22 starts on a new line.

If the file name is followed by ",A" (e.g., test.nc,A), the Control Program does not delete previous information from the file, it appends the information to the end of the existing information. If the file does not exist, it is created.

If you use more than one M22, only the first occurrence must have the file name in the parentheses. The remaining M22's may have empty parentheses, ( ), or may specify a different file.

If you want to generate more than one file at a time, you must include the filename each time you specify M22. If a filename is not specified, the first file opened is used.

Example:

```
. . .              ; code to move to position
; Open my1.xyz, discard contents, write coordinates
M22(my1.xyz)
. . .              ; code to move to next position
; Append to currently open data file
M22( )
. . .              ; code to move to next position
; Open my2.xyz and append coordinates
M22(my2.xyz,A)
```

Information about digitizing is provided with the digitizing package. For additional information, please call Light Machines Technical Support.

Special codes that can be used with M22 to generate run-time reports:

@X	Current X position (in current coordinate system).
@Y	Current Y position (in current coordinate system).
@Z	Current Z position (in current coordinate system).
~ (tilde)	New Line (starts a new line in the file).
@TD	Time of Day (12 hour) "11:59:59AM"
@TC	Time (elapsed) for cycle "99:11:59" (0's trimmed from left)
@TT	Time Total (program run) "99:11:59"
@TA	Time Average (per cycle) "99:11:59" ("??:??" if first part)
@TL	Current Tool #. "5"
@C	Cycle # (Current Pass) "3"
@D	Date "12/31/94"
@FN	Current File (w/o path) "PART.NC" ("UNTITLED.NC" if untitled)
\t	TAB
\\	Outputs a single '\' character.

Example:

```
; Start of file
... ; Process a single part
```

```
; Output part time statistics to file c:\Reports\Stats.txt
        (c:\Reports directory must exist)
M22 (c:\Reports\Stats.txt,A) Part #@C processed in @TC.
M47 L50 ; We want to process 50 parts.
```

M99: Return from Subprogram, Goto

The M99 code has two specific uses; it can be used as a command to return from a subprogram or it can be used as a goto command.

Using M99 with subprograms:

When used in a subprogram, this code returns you to the block following the last M98 (Call to Subprogram) command.

You can use the P code plus a block number to override the block returned to; however, if this feature is used from a nested subprogram call, all return targets are discarded. The rules for a Goto target block apply to this use as well.

Using M99 as a Goto command:

This command can be used in the main NC program as a Goto command to jump to a block on a line before the M02 command.

Use the P code to identify the block number being jumped to. Control is transferred to the first occurrence of this N code; it cannot be used to transfer control between chained programs (see M20).

This command can be used anywhere in the program to change the flow of program execution. It is good programming practice to place this command on a line by itself to improve the program's readability.

M105: Operator Message

This command is used to display messages in the Control Program. It provides a way to display messages to the operator on the Message Bar while an NC program is running. You can also pause the program with a custom message. This code is a non-standard, Light Machines code.

By default, the message is centered, displayed as black characters on a white background, and is persistent (not cleared until the operator manually does so or until the next message is displayed).

The correct format for using this code is:

```
M105(the message);comment
```

For instance, the following line of code displays a simple message:

```
M105(End of Roughing Segment);Display message in white;continue
```

Messages can be altered by using the following alternate characters:

^ Displays the message and performs a pause requiring operator intervention to continue.

~ Displays the message as a Warning Message.

\b Beeps when the message is shown.

The format for using the M105 code with an alternate character is:

M105(alternate character plus the message) ;comment

For example:

M105(~WARNING);Message in yellow, continue

Here are some other examples of how to use this code:

M105( ) ; clears current message
M105(^Please stop and read this!) ; Normal Message, pauses
M105(~^I MEAN IT!) ; Warning Message, pauses
M105(\b\b\b) ; Clears current message, beeps 3 times, and doesn't pause

Block Number (N Code)

N codes have two uses:

- To provide destinations for Gotos (M99) elsewhere in the program.
- To clearly show the organization of the code and improve readability.

Using the N code is optional; however, when you do use the N code, it must be the first character in a block of NC codes.

Other than for the above stated uses, N codes are not recognized by the Control Program. Their presence, absence, or sequential value does not affect the execution of the NC program in any way (unless the target of a goto is missing).

You may have N codes on some blocks and not on others. N code sequence numbers do not have to be in order, but regular sequential order does make it easier to follow the program.The Control Program can change the N codes in a program by inserting, removing, or renumbering them. See Edit Menu, Renumber Command in Section E.

Subprogram Block Number (O Code)

The O code is used to indicate the start of a subprogram, and must be followed by a number which identifies the subprogram. The O code replaces the N code in the first block of the subprogram.

To call a subprogram, use the M98 code; the P code specifies which subprogram to execute. To return from the subprogram, use the M99 code.

Only the first block in the subprogram contains the O code. The remaining blocks may contain N codes. The O and N code numbers may be used to help identify and set apart the subprogram to improve readability, for example:

```
M98P50000;call to first subprogram
...;after first subprogram is finished, M99 code returns to this point
...
M98P60000 ;call to second subprogram
...;after second subprogram is finished, M99 code returns to this point
...
O50000 ;start of subprogram
N50010 ;first line of subprogram
N50020 ;second line of subprogram
N50030M99 ;last line of subprogram
...
O60000 ;second subprogram
N60010 ;first line of second subprogram
N60020 ;second line of second subprogram
N60030M99 ;last line of second subprogram
```

Subprogram Reference Number (P Code)

Use the P code with:

- The G31 code to reference a goto target block.
- The G51 code to specify a uniform scaling factor.
- The M98 code to reference a subprogram using the subprogram block number.
- The M99 code to specify a return block number as a goto target.

Peck Depth (Q Code)

The Q code is used with the G83 code in canned cycle peck drilling to specify the depth of each peck.

Radius of Arc, Drilling Start Location (R Code)

Note: Full circles (360° arcs) cannot be performed with an R code. Split the arc into two arcs or use center point (I, J and K) values for full 360° circles.

As an alternative to specifying the center point of an arc (I Code, J Code, or K Code) you can specify the arc radius. Use the same value for the radius in both absolute and incremental programming modes.

G02 or G03 specifies the direction of motion.

Positive values for R (radius) are specified for arcs up to 180°. Negative values are used for arcs greater than 180°. Full circle arcs cannot be performed with an R code. Spit the circle into two arcs, or use center point (I, J, and K) values for full 360° circles.

Use the R code in canned cycles to specify a Z axis reference point for peck drilling. The point can be at the material surface or at another reference point. The R code is also used to specify the rotation angle, in degrees, with the G68 code.

Spindle Speed (S Code)

Use the S code to set the spindle speed from within the NC program. Spindle speed is specified by the address character "S" followed by a parameter that represents the speed in RPM. For example, S750 is the designation for a spindle speed of 750 RPM. For the S code to have affect the spindle must be turned on by the M03 or M04 command. If the spindle is off, the spindle speed is stored and used when the spindle is turned on again within the program. Use the M05 command to turn the spindle off.

CAUTION

Using multiple tools is an advanced operation, and should not be attempted by persons unfamiliar with using the machining center.

Tool Selection (T Code)

T codes specify the tool offset (number) in the Tool Library in multiple tool machining operations. They ***do not*** specify the Automatic Tool Changer (ATC) station number, but they ***do*** have an ATC station associated with them. Tools are specified by the address character "T" followed by a parameter that represents the number of the tool. For example, T3 is the designation for tool number three.

X Axis Coordinate (X or U Code)

An X code specifies the coordinate of the destination along the X axis. A U code is used in absolute programming mode (G90) to specify an incremental X motion. You cannot use the U code to mix incremental and absolute programming in the same block.

Note: Do not place absolute and incremental commands in the same block. For example:

G90X1V1

will not produce the expected motions.

Y Axis Coordinate (Y or V Code)

A Y code specifies the coordinate of the destination along the Y axis. A V code is used in absolute programming mode (G90) to specify an incremental Y motion. You cannot use the V code to mix incremental and absolute programming in the same block.

Z Axis Coordinate (Z or W Code)

A Z code specifies the coordinate of the destination along the Z axis (spindle axis). A W code is used in absolute programming mode (G90) to specify an incremental Z motion. You cannot use the W code to mix incremental and absolute programming in the same block.

Comment Codes

The Control Program allows you to add *comments* to your NC blocks. The Control Program recognizes two comment codes:

- A semicolon ";"
- An open parenthesis "("

These two comment codes are equivalent. The use of either of these codes at the end of an NC block indicates that a comment follows.

Comments must follow all other NC codes in the block. Comments are ignored when the part program is executed. Comments can be placed on a block without any NC codes to document what is occurring within a program. NC programmers use these comments to annotate their programs.

Here is an example of an NC block with a comment:

```
X0Y0Z0;MOVE TO ZERO POINT
```

The comment tells us that the X, Y and Z codes in this block command the cutting tool to move to the zero point (coordinate 0,0,0).

Comments can be combined with the G05 pause and the M06 Tool Change codes to display messages to the operator during program execution. Here is an example of an NC block with a pause coded comment:

```
G05(ROUGH DIAMETER SHOULD BE 0.5 in.
```

When the program pauses, the program line, and thus the comment, is displayed on the message bar, telling the operator to verify the diameter of the workpiece before continuing. The M105 code provides a more versatile and powerful message facility.

By using the Renumber command you can strip the comments from a program with a single command; however, comments cannot subsequently be replaced automatically.

General Programming Suggestions

The following rules should be followed when writing NC part programs.

- The sequence of words (address characters plus parameters) in an NC block must appear in the following order: /, N (O), G, X (U), Y (V), Z (W), A, I, J, K, R, Q, H, L, F, S, T, M, P, ; A different order may cause unpredictable results.
- In many cases, a word need not be repeated in the next block (line). The system assumes no change in codes unless a new code appears. This does *not* apply to: N words, I, J, and K, G04, G05, G25, G26, G92, F used for dwell, M02, M20, M25, M26, M30, M47, M98 or M99.
- You can use more than one G code in a block; however, you can use only one G code from any one group in a single block.
- N codes (sequence numbers) are not required in a part program; however, they can be useful in identifying a block when editing a long NC part program.
- An O code is required to mark the beginning of a subprogram and does not have to be in sequence with the N codes.
- The first instruction in a part program should move the tool to a safe starting position. This makes restarts much easier.
- The last block of a program should move the tool back to the starting position. The tool will then be in position to start cutting another part.
- Part programs should reference the zero point with Z0 at the point where the tool just touches the work piece. This convention allows for standardization of programming.
- Before running an NC part program:
  a. Look for the typical coding error that places two X codes, two Y codes, or two Z codes in the same block.
  b. Be sure that all required coordinates have been written into appropriate blocks.
  c. Verify the part program to discover any program errors.
  d. Run the part program without mounting stock in the Machining Center to see if the tool movements are logical.
- The first portion of a part program should turn on the spindle and establish the feed rate and spindle speed.
- M codes should be placed on separate blocks to avoid confusion over whether an M code is activated before or after a motion command.
- Double-check all program blocks against your coding sheet to locate and correct typographical errors.

Linear Interpolation Programming

Linear interpolation is the movement of the tool in a straight line from its current position to a coordinate location specified by an NC block. Here's a typical block of NC code using linear interpolation:

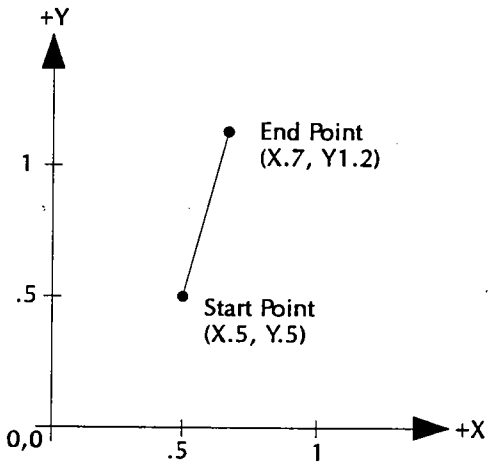
N5G90G01X.7Y1.2F2

Broken down into individual words:

- **N5** The block sequence number is 5
- **G90** Coordinates are given using absolute dimensioning
- **G01** Linear interpolation is specified
- **X.7** X axis coordinate of end point = .7
- **Y1.2** Y axis coordinate of end point = 1.2
- **F2** Feed rate is 2 inches per minute

The G01 code *is* required when switching from circular interpolation or rapid traverse positioning back to linear interpolation. If we assume the current position of the tool is X.5, Y.5, the tool movement generated by the above block is something like this:

Typical tool movement using linear interpolation



An equivalent movement is achieved with incremental dimensioning (G91):

N5G91G01X.2Y.7F2

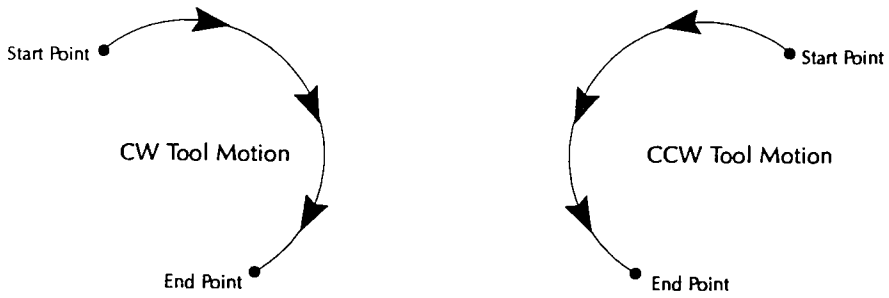
Circular Interpolation Programming

Circular interpolation moves the cutting tool along an arc from the starting point specified in one block, to an end point specified in the next block. The curvature of motion is determined by the location of the center point (I, J, or K), which must also be specified in the second NC block.

The direction of rotation from the starting point determines the actual shape of the arc relative to the spindle axis. A G02 code moves the tool in a clockwise (CW) motion from the starting point. A G03 code moves the tool in a counterclockwise (CCW) motion from the starting point.

Here are two typical blocks of NC code using circular interpolation:

G02 (CW) and G03 (CCW) cutting paths



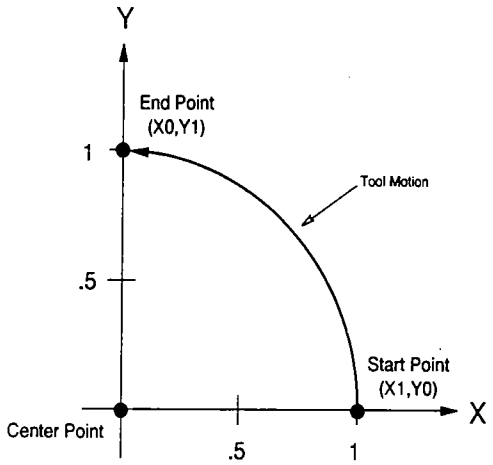

```
N9G90X1Y0;SET START POINT
N10G03X0Y1I-1J0F2;COUNTERCLOCKWISE TO X0,Y1
```

The first block defines the starting point. The second block defines the end point and the center of the arc. Broken down into individual words, the second block reads:

- **N10** The block sequence number is 10
- **G03** The tool will proceed in a counterclockwise direction from the starting point to specified (X, Y) coordinates; center point of arc is specified by (I, J) coordinates
- **X0** X axis coordinate of end point = 0
- **Y1** Y axis coordinate of end point = 1
- **I-1** I coordinate of center point of arc = -1 (relative to start point)
- **J0** J coordinate of center point of arc = 0 (relative to start point)
- **F2** Feed rate is 2 inches per minute

The tool path generated by the preceding block is something like this:

Typical tool movement using circular interpolation



An equivalent movement is achieved with incremental dimensioning (G91):

```
N9G91X1Y0;SET START POINT
N10G03X-1Y1I-1J0F2
```

In this NC block, the X and Y values are the distance the tool is to move from its current position. In both cases, the I and J values are equal to the X and Y distance from the start point to the center point.

Circular Interpolation on Other Planes

To perform circular interpolation on a plane other than the X, Y plane, use a G18 code to select the X, Z plane, or use a G19 code to select the Y, Z plane. This feature is rarely used in manual part programming, but may be used by CAM systems to generate surfaces of revolution. The G17 code is used to return to the X, Y plane. An example of circular interpolation on the X, Z plane is:

```
N9X0Z0
N10G90G18G03X0Z1I0K.5F2
```

In this NC block, the X and Z values are the destination position of the tool. The I and K values are the incremental location of the center point of the curvature of motion.

Helical Interpolation Programming

Helical interpolation is performed when the axis not used in circular interpolation is commanded to move. For example (assuming a start point of 0,0,0):

```
N10G90G03X0Y1Z1I0J.5F2
```

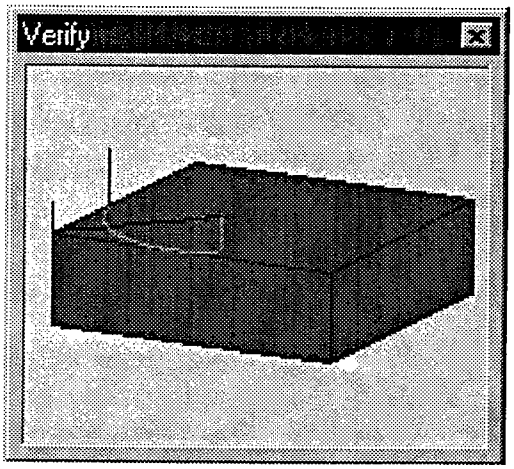
This block would cause the Z axis to move at a constant feed to Z1 while the X and Y axes move in a circular path, resulting in a helical motion. Helical interpolation works with circular motion on the X,Z and Y,Z planes as well.

Here is an example of an NC program using helical interpolation.

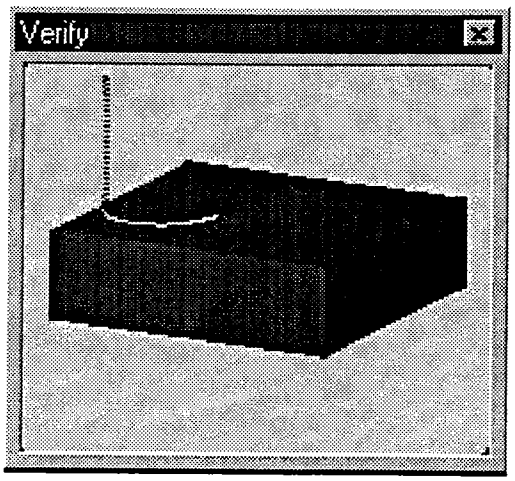
This program uses incremental arc centers. Include the % code if you have changed the Control Program default to absolute.

```
G90M03S1500
G0X0Y0Z0.070
G0X2Y2
G1Z-0.5F10
G02X0Y2Z0I-1J0F10
M02
```

In the example program, the tool plunges into the workpiece then makes the helical interpolation move to the back corner of the stock (X0Y2Z0).



Centerline View



Solid View

Rapid Traverse Programming

On the Machining Center, the rapid traverse code (G00) can move the tool at the maximum available feed rate (200 ipm) to specified coordinates. Rapid traverse is used to reposition the tool before ending a program, or in preparation for the next cut.

WARNING

The tool should not be engaged in a cutting operation while traversing to a new location!

Rapid traverse can be used for all tool positioning motions. This will reduce the run time for the part program. The G00 code remains in effect until linear (G01) or circular (G02, G03) interpolation is again specified. Linear or circular interpolation resumes at the feed rate last specified prior to the rapid traverse motion(s) unless you specify a new feed rate.

Here's a sequence of typical NC blocks using rapid traverse:

```
G90G01X1F2; MOVE IN A STRAIGHT LINE TO X = 1 AT 2 IPM
G00X2; RAPID TRAVERSE TO X=2
X3; RAPID TRAVERSE TO X=3
G01X4; MOVE IN A STRAIGHT LINE TO X=4 AT 2 IPM
```

Canned Cycle Programming

Canned cycle commands allow you to perform drilling operations by specifying just a few codes. They are typically used for repetitive operations to reduce the amount of data required in an NC program. Canned cycle codes are retained until superseded in the program by another canned cycle code. The supported canned cycles codes are:

- **G80** Canned cycle cancel
- **G81** Straight drilling
- **G82** Straight drilling with dwell at bottom
- **G83** Peck drilling
- **G85** Boring cycle
- **G86** Boring cycle with spindle off (dwell optional)
- **G89** Boring cycle with dwell

These codes are used in conjunction with canned cycle codes:

- **G98** Rapid to initial position after canned cycle complete; this is the system default
- **G99** Rapid to point R after canned cycle complete
- **Q Code** Specifies the depth of cut. In peck drilling each peck uses the same Q value. The Q value is always positive. If a negative value is specified, it is converted to a positive value.
- **R Code** Used for specifying a starting reference point for peck drilling. The point can be at the material surface or at another reference point.

Using G80

To cancel a canned cycle, use the G80 code. This code cancels the currently running canned cycle and resumes normal operation. All other drilling data is cancelled as well. You can also cancel canned cycles by using a G00 or G01 code; a G80 is automatically performed before the G00, G01, G02, or G03.

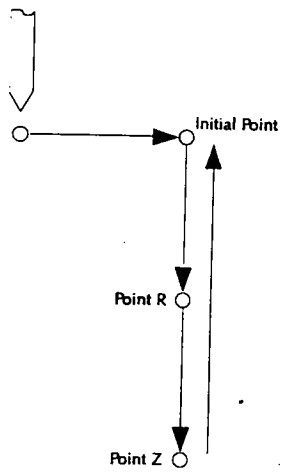
Using G81

The G81 code performs straight drilling operations. By specifying an R value of zero, the tool will return to the initial point after drilling to point Z. Here is a sample G81 program.

```
G0X1Y1Z.1;RAPID TO 1, 1, .1
G81G98Z-.5R0F2;DRILL TO DEPTH OF -.5, RAPID TO INITIAL POINT
G80;CANCEL CANNED CYCLE
M2;END PROGRAM
```

This program will generate tool motions similar to this:

If we specified a G99 here instead of a G98, the tool would rapid to point R instead of the initial point.



More than one canned cycle can be accomplished by specifying only X and Y coordinates. For example:

```
G0X1Y1Z.1;RAPID TO 1, 1, .1
G81G98Z-.5R0F2;DRILL TO DEPTH OF -.5, RAPID TO INITIAL POINT
X.5Y1;PECK AT NEW X,Y COORDINATES
X.25Y1;PECK AT NEW X,Y COORDINATES
G80;CANCEL CANNED CYCLE
M2;END PROGRAM
```

Running a Sample NC Program

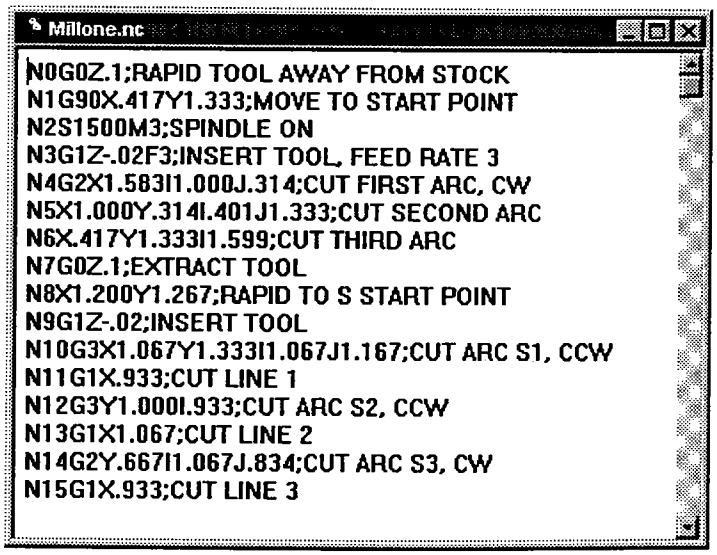
When you installed the Control Program, an NC part program file named MILLONE.NC was copied into the BENCHMAN 4000 directory along with the other files. This program is meant to machine a 3" x 2" x 1.5" piece of machinable wax. You will be using this file to create your first workpiece on the machining center.

WARNING

Do not attempt to operate the BENCHMAN 4000 without reviewing all of the safety precautions set forth in the Reference Guide: Section J.

Open MILLONE.NC

1. Select the Open command from the File Menu, or click on the Open button on the Standard Tool Bar. The Open dialog box appears.
2. Double-click on MILLONE.NC, or click on the filename and then click on the Open button. The edit window for MILLONE.NC appears.



Adjust the Verify Settings

After loading the NC program, you need to adjust the Verify Settings specifically for the part you are about to machine. To view the Verify Setup dialog box, double click on the Verify window. You may also select Verify from the Program Menu, or select Verify from the Standard Toolbar then click on the Verify Settings button.

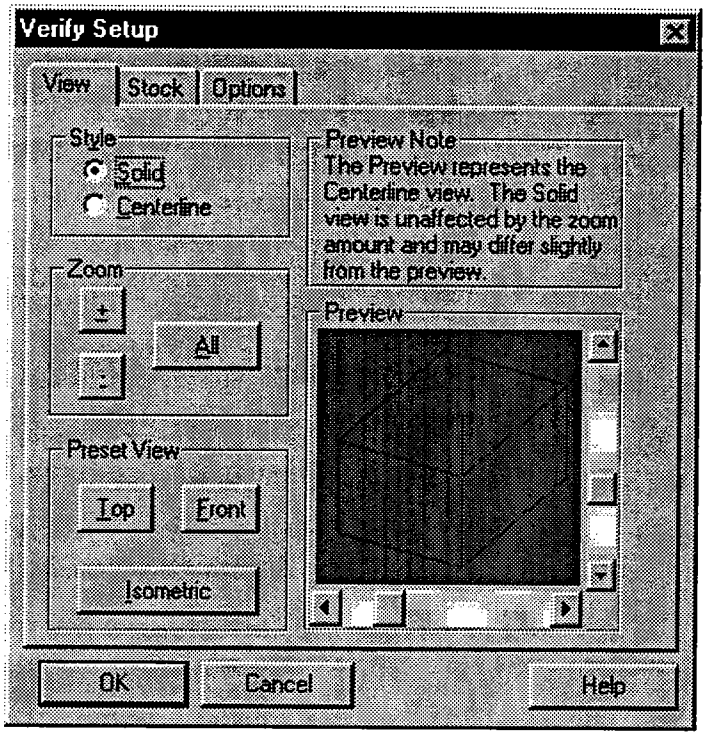
The Verify Setup dialog box appears.

The Verify Setup dialog box allows you to alter the viewpoint of the tool and workpiece in the Verify Window.

The View panel allows you to:

- Alter the Style (Solid or Centerline)
- Zoom in, zoom out or fill the window by selecting All
- Look at the workpiece in two or three dimensions

You can also alter the view of the part by adjusting the sliders on the Preview box.



Adjust the View

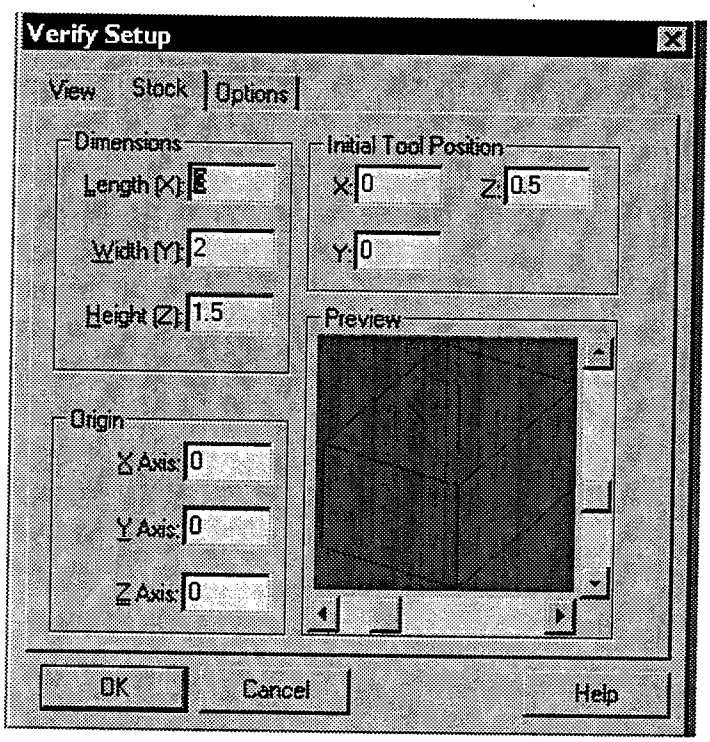
1. Select the View tab.
2. Select either Solid (for a solid representation), or Centerline (for a centerline representation of the tool and workpiece) from the Style area.
3. Select Isometric from the Preset View area for a three dimensional view of the part.

Adjust the Stock

1. Select the Stock tab.
2. Enter the stock Dimensions for the MILLONE.NC program. The stock dimensions are X=3", Y=2" and Z=1.5".
3. Set the Initial Tool Position to X=0, Y=0 and Z=0.5.
4. Set the point of Origin to zero on all three axes.
5. Select OK. The dialog box closes, and the shape of the workpiece in the Verify Window changes.

The Stock panel allows you to:

- Enter the dimensions of the workpiece
- Set the initial position of the tool
- Set the point of origin for the workpiece



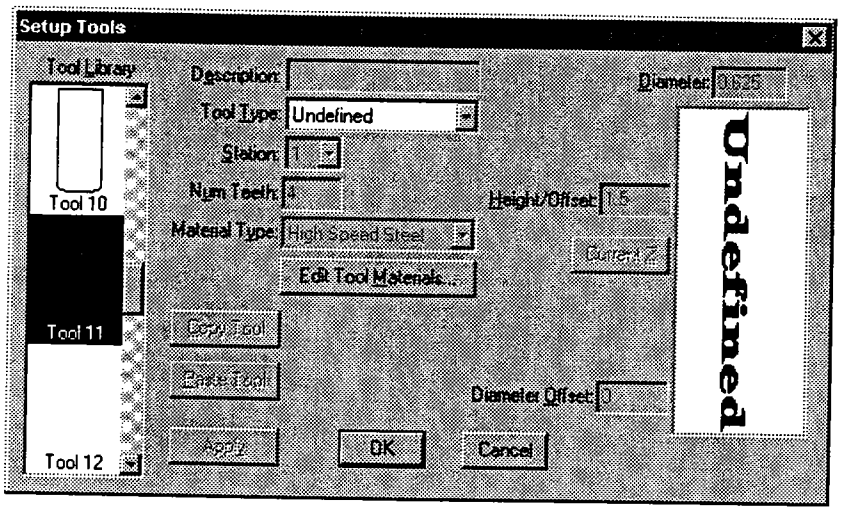
Define the Tool

To machine this part, you will use an 1/8" HSS end mill. You will use the parameters for this particular end mill for the tool path verification as well. To define the tool parameters, first add the tool to the tool library, then select the tool for verification.

Add the Tool to the Library

1. Select Setup Library from the Tools Menu. The Setup Tools dialog box appears. There is a 1/8" end mill already defined. However, our purpose is to learn to use the system, so we will create another 1/8" end mill.

To define a new tool, select an undefined tool, as we have here, then select a Tool Type.

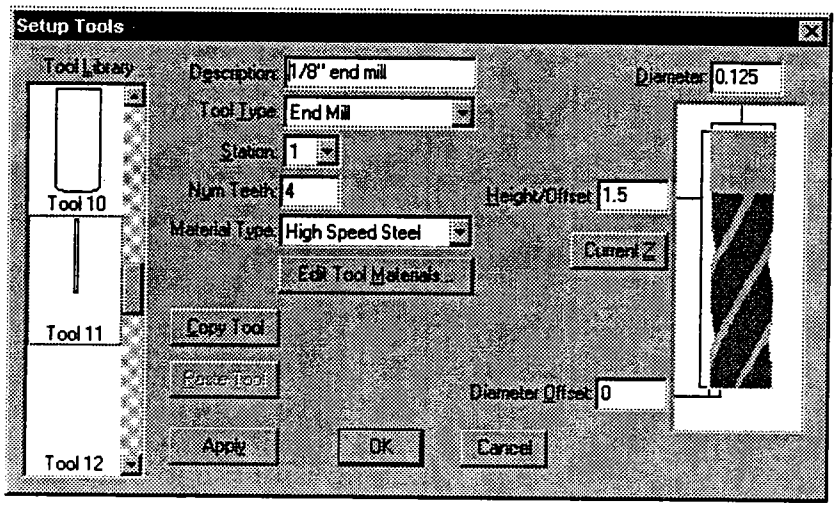


2. In the Tool Library scroll box, scroll down to tool 11, which is undefined at this point. Highlight Tool 11 by clicking on it with your mouse.
3. In the Tool Type pull-down list, select End Mill.
4. Enter "End Mill" in the Description box.
5. Enter 0.125 in the Diameter box.

6. Click on the Apply button. You have just defined a new tool in the library. From now on, whenever you need an 1/8" HSS end mill it will be there.
7. Click on OK to exit the Tool Library.

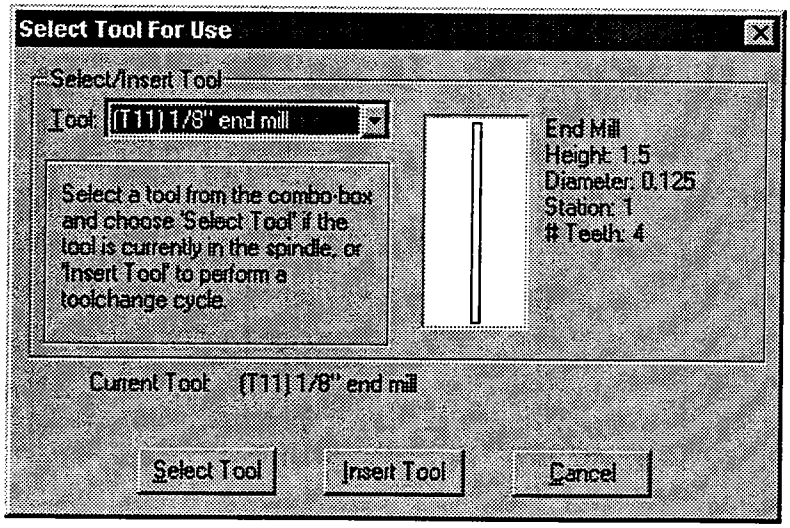
Here is the newly defined tool, an 1/8" HSS end mill that has four teeth and a height offset of 1.5 inches.

These are the same tool parameters you will use when actually machining the MILLONE.NC part.



Select the Tool for Verification

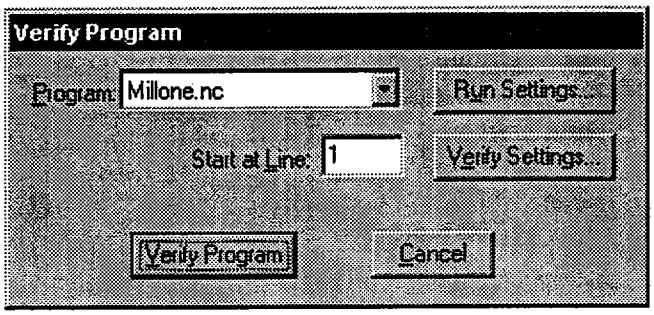
1. Select Tool from the Tools Menu. The Select Tool for Use dialog box appears.
2. Select Tool 11 from the pull-down list.
3. Click on the Select Tool button.



Verify MILLONE.NC

Tool path verification allows you to check for programming errors before actually running the part program on the Machining Center.

1. Select Verify from the Program Menu. The Verify Program dialog box appears. The default starting line for the program is Line 1. When verifying a program for the first time, you should begin at the first block.



2. Click on the Verify Program button, then watch the Verify Window. You will see MILLONE.NC executed on the graphic workpiece.