Note: these directions are based on a normal operator position in front of the machine.

In addition to linear movement, the production of a part may also require rotary movement which is provided by the use of ancillary equipment such as rotary tables and indexers. These movements are also controlled via the machining program and are identified by the letters A, B, and C as illustrated in Figure 8.22.

DATUMS

There are two datums involved in CNC machining that concern the part programmer.

The first of these is the program datum, which is established by the programmer when writing the program. This datum is at the intersection of the *X*, *Y*, and *Z* axes when milling, and at the intersection of the *X* and *Z* axes when turning. In both cases it is given the numerical identity of zero. The actual position of this datum in relation to the workpiece is optional, although there are certain factors to be taken into consideration.

The program datum is, in effect, the point from which the slide movements in each axis will be dimensionally related. It should be noted that part setup on a machine tool must allow for the proper machine zero/program zero relationship. When program data are stated in absolute terms (see below), all subsequent moves will also be dimensionally related to that point.

The second datum that concerns the part programmer is the machine datum. This datum is a set position for the machine slides where the axes intersect, and it has a numerical identity of zero within the control system.

On some machines the machine datum is a permanently established position (referred to as a fixed zero) and cannot be altered, although it can be repositioned on a temporary basis via a zero "shift" or "offset" facility. On other machines a new datum can be established anywhere within the operating pocket of the machine, a facility referred to as a "floating zero."

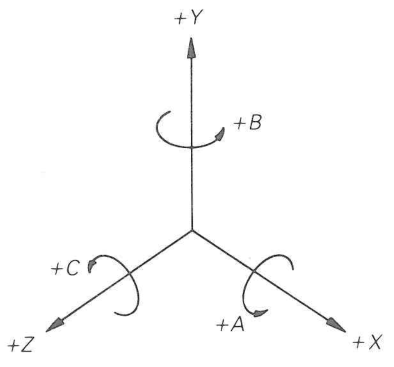


Figure 8.22 *Identification of rotary movements.*

Clearly, there must be some correlation between the machine datum and the program datum when setting the machine if the programmed slide movements are to achieve the intended effect, and the practicalities involved are discussed in more detail in Chapter 6.

ABSOLUTE AND INCREMENTAL POSITIONAL DATA

Once the direction of movement has been established, the distance moved by the machine slide to bring it to a desired position has to be defined dimensionally. This is achieved by the use of linear coordinates, with the dimensions being stated in absolute or incremental terms.

A third method is sometimes used, involving the use of polar coordinates. This requires a distance (the radius) stated in relation to a defined point and at an angle stated in relation to a datum axis. It generally requires the control system to include a special programming facility.

Absolute dimensional definition requires all slide movements to be related to a predetermined zero datum.

Incremental dimensional definition requires each slide movement to be related to the final position of the previous move.

Figure 8.23(a) shows the details of a turned component. The intersection of the spindle centerline and the face of the work is the program zero datum. Assume that a final trace of the component profile is to be programmed.

The dimensional definition in absolute and incremental values that would be required to define slide movement is shown in Figures 8.23(b) and 8.23(c) respectively.

In Figure 8.24(a) the details of a milled component are given. Absolute and incremental dimensional values required to program the machining of the slot are shown in Figures 8.24(b) and 8.24(c) respectively.

Earlier programming languages required dimensional data to be stated in *either* absolute or incremental terms. Modern controllers often provide a "mix and match" facility that permits the use of both within the same program, and even within the same data block. The distinction is achieved by the continued use of the G90 (absolute) and G91 (incremental) preparatory function codes, or X, Y, and Z word addresses for absolute values and U, V and W for incremental values. The use of the "G" code for changing from absolute to incremental is much more common.

CIRCULAR INTERPOLATION

Circular interpolation allows for programming a machine to move the cutting tool in a circular path with a uniform arc, with only a few commands. Circularlike motion is achieved through the machine controls capability to cal-

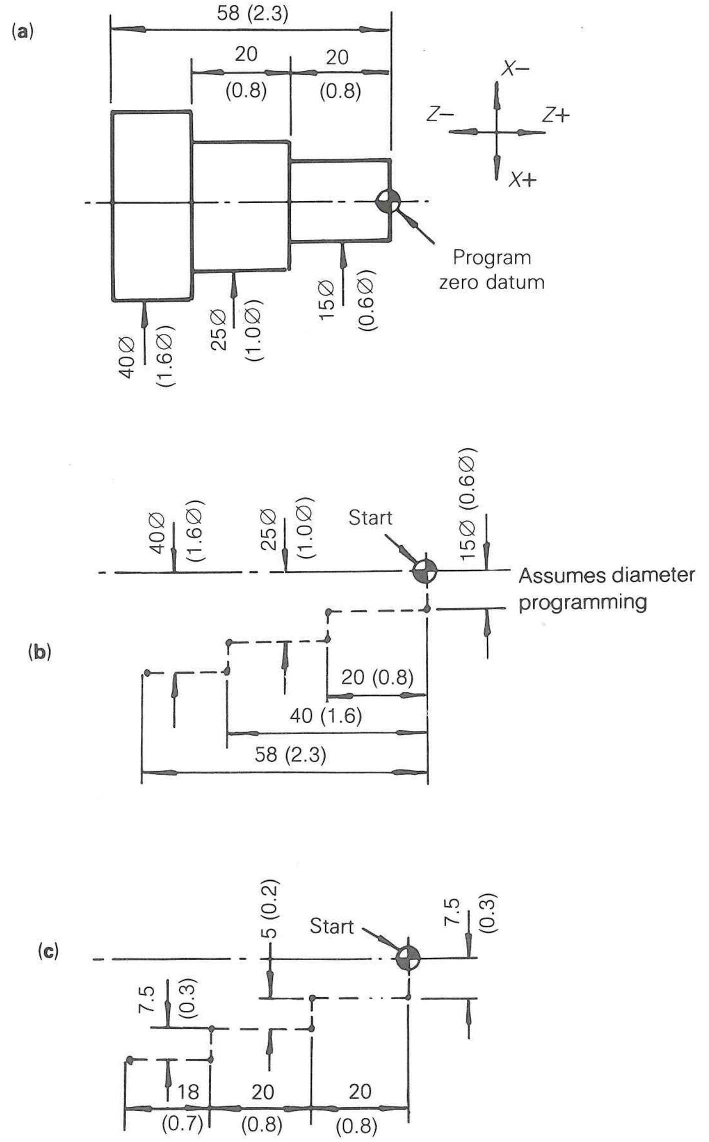


Figure 8.23 (a) *Component detail*, (b) *absolute dimensions, and* (c) *incremental dimensions. (Inch units are given in parentheses.)*

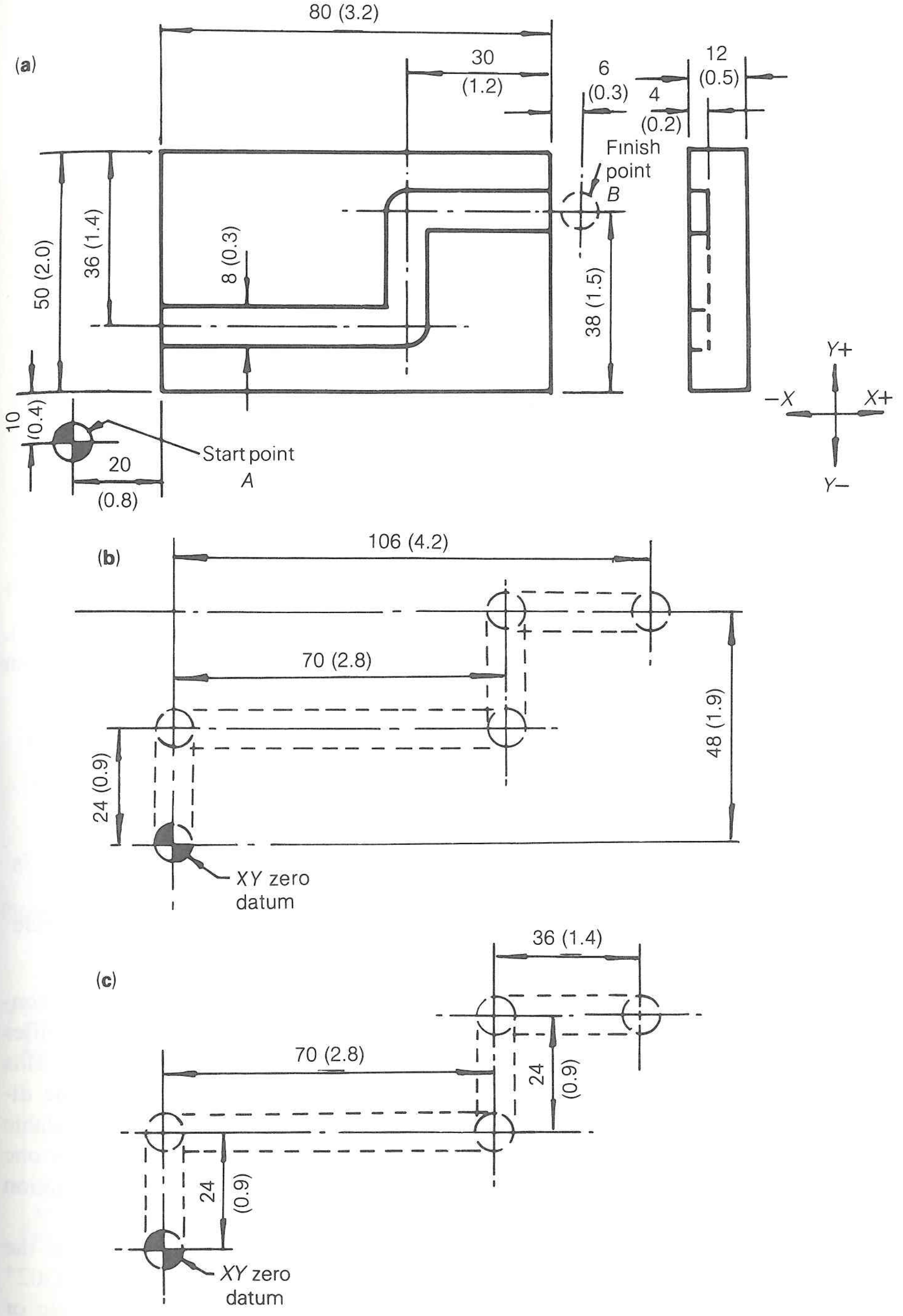


Figure 8.24 (a) *Component detail*, (b) *absolute dimensions, and* (c) *incremental dimensions. (Inch units are given in parentheses.)*

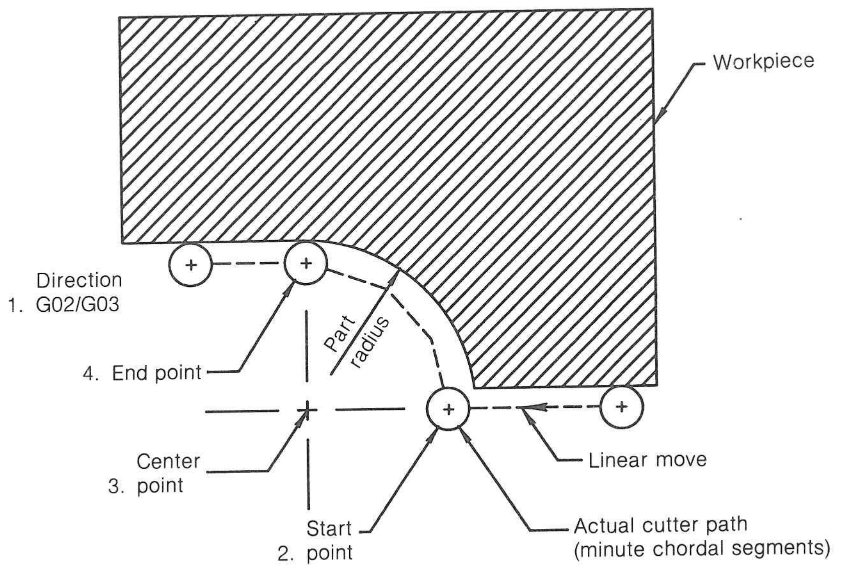


Figure 8.25 *The four basic elements of circular interpolation.*

culate minute cordal movements between commanded points (Figure 8.25). The basic elements of circular interpolation will deal with the following four items:

1. *Direction of rotation*—The direction of rotation that the cutter path will follow around the circle or arc segment.
2. *Start point*—The actual beginning point of the circle or arc segment where the cutter has previously been positioned.
3. *Center point*—The point of rotation in which the circle or arc segment is developed around.
4. *End point*—The finish point of the cutter path used to generate the circle or arc segment.

It was explained earlier in the text that circular arc programming on conversational MDI systems has been reduced to a simple data entry that specifies the target position, the value of the radius, and the direction of rotation. This simple method of defining circular movement in a single block with the direction of rotation being defined by the appropriate G code is also available on some word address systems (see method 3), but many systems employ one of two slightly more complex techniques of quadrant circular interpolation (methods 1 and 2).

Common to all programming systems is the need to determine whether the relative tool travel to produce a particular arc is in a clockwise (CW) "G02" or counterclockwise (CCW) "G03" direction, and the location of the arc or circle center. The following approaches are usually helpful:

1. For turning operations look down onto the top face of the cutting tool. (For inverted tooling this involves looking up at the tool from below.)
2. For milling operations look along the machine spindle centerline toward the surface being machined.

It should be noted that this technique does not always correspond with the definition adopted by the control system's manufacturer. A simple trial program entered into the machine will clarify the situation if it cannot be determined from the machinery manuals.

Determining clockwise and counterclockwise direction on milling machines with multiple plane circular interpolation capability can be done using the following rule and Figure 8.26.

Clockwise and counterclockwise are determined by looking at the plane of the circle from the positive end of the coordinate axis normal to the plane of the circle.

The arc center definition for circular interpolation is also a standard requirement. It informs the control of the point of rotation for an arc along standard axes from the arc start point or program datum. Use the following definitions and Figure 8.27 to determine proper arc center addresses needed for various planes of rotation.

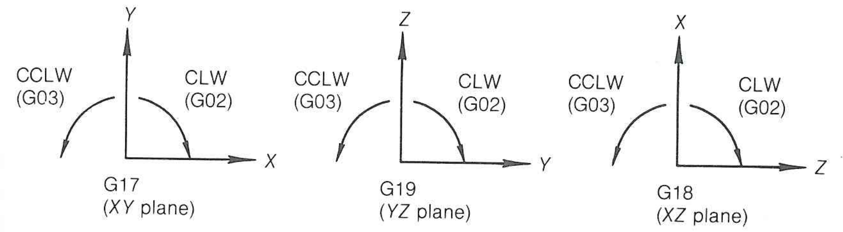


Figure 8.26 *Determining clockwise and counterclockwise directions.*

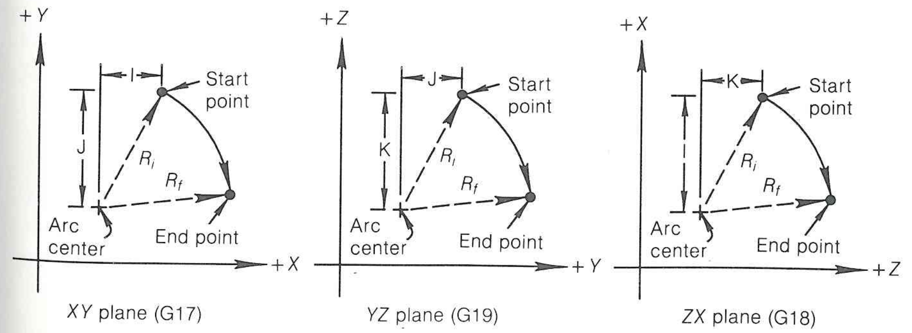


Figure 8.27 *Arc centers.*

Use of I, J, and K to Specify the Arc Center

I is the *increment* along the *X* axis from the start point of the arc to the arc center.
J is the *increment* along the *Y* axis from the start point of the arc to the arc center.
K is the *increment* along the *Z* axis from the start point of the arc to the arc center.

Note: Refer to your machinery manual to determine whether or not positive or negative signs will be required on the I, J, K values.

QUADRANT CIRCULAR INTERPOLATION

Quadrant circular interpolation allows up to 90° of arc to be programmed as long as the entire segment does not cross quadrant lines; refer to Figure 8.28

For any circle there are four quadrants that are created by axis lines which cross at the center of the circle. These axis lines are parallel to the part coordinate axes. No circular block may cross either of these axes. Starting at any point on the circle, it can be seen that the maximum number of circular controller data blocks to cut a 360° arc is five.

An individual circular controller data block cannot cross an axis line. To continue into the next quadrant, terminate the present circular controller data block and program another circular controller data block, starting on the axis. This procedure is quite straightforward.

A minimum of four sets of dimensional data are required to complete a full circle with up to four dimension words per block. Two dimension words are required to define the arc and one or two are required to define the arc center. Arc center values of zero are normally not required. Machines that are capable

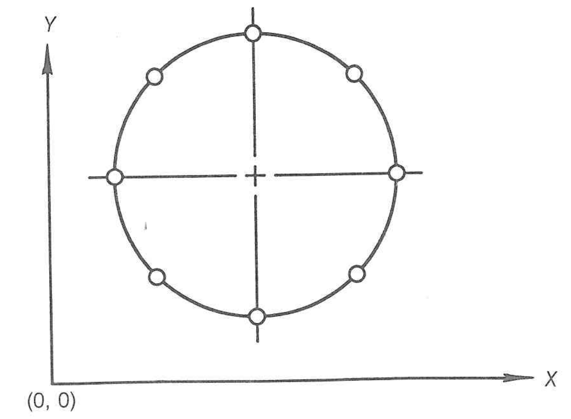


Figure 8.28 *Circle quadrants.*

of either quadrant or 360° circular interpolation will require a modal G code to be programmed (G74 quadrant; G75 360°). *Note:* For exact programming format and arc size limits refer to your machinery manual.

The two arc center address programming techniques previously referred to can be described using the component detail shown in Figure 8.29, and assuming that the last programmed move has brought the cutting tool to the start point indicated.

Method 1

1. The target or finish point of the arc is dimensionally defined, using *X*, *Y*, or *Z* values, in relation to the program datum when using absolute mode, or to the finish position from the previous move when using incremental mode.
2. The center of the arc is dimensionally defined in relation to the start point using I, J, and K values measured along the *X*, *Y*, and *Z* axes respectively.

Using this method the arc shown in Figure 8.29 would be programmed as follows:

Inch:

In absolute terms:	G02 X1.6 Z1.6 I0 K.8 (Diameter programming of lathe part.)
In incremental terms:	G02 X.8 Z-.8 I0 K.8

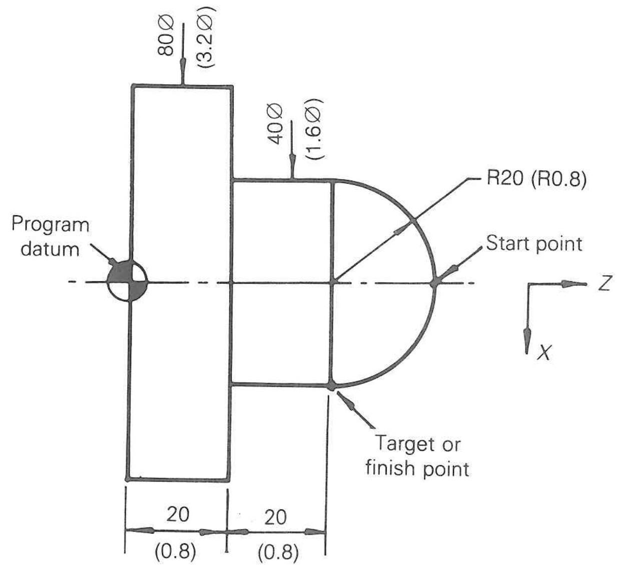


Figure 8.29 *Component detail. (Inch units are given in parentheses.)*

Metric:

In absolute terms:	G02 X40 Z40 I0 K20 (Diameter programming of lathe part.)
In incremental terms:	G02 X20 Z-20 I0 K20

Note that the I code has no value because the center and start point of the arc are in line with each other. In practice, when a value is zero, it is not entered into the program. In this method of arc definition the I, J, and K values are unsigned.

Method 2

The second method of word address arc programming differs in the way the arc center is defined. The following data are required:

1. The target or finish point of the arc is dimensionally defined, using X, Y, and Z values, in relation to the program datum when using absolute mode, or to the finish position from the previous move when using incremental mode.
2. The center of the arc is dimensionally defined in relation to the program datum using I, J, and K values measured along the corresponding X, Y, and Z axes, respectively, in absolute mode. The arc center is still defined from the previously defined point in incremental mode.

Using this second method of programming the arc shown in Figure 8.30 would be programmed as follows:

Inch:

In absolute terms:	G02 X1 Y2.4 I1 J1.4
In incremental terms:	G02 X1 Y1 I1 J0

Metric:

In absolute terms:	G02 X25 Y60 I25 J35
In incremental terms:	G02 X25 Y25 I25 J0

It is possible when using this approach for the I, J, and K values to be negative, as illustrated in Figure 8.31. These values are, therefore, signed plus or minus.

The two arc programming methods described will cater for movement within one quadrant only with each block of program data. Thus programming a complete circular move would require four blocks of data minimum. Similarly, blending arcs would require a separate block of data for each quadrant involved. This latter situation is illustrated in Figure 8.32; the first block would take the tool from point A to point B, and the second block would continue the movement from point B to point C.

When the start or stop points or both do not coincide with an *X*, *Y*, or *Z* axis, that is the arc is not exactly 90° or a multiple of 90°, it will be necessary to make a series of calculations.

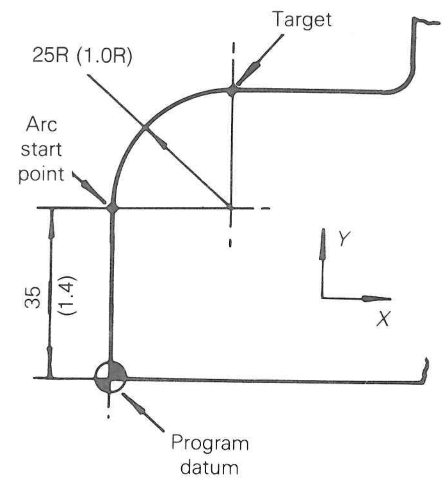


Figure 8.30 *Milled component detail. (Inch units are given in parentheses.)*

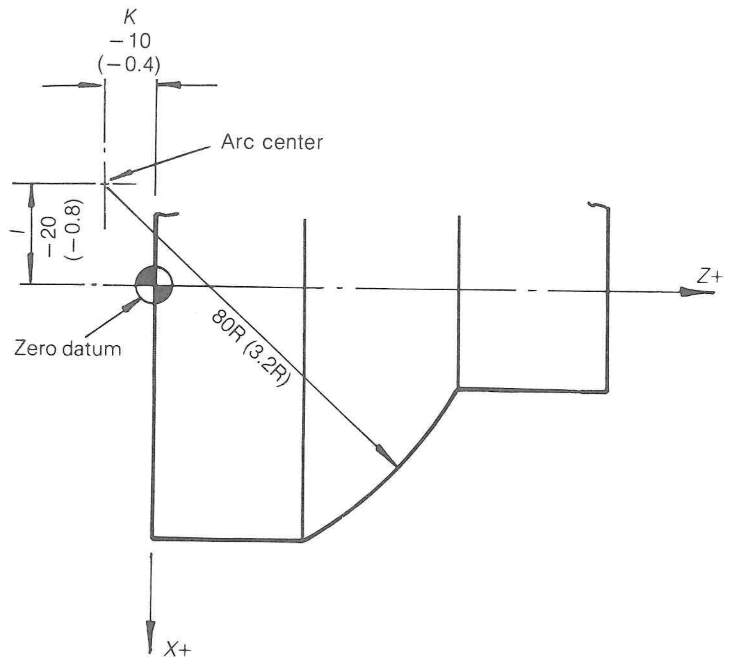


Figure 8.31 *Circular interpolation: negative I and K values for an absolute program. (Inch units are given in parentheses.)*

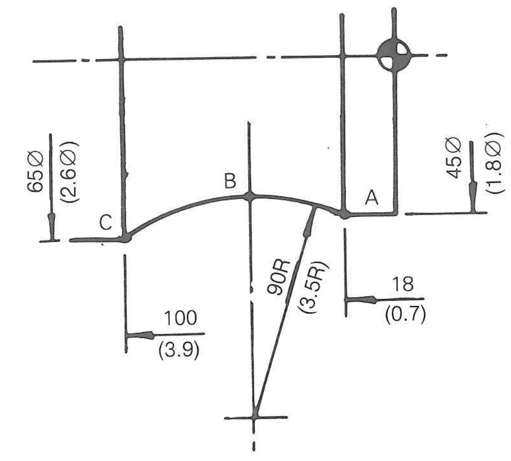


Figure 8.32 *Profile detail requiring two arcs to be programmed. (Inch units are given in parentheses.)*

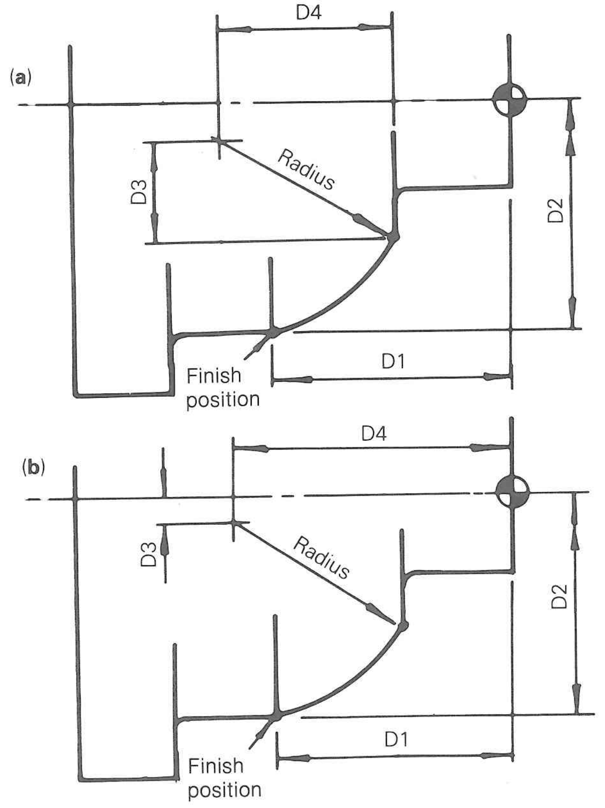


Figure 8.33 *Calculations for arcs less than 90°: (a) method 1; (b) method 2. (Inch units are given in parentheses.)*

Consider the part detail shown in Figure 8.33(a) and again in Figure 8.33(b), each diagram relating to an arc programming method as indicated. Whichever of the two methods is used, the finish arc position indicated by the dimensions D1 and D2 will have to be defined dimensionally. It will also be necessary to calculate the additional dimensions indicated on each drawing as D3 and D4, these latter dimensions being expressed in the part program as I and K values. A number of circular interpolation examples follow.

PROGRAMMING EXAMPLES

The circle in Figure 8.34 will be used to illustrate programming various arcs, clockwise and counterclockwise, absolute and incremental. (All units in the following examples are in inches.)

Example 1 (Method 1)

Go from P2 to P5 clockwise (absolute) (G74 default).

Signed I and J	Unsigned I and J
G17	G17
G90	G90
G2X3Y2I-.7071J-.7071F10	G2X3Y2I.7071J.7071F10
X2Y1I-1	X2Y1I1
X1Y2J1	X1Y2J1

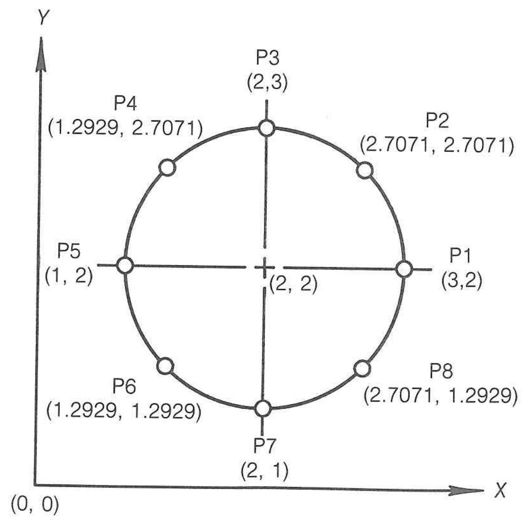


Figure 8.34 *Example of programming arcs.*

Example 2 (Figure 8.35)

Go from P2 to P5 counterclockwise (incremental) (G74 default).

Signed I and J	Unsigned I and J
G17	G17
G91	G91
G3X-.7071Y.2929I-.7071J-.7071F10	G3X-.7071Y.2929I.7071J.7071F10
X-1Y-1J-1	X-1Y-1J1

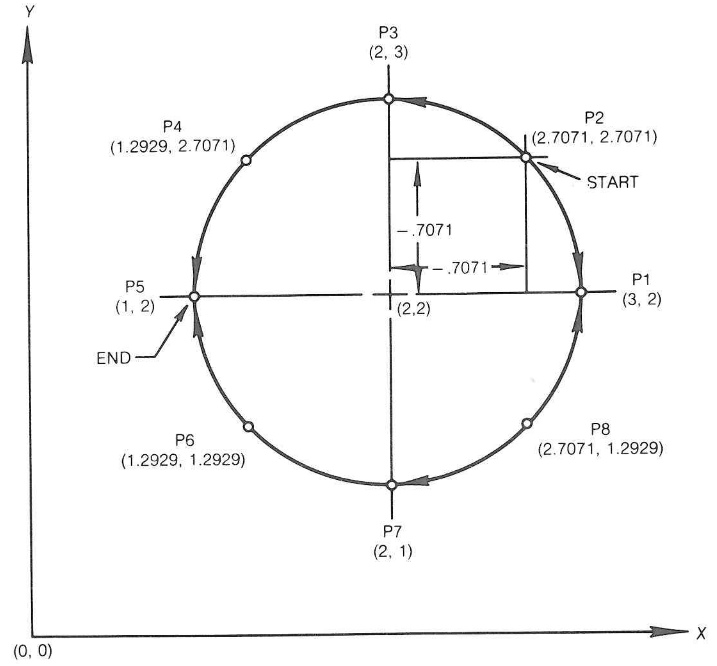


Figure 8.35 *Examples 1 and 2.*

Example 3 (Figure 8.36)

Go from P4 to P7 clockwise (incremental) (G74 default).

Signed I and J	Unsigned I and J
G17	G17
G91	G91
G2X.7071Y.2929I.7071J-.7071F10	G2X.7071Y.2929I.7071J.7071F10
X1Y-1J-1	X1Y-1J1
X-1Y-1I-1	X-1Y-1I1

Example 4 (Method 1) (Figure 8.36)

Go from P4 to P7 counterclockwise (absolute) (G74 default).

Signed I and J	Unsigned I and J
G17	G17
G90	G90
G3X1Y2I.7071J-.7071F10	G3X1Y2I.7071J.7071F10
X2Y1I1	X2Y1I1

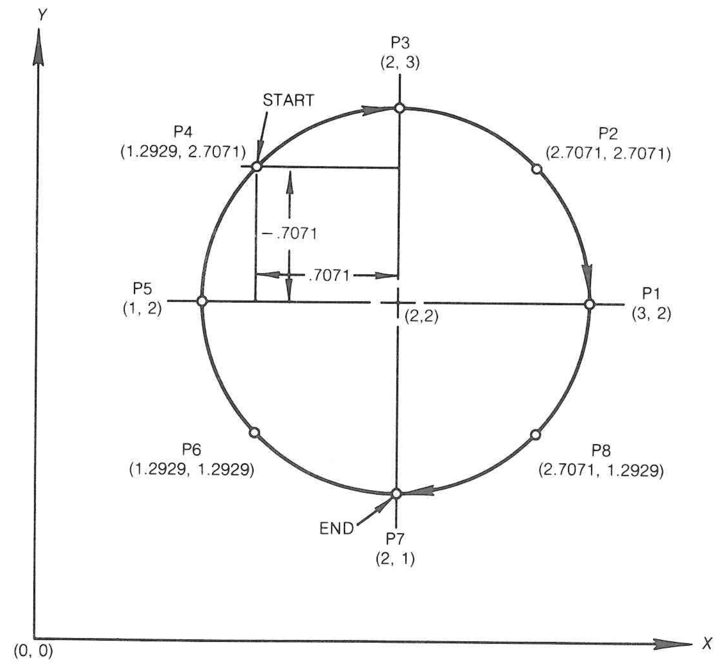


Figure 8.36 *Examples 3 and 4.*

Method 3

Single block or 360° circular interpolation is now available on more sophisticated CNC controls. Single block circular interpolation allows up to 360° of circular movement to be programmed in a single block of information when the modal G75 code is active. All previous circular interpolation command words defined are still used to determine rotation, finish/end position, and arc center. It should be noted, though, that in this form of circular interpolation the I, J, and K combination used must be signed to determine the circle's center.

Another difference that will be noted in this method of programming is if a full 360° of circle is to be produced, the starting and ending points of rotation will be identical. When identical starting and ending points occur, the end point of the arc does not have to be programmed, only the rotation and arc center information are required; refer to the examples with Figure 8.37.

Example 5 (Method 3) (Figure 8.37)

Go from P3 to P3 clockwise.

Incremental	Absolute
G17	G17
G91	G90
G75	G75
G2I0J-1F10	G02I0J-1F10

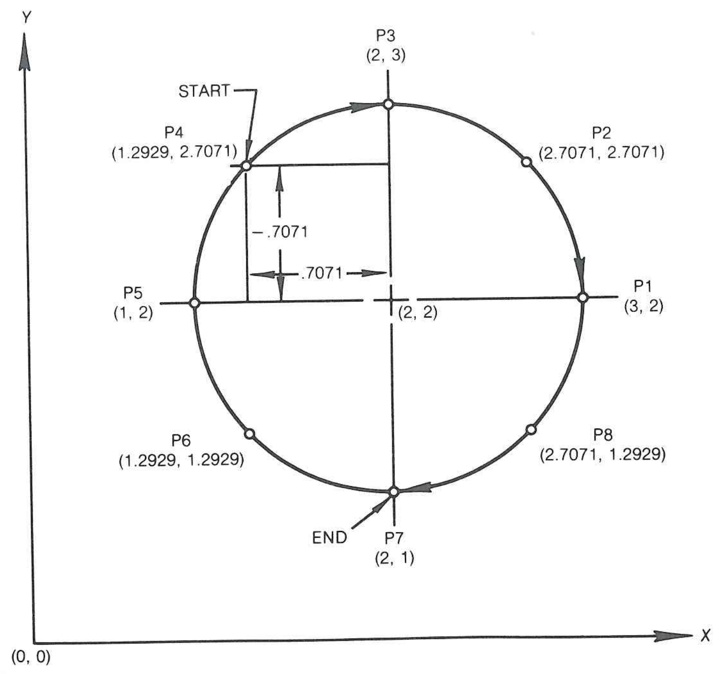


Figure 8.37 *Examples 5 and 6*

Example 6 (Method 3) (Figure 8.37)

Go from P4 to P7 clockwise.

Incremental	Absolute
G17	G17
G91	G90
G75	G75
G2X.7071Y-1.7071I.7071J-.7071F10	G2X2Y1I.7071J-.7071F10

TOLERANCED DIMENSIONS

It is often the case that dimensions on drawings are toleranced, thus permitting a higher and lower limit. Since it is only possible to enter one value into the control unit, it is logical that this should be the middle or mean value of the tolerance band.

REPETITIVE MACHINING SEQUENCES

There are a number of repetitive sequences that are commonly used when machining a variety of components like drilling a hole. Other less common sequences are also repetitive, but on only one particular component (hole patterns—milling path). It is helpful, since it reduces the program length and also simplifies programming, if such a sequence can be programmed just once; it is then given an identity so that it can be recalled into the program as and when required.

Repetitive machining sequences can be generally classified as follows:

(a) Canned or fixed cycles that are a built in feature of the machine control system.
(b) User or programmer defined routines to suit the particular job in hand (custom canned cycles, program macros).

The facility for the programmer to devise special routines may be restricted, especially on small training machines. However, even the most simple system will usually include one or two canned cycles. The controls fitted to advanced machines will have many available.

CANNED CYCLES

In a text of this nature it would be impossible to deal in detail with all the canned cycles that are available, but the review that follows will convey a good impression of the range currently in use and likely to be encountered. More specifically, the student is advised to make a careful study of the programming methods and techniques associated with the control system he or she will be using. In this respect a close examination of the examples found in the programming manuals will be found to be helpful. The point to remember is that the use of canned cycles is an aid to programming efficiency and accuracy, and they should be used whenever possible. You will now notice that the previously mentioned special cycles for conversational programming are not necessarily unique to that form of programming. Word address programming has many of the same options that are used through the use of special "G" codes or possibly a switch on the control rather than a push button.

Fixed or canned G code cycles (G81–G89) were developed, as stated, to simplify the programming of hole-making operations with repetitive motions at each hole location. These codes are normally modal and must be cancelled with a G80 code (linear rapid transverse mode) before other program movements are commanded. The normal action of the canned cycles mode is to position the programmed nonspindle (tool penetration) axes, including the rotary table axes on milling machines, with the (*Z*) coordinate axis remaining stationary. The tool will then rapid the *Z* axis to the *R* or tool-clearance plane preceding the part surface and feed at the programmed rate to *Z* depth. After feeding the tool to depth, the tool will be retracted from the hole automatically as required by the cycle. Note: different codes will cause various forms of retraction (refer to your programming manual for specific retraction action). Once all of the necessary modal information is programmed (G code, depth, clearance position, spindle speed, feed, and spindle direction), additional holes can be produced by programming new axis or axes positions until the canned cycle is cancelled. Most canned cycles will pick up the last programmed spindle speed/direction and feedrate if they are not initiated in the canned cycle block of information.

Perhaps the most widely used machining sequence is that of drilling a hole, and there are few controls that fail to cater to this requirement by including a canned cycle. Indeed, with word address programming, early attempts were made to standardize a drill cycle. That this was quite successful is evident by the fact that the use of G81 for the purpose is as common as the use of G00 and G01 for linear movement control and G02 and G03 for circular interpolation.

There are a number of machining variations necessary in the production of drilled holes.

One of the most commonly used is the basic drilling movement, catered for by the drilling cycle illustrated in Figure 8.38. This involves a drill movement to the required depth at a controlled feed rate, followed by rapid withdrawal.

Also widely used is the intermittent or "peck" drill cycle for deep holes illustrated in Figure 8.39. This illustration shows a complete withdrawal to the *Z* axis clearance plane after each peck, but variations of this cycle provide for a smaller withdrawal that conveniently breaks the chip but does not give total

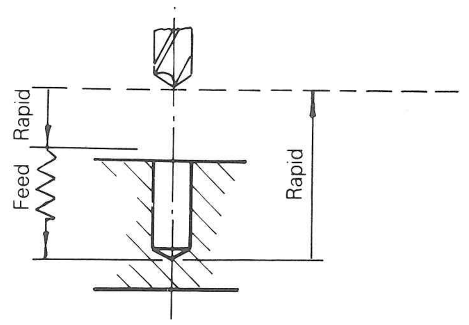


Figure 8.38 *Drill cycle.*

retraction for chip clearance. The peck depth is established with an additional *Z* axis command word, which many times is a "K" value.

A further refinement of this cycle provides for automatic variation of the peck length as the hole deepens. This is achieved by including a "multiplier" in the cycle data. For example, a multiplier of 0.8 will have the effect of reducing each peck length to 0.8 of the previous peck. To avoid the reduction continuing indefinitely a minimum peck length is also programmed.

Closely allied to the drilling cycles are those for boring and tapping. To ensure a clean surface, boring may require special or no drag line retraction, and counterboring may require the inclusion of a time dwell at the extent of cutter travel. Tapping requires that the direction of spindle rotation is reversed to allow withdrawal of the tap.

Many turning operations involve machining chamfer or radius features illustrated in Figures 8.40(a) and 8.40(b). It is possible to program a special cycle on some machines that would normally require two data blocks in just one block. Figure 8.40(a) shows the programming case which would otherwise require a *Z* axis movement followed by simultaneous movement in both the *X* and *Z* axes to produce an angle or chamfer. Figure 8.40(b) shows a similar

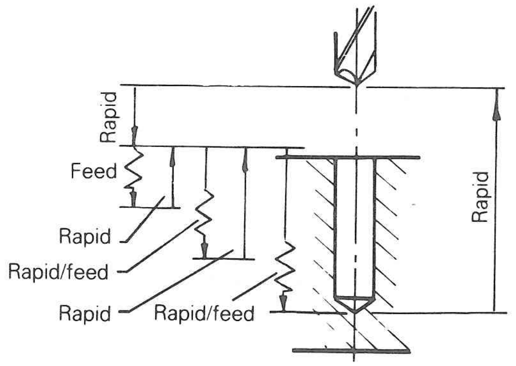


Figure 8.39 *Peck drill cycle.*

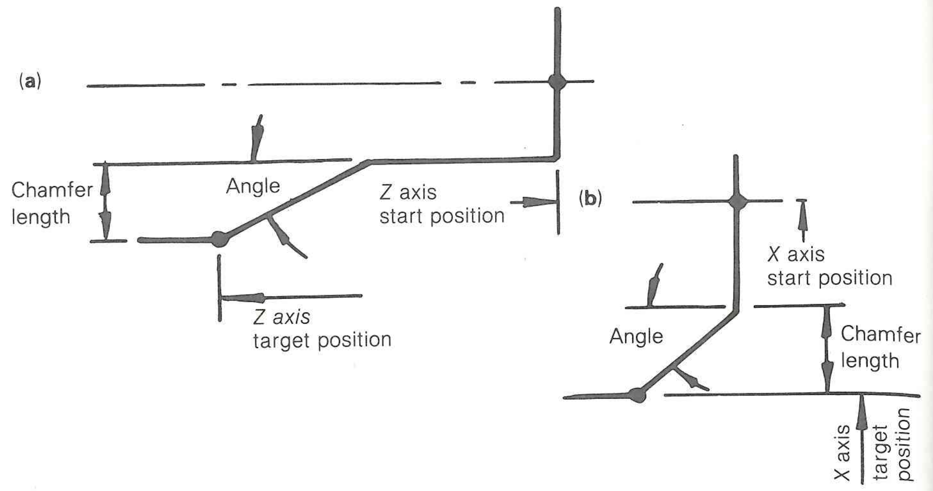


Figure 8.40 *Chamfering cycles.*

situation, but in this case the angular movement is preceded by linear movement in the *X* axis.

Very similar to the cycles described previously are those illustrated in Figures 8.41(a) and 8.41(b). Instead of a linear move being followed by an angular move, the linear moves are followed by radial movement. In both cases the radial movement must be a full 90°.

Both the chamfering and radius cycles may be automatic with little or no programming required for a standard size or there may be special G code and axis information required. Refer to your specific programming manual.

Automatic threading cycles are the counterpart of the G84 tapping canned cycle for single-point threading tools. Single-point thread cutting tools require

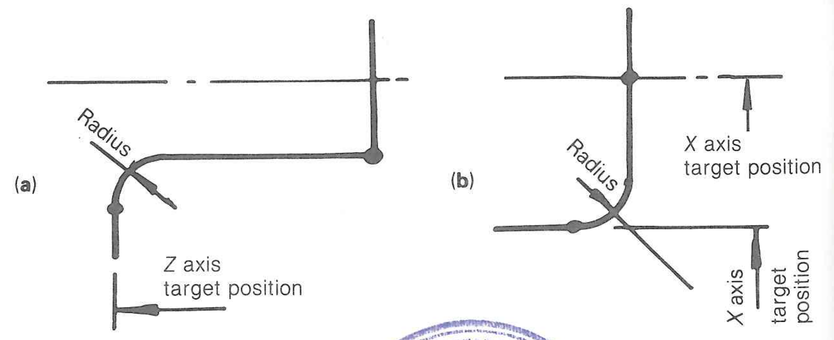


Figure 8.41 *90° arc cycles.*

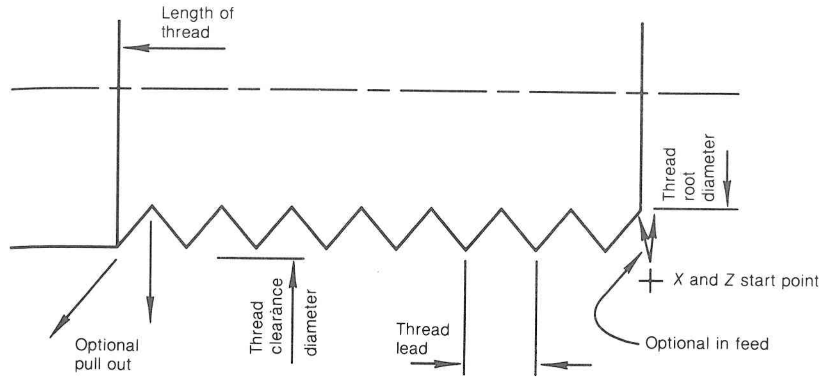


Figure 8.42 *Automatic threading cycle.*

a number of passes to produce a finished thread. Special G codes, like G33, have been developed for lathe work, which allows the programmer to call thread approach and depth, cut distance, retract to clear part, and return to start point all in one block/line of information. Various controls will handle this differently, some requiring a line of instruction for each thread pass and others allowing the entire thread cutting operation to be cut with one statement. G codes like G28 and G29 may be used to define the basic thread pass for later recall. The various codes have been developed to handle both internal and external threads as well as straight or tapered threads. Figure 8.42 shows an example of the types of program information required to make this cycle work.

USER-DEFINED ROUTINES

Canned cycles cater for the easy programming of machined features that are often required on a wide range of components. But the part programmer is often confronted with a feature that is repeated a number of times on a particular component but is found only on that component or a limited range of components. It is in situations such as this that the facility to devise a special routine for use as and when required is very helpful.

Consider the component detail shown in Figure 8.43. Along the length of the shaft is a series of identical recesses. If there is no facility to write a special program, or routine, to machine these recesses the programmer is faced with the rather cumbersome task of detailing each move necessary to machine one recess and then repeating the data for each of the others.

When preparing a routine to accomplish a specific machining task such as this, the programmer can include any of the available canned cycles that might be appropriate. For example, the profile of the shaft recess referred to could be machined using the cycle that permitted one-block programming of a linear

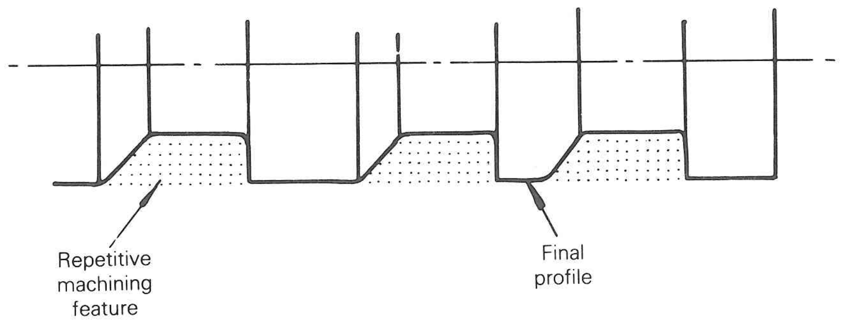


Figure 8.43 *Application of a turning subroutine.*

move in the *Z* axis folowed by an automatic chamfer or radius. When specific routines are programmed within other routines, they are said to be "nested."

Assume the component shown in Figure 8.44 has a repetitive feature as indicated that justifies using a specially devised routine to clear the recess and mill it to the required profile. Now assume that within each of these recesses there is a series of three smaller recesses as shown in Figure 8.45. Since there will be three times as many smaller recesses as larger recesses, a further specially devised routine to machine them will be justified.

The routine for machining the larger recess will therefore contain the subroutine for machining the smaller recess. The subroutine is nested within the first routine and will be activated three times during the machining of the larger recess.

It would be quite feasible for each of the smaller recesses to include a drilled hole, and this could also be produced using a canned cycle. The subroutine for the smaller recess would now include a nested drilling cycle.

There are usually some limitations regarding nesting. Some controllers permit subroutines within subroutines up to eight deep; others may accept only half this number.

Specially devised routines can also be used to control machine movements and functions not directly associated with metal cutting. For example, a special

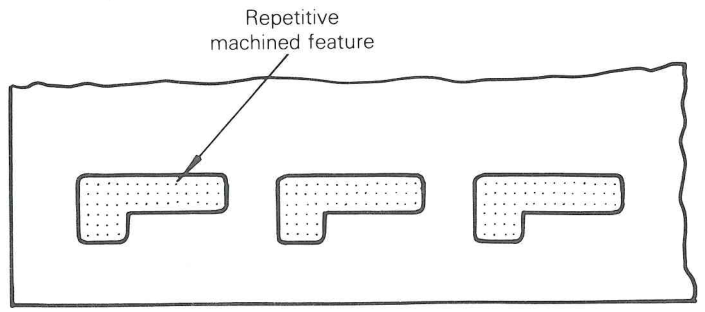


Figure 8.44 *Component detail: application of a milling subroutine.*

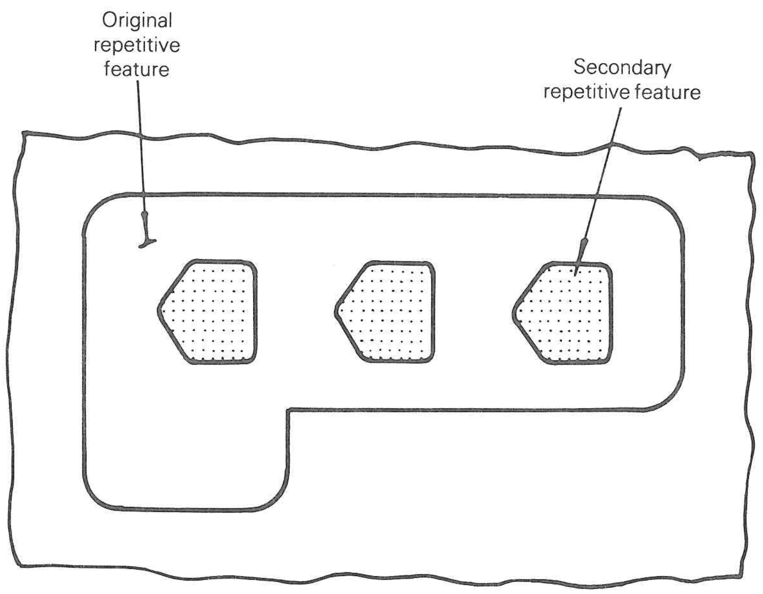


Figure 8.45 *Nesting of subroutines.*

routine can be used to establish and readily recall predetermined parameters relating to machine slide positions and programming modes which may, for safety reasons, need to be established from time to time throughout the program run. The application of a programmer-devised safety routine is included in the sample program listed in Figure 8.46.

The safety routine in this instance is determined by the blocks N10 to N70. It can be seen that, in the program, these data blocks are activated by a G25 program entry. Blocks N100 and N160 are examples where the safety routine is being called. Each time the routine appears in the program the slides return to a safe indexing position and a set of known operating modes is reestablished. This provides a basis from which the programmer can proceed with the programming of further machining operations.

Machine: Hardinge HXL Turning Center Control: GE 1050 Sample Program for Drawing Figure 8.46

```
N0010   G71                  }            Metric
N0020   G40                  }            Cancels Tool Nose Radius
                             }              compensation (TNRC)
N0030   G95                  }            Feed mm/rev
N0040   G97 S1000 M03        }  SAFETY    Spindle rev/min. CW
N0050   G00                  }  ROUTINE   Cancels G01, G02, G03, etc.
N0060   G53X177.8Z254 T00    }            Return to safe indexing
                             }              position, offsets cancelled
N0070   M01                  }            Optional stop
```

N0100	G25$P_1$10$P_2$60		Calls safety routine
N0110	T1200		Calls tool
N0120	G54X0Z3T1212		Move in X and Y with zero shift to work face and tool offset active
N0130	S2500F.1	CENTER	Spindle speed and feed
N0140	G01Z-6	DRILL	Center drill
N0150	G00Z2		Drill retract
N0160	G25$P_1$10$P_2$70		Safety routine call with optional stop turret returns to safe indexing position

Note: In the programming areas of special canned cycles and routines, controls vary drastically and your programming manual should be reviewed.

LOOPS

Some control systems provide a "loop" facility. This enables the programmer to devise a routine and to repeat that routine within the part program a specific

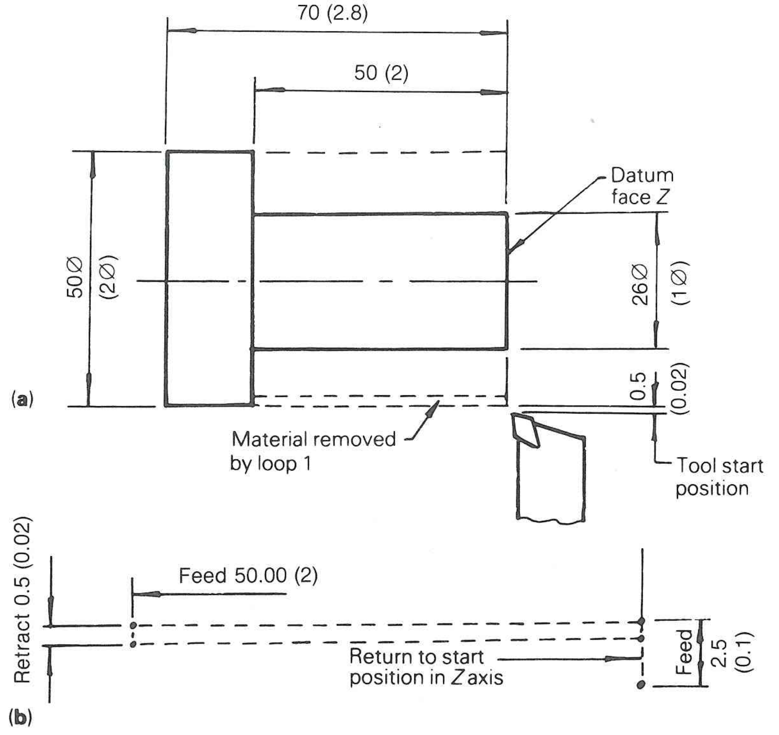


Figure 8.46 *Looping cycle:* (a) *component detail,* (b) *loop detail, repeated six times. (Inch units are given in parentheses.)*

number of times. In other words, when the program reaches the end of the routine the control will return, or loop, back to the beginning of the routine again.

Consider the component shown in Figure 8.46, which is to be reduced from 50 mm (2 in.) diameter to 26 mm (1 in.) diameter by a series of cuts each of 2 mm (0.08 in.) depth. Assuming the starting point for the tool is as shown, the tool will first move in to a depth of 2.5 mm (0.1 in.), thus taking a 2 mm (0.08 in.) depth of cut, travel along a length of 50 mm (2 in.), retract 0.5 mm (0.02 in.) and return to the Z datum, so completing a loop. It will then move in a distance of 2.5 mm (0.1 in.), feed along 50 mm (2 in.), retract 0.5 mm (0.02 in.) and return to the Z datum, and so on. The loop, including the feed rate, is programmed just once, but is repeated via the loop count data included in the main program as many times as necessary to reduce the work to the required diameter.

MACROS

A somewhat specialized type of programmer-devised routine is referred to as a "macro." This facility can be used for machining a complete component or a feature of a component that, while not standard in the wider sense, is nevertheless frequently required. For instance such a feature may commonly occur within the production schedule of a particular company. The macro is given an identity and stored within a separate macro file, or memory, and is called into use as and when required, possibly as an element within a much larger machining program.

A macro may have fixed dimensions, or it may have parametric variables which enable the dimensions to be varied to produce different versions of a basic component. This technique is referred to as "parametric programming." The parametric or variable version of macro programming is very useful when programming for a family of parts that have the same shape but vary in size.

PARAMETRIC PROGRAMMING

A parameter is a quantity that is constant in one particular case but variable in others. A simple engineering example of a parameter is the length of a bolt. One version of the bolt will have a certain length; all other versions will be identical, that is, they will have the same thread form, diameter, and hexagon head, but they will all vary in length. Thus, the length of the bolt is a parameter, constant in one particular case but variable in others.

Parametric programming involves defining parameters and then using those parameters as the basis for one part program that may be used to machine not only the original component but a number of variations as well.

Figure 8.47(a) shows a component the dimensional features of which have been defined as parameters using the symbol # and a number: #1, #2, #3, and so on.

Figures 8.47(b)–8.47(g) show six variations of the component, the variations being indicated. A range of components such as this is referred to as a family of parts.

The machine movements necessary to machine each of the variations are all included in the original component. Some components require exactly the same movements, but with varying lengths of travel. Other components do not require all of the movements to be made. Using the more usual programming techniques, the production of each component would require a separate part program. Using the parametric part programming technique, instead of defining each dimensional movement individually in the *X* and *Z* axes, the parametric reference is programmed. Thus, to turn along the stepped diameter, the entry in the main program, referred to as the "macro," would read as follows:

```
N07   G01  X#4
N08   Z#2
```

These entries would suffice for all components requiring a stepped diameter. Equally, one entry using parametric identification would suffice for facing all the components to length or drilling the hole.

Having programmed all movements and the sequence in which they are to occur, it remains to define them dimensionally. The dimensional details are entered as a list at the start of the part program. Thus the parameters and their metric dimensional values for the original components would read as follows:

```
#1 = −50.00
#2 = −30.00
#3 = 30.00
#4 = 22.00
#5 = 10.00
```

As each parameter is called in the macro body, the programmed dimensional entry made previously will be invoked.

To machine any of the variations in the family of parts requires a simple amendment of the original parametric values. The parameters to machine the component shown in Figure 8.47(b) would be

```
#1 = −50.00
#2 = −40.00 (amended)
#3 = 30.00
#4 = 22.00
#5 = 10.00
```

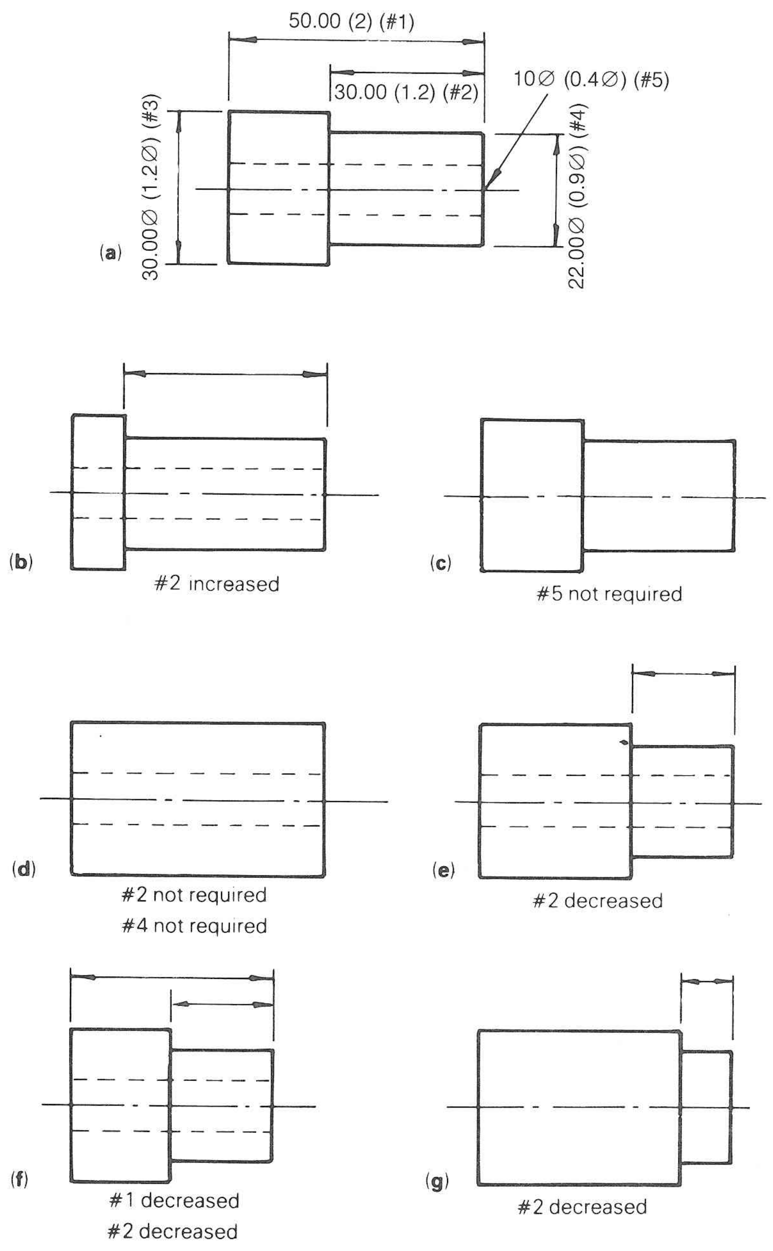


Figure 8.47 *Parametric programming: a family of parts. (Inch units are given in parentheses.)*

and to machine the component in Figure 8.47(f)

#1 = −40.00 (amended)
#2 = −20.00 (amended)
#3 = 30.00
#4 = 22.00
#5 = 10.00

Now consider the components where the programmed movements necessary for machining the basic component are not required. By using a relatively simple programming technique, the control unit can be caused to skip the redundant blocks. The necessary program entry involves the use of certain conditional expressions in which assigned abbreviations are used, such as the following:

EQ =	equal to
NE =	not equal to
GT =	greater than
LT =	less than
GE =	greater than or equal to
LE =	less than or equal to

Consider Figure 8.47(d) and assume the #1 and #3 have been machined. In the macro body the next call will be to machine the stepped diameter. To avoid this, blocks must be skipped and so an entry in the macro body will read as follows:

```
N15 IF [#4 EQ 0] GO TO N18
```

This statement says that if #4 is zero, move on to block number 18. Since #4 is nonexistent in the component, the parametric value will be entered as zero and, consequently, the control unit will move ahead.

The preceding description of the use of the parametric programming technique is a very simple one. It is in fact a very powerful concept and its full application is quite complex. For instance, parameters may be mathematically related within the macro body, that is, they may be added together, subtracted from one another, and so on.

In addition, the parametric principle may be extended to include speeds and feeds, when all the likely variations for roughing, finishing, etc., may be given a parametric identity and called into the program as and when required. *Note:* parametric and macro programming, if available on particular machinery, very drastically in programming techniques, so consulting the programming manual is required.

POINT DEFINITION

Point definition is a programming facility, not widely available, which simplifies programming for drilling operations. With this facility it is possible to

define dimensionally as many as 99 points or positions, and enter them into a special file within the control memory. The file can be accessed as required, the points' positions appearing in tabular form.

The points required for inclusion are entered at the start of a part program, and might look as follows:

```
N1    G78    P1    X15    Y20
N2    G78    P2    X20    Y20
N3    G78    P3    X50    Y30
N4    G78    P4    X65    Y60
N5    G78    P5    X75    X75
N6    G78    P6    X98    Y78
```

To drill a hole to a specified depth of 20 mm, using a G81 drilling cycle at points 2, 5, and 6, would require a program entry as follows:

```
N095    G81    Z-20    F150    S1850
N100    G79    P2      P5      P6
```

The more holes to be drilled, the more advantageous the use of the facility becomes. The dimensional data relating to each point can be modified to suit any particular job.

MIRROR IMAGE

Mirror image is the term that describes a programming facility used to machine components, or features of components, that dimensionally are identical but geometrically opposite either in two axes or one axis. By using the mirror image facility such components can be machined from just one set of data.

In Figure 8.48 an original component feature is indicated in the bottom left-hand corner of the diagram. A complete mirror image of that feature is shown in the top right-hand corner, while mirror images in the *X* axis and the *Y* axis are shown, respectively, in the bottom right- and top left-hand corners of the diagram.

To produce a complete mirror image both the *X* axis and *Y* axis dimensional values will change from negative to positive. For half mirror images the dimensional values will change from negative to positive in one axis only. Mirror imaging is normally achieved by repeating a series of program instructions after activiting a *X* and/or *Y* axis switch or button on the machine controller.

TOOL OFFSETS

Most machining operations involve the use of more than one tool, and usually they vary in length and/or diameter. To accommodate these size variations,

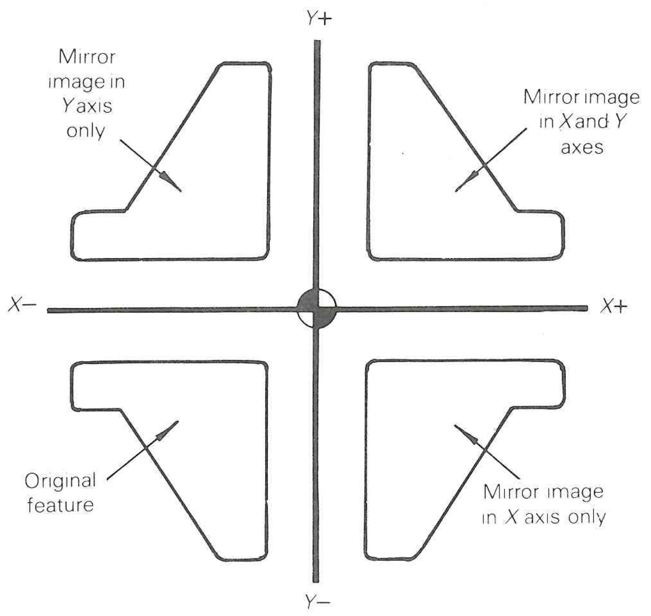

Y+
Y−
X−
X+
Mirror image in Y axis only
Mirror image in X and Y axes
Original feature
Mirror image in X axis only

Figure 8.48 *Mirror image.*

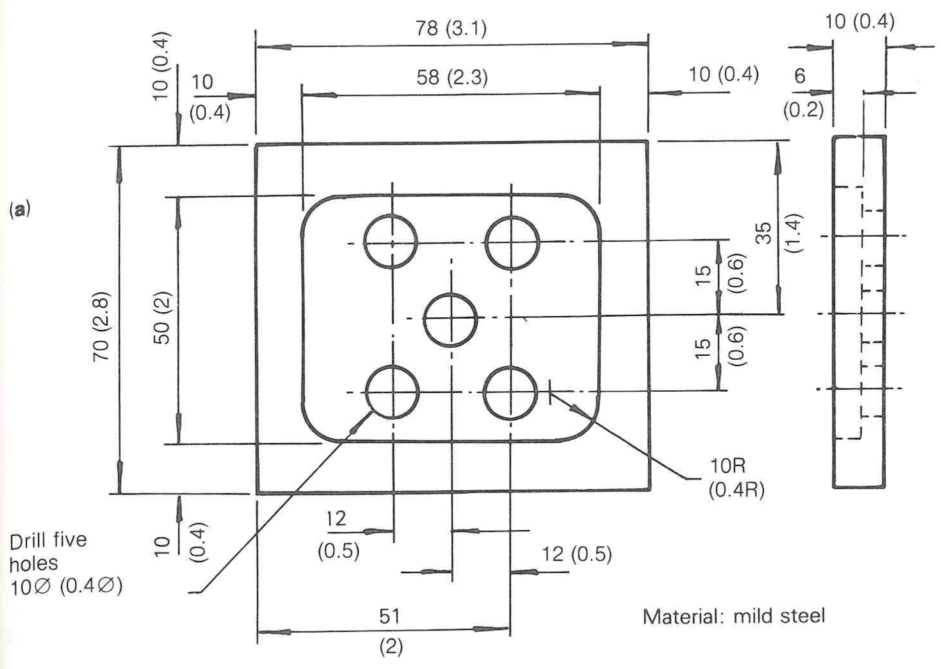
and to permit the programmer to assume that all the tools are identical, machine controls are provided with a cutter compensation facility that will, when activated, automatically adjust the programmed slide movements. Thus, it enables the programmer to totally ignore tool size and simply program movements that are exactly the same as the profile detail, a technique referred to as "point" programming or zero-gage-length programming. Some programmers do not agree with this form of programming owing to the inherent dangers if a compensation is missed or gage length incorrectly measured. Therefore, in gage length programming techniques where tool length and radius/diameter are allowed for in program calculations, the tool compensation is used for variations in setup dimensions versus programmed tool lengths.

The task of dealing with variations in tool size is left to the tool setter or operator, since it is essentially a case of ascertaining the tool sizes or size variations and entering these numerical values into the machine control.

The numerical values that are required to be entered relate to tool length and tool radius/diameter.

The manner in which tool length variations are determined and entered varies with machine type. On some controls they are entered simply as offset values, one tool being used as a reference tool and thus having no offset value, and all other tools having data entries corresponding to their dimensional variation from the reference tool. This principle is illustrated in Figure 8.49.

On other machines the size variations of all tools are determined in relation to a fixed point on the machine, such as the corner of a tool post, as illustrated



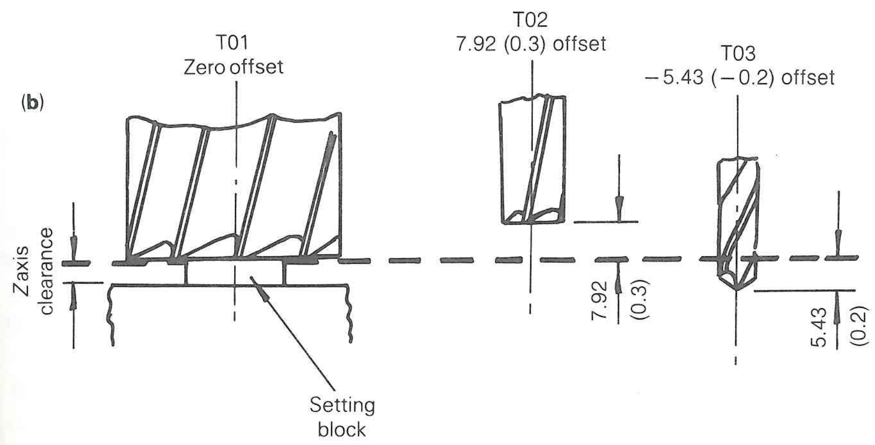


Figure 8.49 *Tool offsets related to a reference tool:* (a) *component detail;* (b) *tool offsets. (Inch units are given in parentheses.)*

in Figure 8.50. In this particular example the variations are entered in a tool data file, there being a second file for programmer-devised offsets that are, through appropriate program data, paired with the tool data file entries.

Tool radius and diameter entries are less complex, being simply a case of ascertaining the correct value, by actually measuring the tool if need be, and

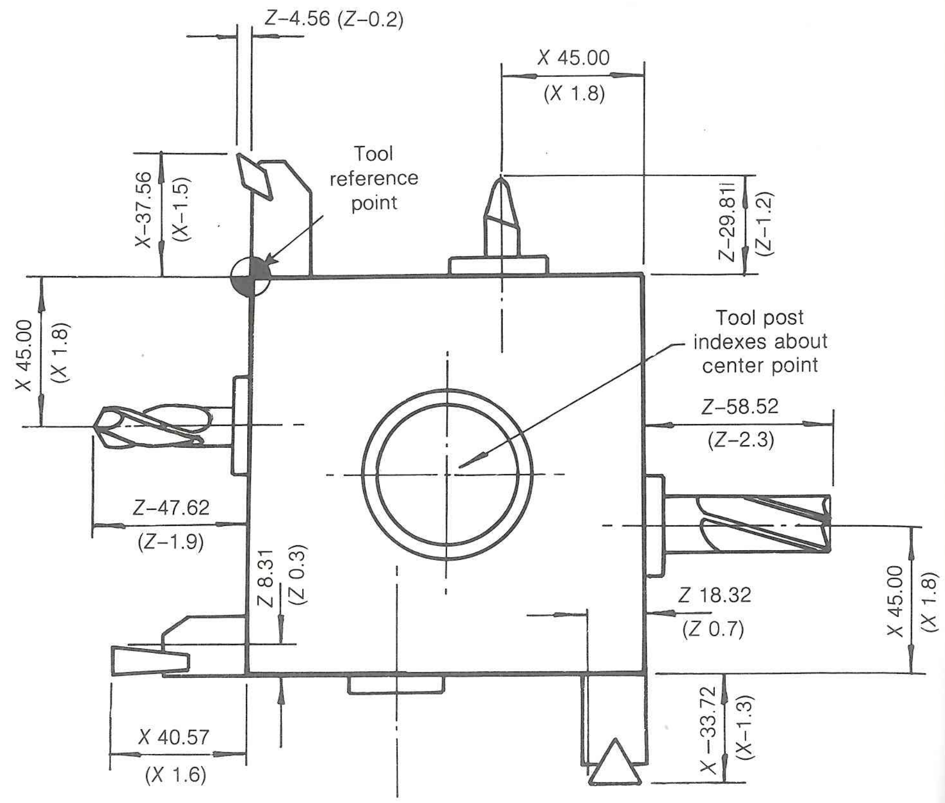


Figure 8.50 *Tool offset data related to a fixed point on the toolpost. (Inch units are given in parentheses.)*

entering the numerical value as required by the control system of the machine being used.

Since tool data can be entered, modified, or erased by the machine operator at will the facility can be used to

(a) accommodate replacement tooling that varies from the original;
(b) make variations to the component size;
(c) initiate a series of cuts, say roughing and finishing, using the same dimensional programmed data.

If the machine operator varies the tool offset values for whatever reason, the effect is temporary. It has no permanent effect on the original program. However, the programmer can utilize the offset facility within the part program, and this creates a situation where accurate communication between the programmer and the shop floor personnel is essential if the programming objectives are to be clearly understood.

Two situations involving tool offsets that are particularly useful from a programming point of view are when variations in the sizes of components are to be made from the same basic program and when a series of cuts are to be initiated along a profile using the same programmed dimensional data.

The ability to use offsets in this manner is based on pairing offset values with specific tools. Just as tools are allocated a numerical identity, so are offsets. Two digits are commonly used, just as in the case of tool identity. The offset digits are paired with the tool digits and are included in the program as part of the tool call. Thus, tool number three with offset number six would be entered in the program as T0306. Older word address programming systems may have special letter codes for calling the tool offset active, such as "H" for length and "D" for diameter.

Control systems usually provide for more offset entry capacity than there will be tools available, so it is possible to call any offset with any tool. The ability of programming any offset with any tool also allows programming more than one offset per tool. An example when to use this feature is when one tool is used to create more than one critical dimension.

The technique of using a number of offsets to make a series of cuts along a profile is illustrated in Figure 8.51. (A similar technique can be used when milling profiles and this is discussed later in the text.)

While the use of offsets as described previously is a very useful programming facility, it should be remembered that the prime objective of an offset facility is to make zero-gage-length programming a possibility and to allow for variation in cutter size during replacement simplifying the programming process. The preceding text has dealt only with tool lengths. It is now necessary to consider the way in which a variety of cutter diameters and tip radii can be accommodated.

To facilitate zero diameter programming with a variety of cutters of varying radii, the control should move the cutter away from the work profile a distance equal to its radius. This facility is referred to as "cutter radius compensation" or "cutter diameter compensation." This compensation also allows for variation in size do to cutter sharpening or deflection.

The distance the cutter will actually move away from or toward the profile—the offset—will be related to the data entry made by the machine setup person or operator. The offset is activated via the appropriate program entry.

The offset can be programmed to occur to the right or left of the required profile, commonly by the use of G41 or G42 when programming in word address mode. To determine which offset code should be programmed, the technique is to imagine being the tool and facing the direction of tool travel. The tool can then be visualized as being either to the right or left of the profile. It is very important to ensure that the correct offset is programmed since a move in the wrong direction may have disastrous results, particularly when large diameter cutters are being used. Tool radius to the right and left of a profile is shown in Figure 8.52.

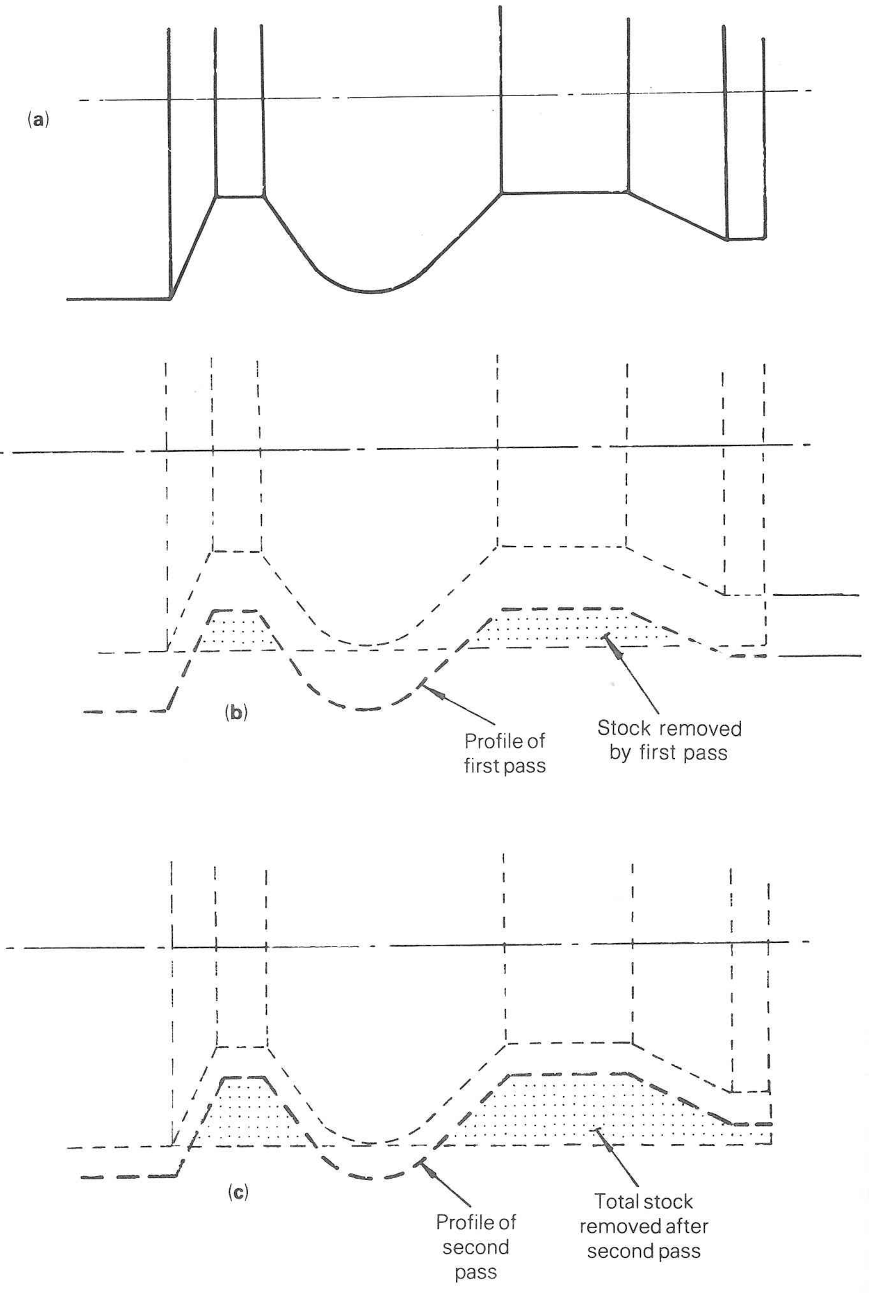

(a)
(b)
Profile of first pass
Stock removed by first pass
(c)
Profile of second pass
Total stock removed after second pass

Figure 8.51 *Use of tool offsets for progressive stock removal to a set profile:* (a) *profile detail;* (b) *first cut;* (c) *second cut.*

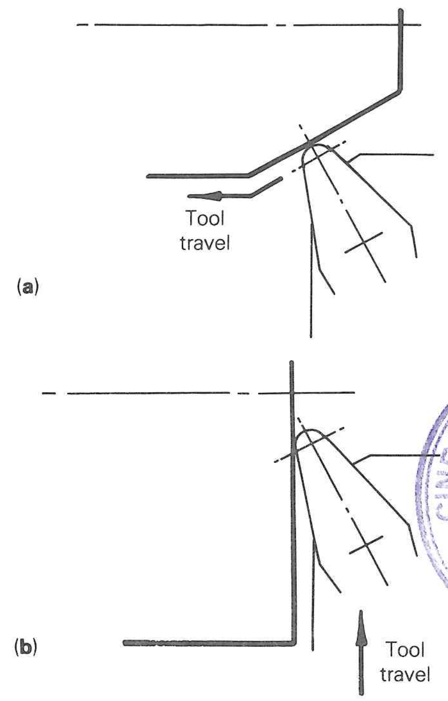


Figure 8.52 *Tool nose radius compensation (TNRC):* (a) *radius compensation left;* (b) *radius compensation right.*

When activating cutter radius compensation, it must be ensured that the slides will first make a noncutting move, to enable the correct tool and workpiece relationship to be established. A similar move is necessary prior to cancellation of the radius compensation. These noncutting moves are referred to as "ramp on" and "ramp off," respectively.

It is now possible to return to the technique referred to earlier, of using offsets to make a series of passes along a milled profile. It is achieved by simply entering a bogus value for the cutter diameter into the control system. By making an entry that is greater than the size of the cutter being used, the actual offset activated via the program will be greater. Thus, the final profile will remain oversize as illustrated in Figure 8.53. The technique can also be used progressively to remove surplus material before making a final cut.

The reverse application of this technique, that is, entering a value smaller than the actual cutter size, will result in a smaller offset and, in the case illustrated, an undersize component profile. Thus, it is possible to produce components of varying dimensions from the same program when milling, just as it is possible to do so when turning.

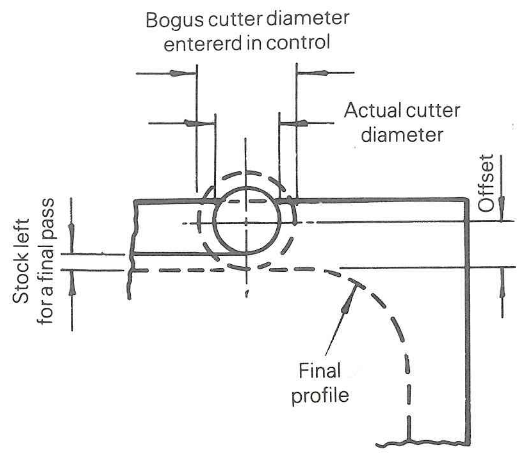


Figure 8.53 *Using an offset to create an oversize milled profile.*

BLOCK DELETE

Production engineering often involves machining a range of components that have slight variations from each other. For example, a hole that has to be drilled in one component is not required in a second component, although all the other details remain the same. Thus, one program would serve for both components providing that some means exists for not drilling the hole when it is not required. The way this is achieved is by use of a "block delete" facility.

Blocks relating to machining features that may not always be required incorporate the symbol /, which is referred to as a slash. The exact position of the slash within the block may vary from one system to another, but it is usually at the start before the sequence number (/N0245).

The machine operator will need to be instructed as to whether the data are to be retained or deleted from the current machining task. If the data are to be retained, the operator takes no action. If the data are to be deleted, the operator has to activate the block delete button on the control panel before running the program. Activation of the button is usually indicated by a light.

If the slash delete button is not activated, the control will respond to all the data contained in the program. If the slash delete button is activated, then all the blocks containing a slash will be ignored.

On some control systems if the slash delete button is not activated, the program will automatically stop when the first slash is reached and the operator then has to make a positive response, either to activate the data contained within the slashes or delete them.

The block delete facility is also useful when machining castings or forgings, where stock removal requirements may vary. The operator is given the option to include an extra cut or not as necessary.

The use of the block delete facility relies on a clear and concise relay of instructions between the part programmer and the machine operator. The machine operator must be left in no doubt as to what is required.

PROGRAM STOPS

Apart from the program stop that is automatically effected when the end of a program is reached, and which arrests all slide and spindle motions, there are two other situations where a halt in proceedings may need to be included in the part program.

The first of these is the point at which the operator is required to carry out some specific task directly associated with the machining program, such as resetting the work or replacing a tool. With word address programming this is normally achieved by programming M00. When such a stop is effected, it is essential that the operator knows exactly what has to be done before the program is reactivated. Some controls will allow for a man readable message to be read from the program and placed on the screen.

The second type of program stop is used when a halt in activity is not quite so critical, and the operator decides whether a stop is actually made. This type of stop is referred to as "optional" and will only take place if the operator has activated the optional stop button on the control console. The programmer may include an optional stop in the program whenever he or she considers it may be of value to the machine operator, such as when a dimensional check or an inspection of the tooling condition is appropriate. But quite often it is the operator who will edit into the program, via the control console, stops that will permit the solving of particular problems that have presented themselves during the machining process. The optional stop in a word address program is normally effected via a programmed M01.

In addition to the stops included in the part program, the operator has, of course, recourse to an emergency stop should the need arise.

CALCULATIONS

It could be argued that, in a well-organized CNC machining environment, the people responsible for the production of the detail drawings of the components to be machined should appreciate the needs of the part programmer and ensure that the drawings are dimensioned accordingly. For example, it is of considerable help when positional slide movements are to be programmed in absolute mode if all dimensions on the drawing are given in relation to a suitable datum. This is especially valuable when the programming technique involves conversational or manual data input, since the last thing a programmer wants to do is to interrupt his thought process in order to calculate unspecified dimensions.

However, whatever the ideal situation may be, it is almost certain that eventually the part programmer will be confronted with a detail drawing that does not cater to his or her requirements. He or she will then find it necessary to make calculations and add dimensions, and perhaps in some cases to completely redimension the drawing.

(The reader should differentiate between poor industrial practice and situations with which he or she may be deliberately confronted in a learning situation, where the objective will be to provide an understanding of the problems likely to be encountered in practice.)

Mention has already been made of the need for dimensions to be given in relation to a set datum when absolute programming is to be used. The opposite situation could also arise, whereby the dimensions are stated in relation to a datum but the programmer needs to program incremental slide moves. In this case the stated dimensions will need to be subdivided. The programmer should exercise caution in this particular situation, and ensure that such an approach is acceptable from a design point of view; minor errors on each of a series of incremental moves could, due to inaccuracies in calculations or round off, accumulate into a larger error that would be unacceptable.

There are other situations that are more complex than simply converting absolute dimensions to incremental and vice versa. Two of these in particular are the need to determine:

(a) profile intersection points;
(b) the location of arc centers.

However complex the profile or shape of a machined surface may appear to be, it can be broken down and defined geometrically as a number of intersecting straight lines or arcs or a combination of the two. To program appropriate machine slide movements, the programmer is required to determine this geometry, and translate production of the profile into a series of linear or circular movements.

Thus, to finish machine the profile shown in Figure 8.54 the following movements will be necessary:

Move 1	Linear
Move 2	Circular, clockwise
Move 3	Linear
Move 4	Circular, counterclockwise
Move 5	Linear

If word address programming is being used, these moves can be described using the appropriate G code: G01, G02, G01, G03, and G01. It may be helpful to mark the drawing accordingly, as in the illustration.

The reader will already appreciate that when programming positional moves, whether they are linear or circular, the target position has to be numerically defined. In this particular example, because the component is dimensioned correctly and the arcs are conveniently 90°, the target position, that is, the intersection points of the geometrical elements of the profile, are readily discerned. No further calculations are necessary.

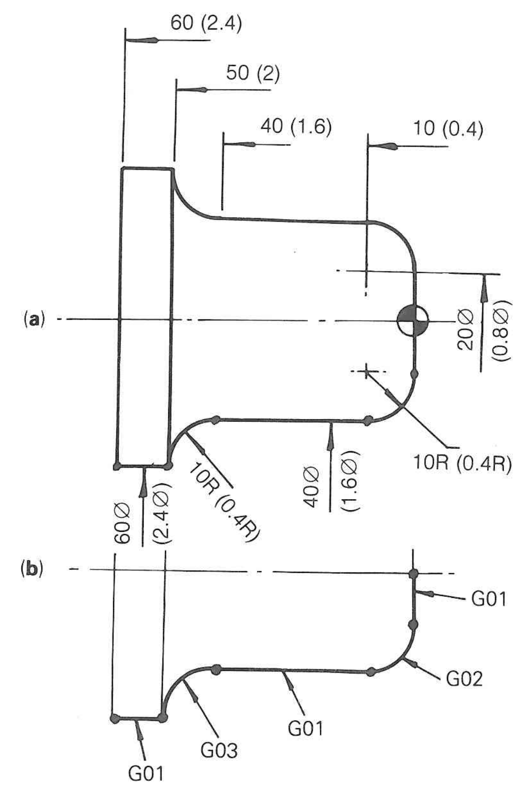


Figure 8.54 *Geometric elements of a profile:* (a) *component detail;* (b) *profile definition. (Inch units are given in parentheses.)*

Consider now the component shown in Figure 8.55(a). Although it is a relatively simple profile, from a programming point of view the drawing is not as helpful as it might be. The target position of the arc, that is, the point at which the circular move ends and the linear move commences, is not defined. Calculations are required as follows in order to determine the target position in the X and Z axes. (Only the metric unit calculation is given.)

In Figure 8.56:

$$\varnothing X_1 = 2(CB + 10) \text{ and } Z_1 = CD$$

In triangle ABC:

$$\angle ABC = 15° \text{ and } AB = 12.5$$

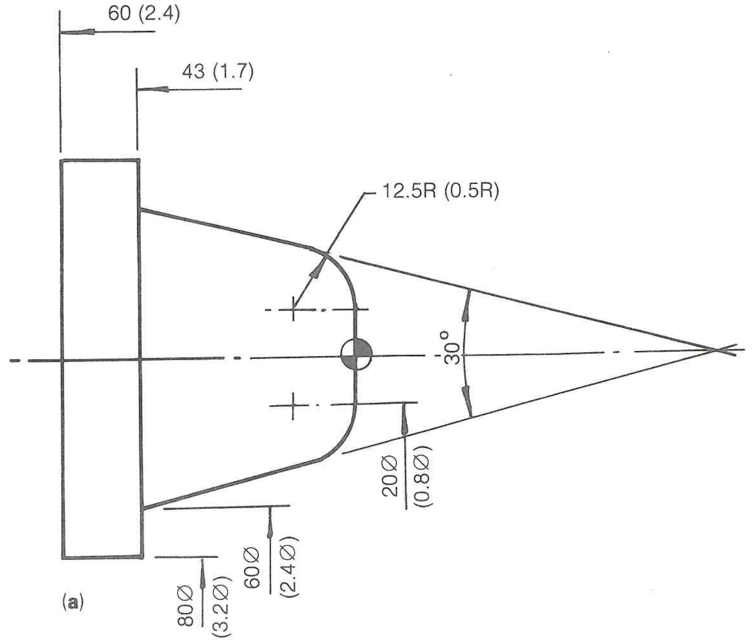


Figure 8.55 *Component detail. (Inch units are given in parentheses.)*

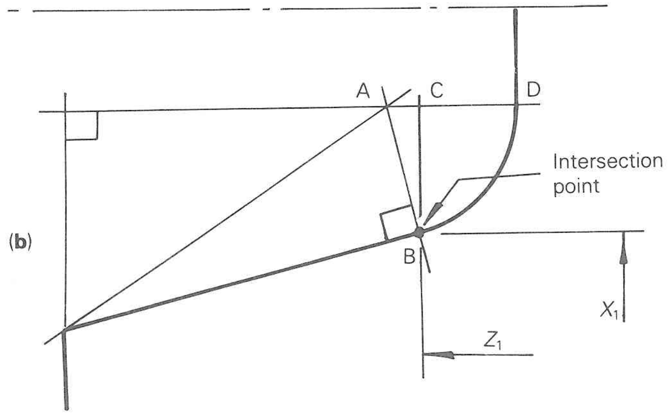


Figure 8.56 *Profile intersection calculation.*

1. To calculate CB:

$$\cos \angle ABC = \frac{CB}{AB}$$

$$\begin{aligned} CB &= \cos \angle ABC \times AB \\ &= \cos 15 \times 12.5 \\ &= 0.966 \times 12.5 \\ &= 12.074 \end{aligned}$$

$$\varnothing X_1 = 2(12.074 + 10) = 44.148$$

2. To calculate AC:

$$\sin \angle ABC = \frac{AC}{AB}$$

$$\begin{aligned} AC &= \sin \angle ABC \times AB \\ &= \sin 15 \times 12.5 \\ &= 0.259 \times 12.5 \\ &= 3.235 \end{aligned}$$

$$\text{Since } Z_1 = CD, \text{ then } Z_1 = AD - AC = 12.5 - 3.235 = 9.265$$

Thus, the target position, with the *X* value being programmed as a diameter, is X44.148 Z−9.265. These values would need to be included in the part program.

This particular calculation is fairly typical of the situations the part programmer has to deal with. A similar situation presents itself in the profile shown in Figure 8.57(a) where one radius blends with another. The problem is: where does one radius end and the second one start? Calculations are necessary to determine the location of point P in the *X* and *Y* axes as indicated in Figure 8.57(b). The reader may like to consider the solution to the problem. (Answers: 47.81 mm and 71.25 mm, respectively.)

This type of profile also presents the second type of calculation referred to earlier, namely, determining the location of arc centers.

From the previous information the reader will recall that when circular arcs are programmed using word address programming one of three techniques may be involved. All require the target positions to be identified, but the radius definition varies. The first involves defining the arc center in relation to the program datum, and the second requires the arc center to be defined in relation to the arc starting position.

Using the first method, the center of the 30 mm radius arc is easily determined from the dimensions already on the drawing. But the location of the center of the 50 mm radius is not so straightforward and a calculation is required.

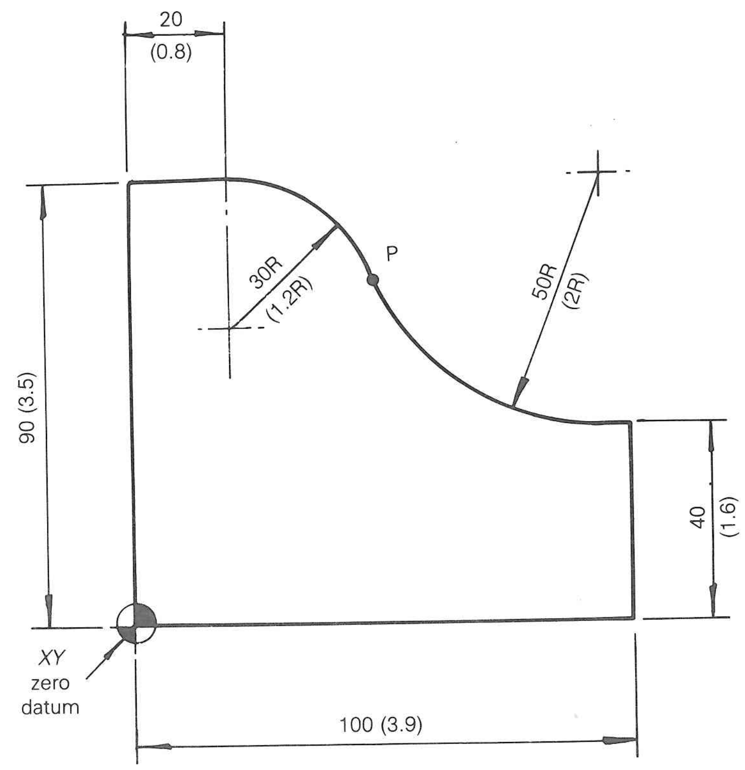


Figure 8.57 (a) *Profile detail. (Inch units are given in parentheses.)*

Using the second method of circular interpolation the definition of the arc centers in relation to the start points again presents a problem as far as the 50 mm radius is concerned, and a calculation will be necessary before the program can be written.

Exercises involving the calculation of profile intersection points and arc centers are included in this chapter's section titled "Part Programming Calculations."

TOOL PATHS

A prime objective of the part programmer should be to ensure that a component is machined in the shortest possible time. Earlier in the text reference was made to the way a well-planned sequence of machining operations can contribute to this objective. But often within each individual machining sequence there is room for further efficiency, which results in time saving.

Consider the drilling of the series of five holes in the component shown in Figure 8.58. Two sequences in which the holes might be drilled are indicated. Which sequence would be the quicker?

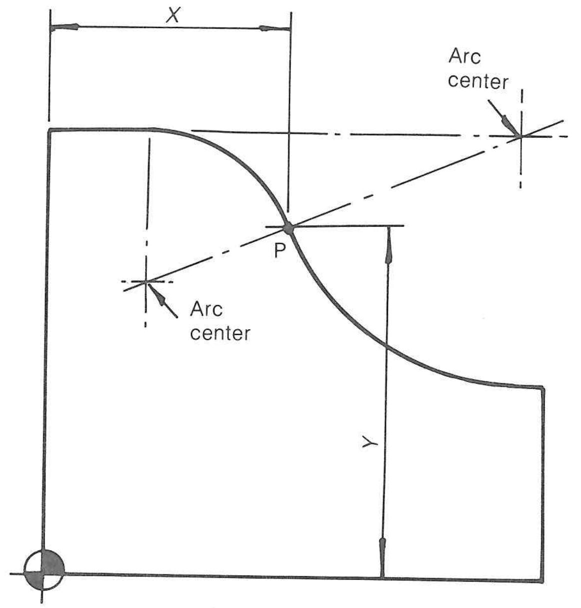


Figure 8.57 (b) *Required profile intersection dimensions.*

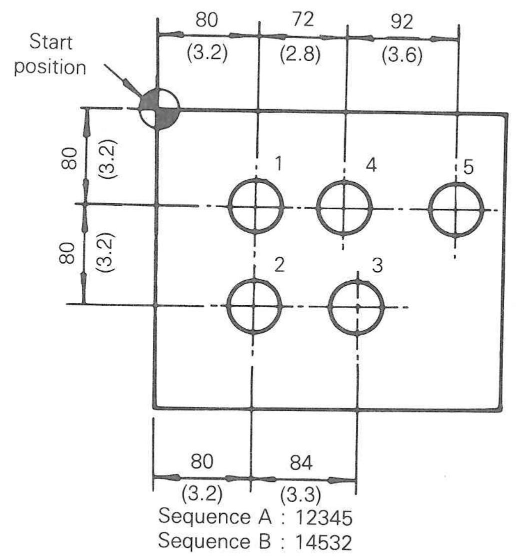


Figure 8.58 *Alternative drilling sequences. (Inch units are given in parentheses.)*

The actual operation of drilling the holes, that is, movement in the Z axis, would be identical in both cases. Therefore, any saving that can be achieved must be by reducing the total length of the positioning moves, and therefore the time taken.

Providing the detail drawing is reasonably accurately drawn a simple rule measurement check may suffice to determine the shorter route. Applying this technique to this particular example will reveal that the second sequence is quicker than the first.

The need to give careful consideration to tool paths is also important during stock removel operations. This is particularly so when there are no stock removal canned cycles available within the control system, or if they cannot be utilized in a particular situation.

Consider the removal of stock, or area clearance as it is also known, in order to machine the step shown in Figure 8.59.

If a pocket milling cycle is available on the control system of the machine, this could be used, the missing sides of the "pocket" being indicated by the dotted line. Use of the cycle would ensure that efficient tool paths are employed.

If such a cycle is not available, then the matter becomes a little more complex. The process of producing the step will involve programming a series of linear moves, with careful attention being given to providing an appropriate cutter overlap to ensure a clean face. The lengths of relative cutter travel will also have to ensure a uniform amount of metal is left for a finish pass along

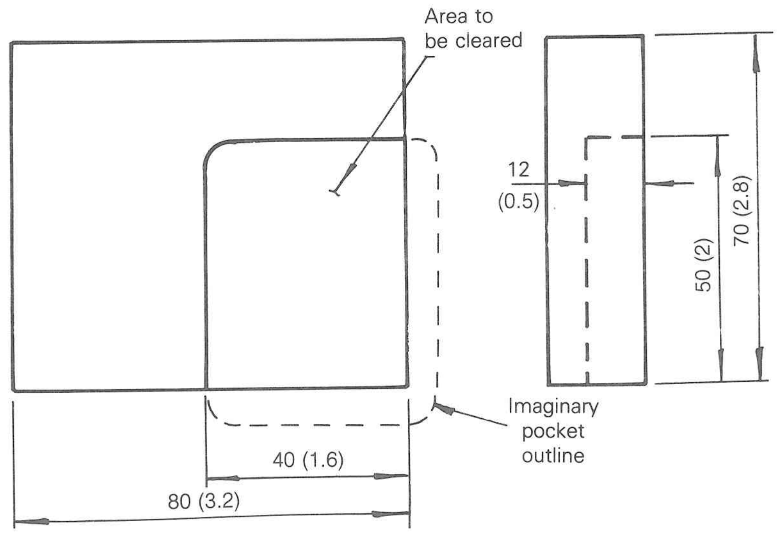


Figure 8.59 *"Pocket" detail. (Inch units are given in parentheses.)*

the profile. The programmer should also ensure that the cutter paths used are the shortest and therefore the quickest.

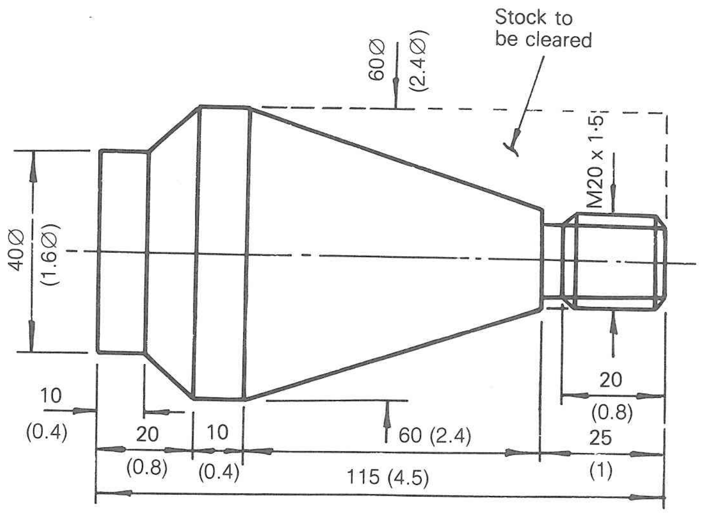
Similar problems often present themselves during turning operations. Figure 8.60 shows a typical example.

There is no short cut when solving this type of problem. After a little experience of dealing with situations of this nature, the trainee programmer soon comes to appreciate the value of canned cycles, which reduce the amount of machining dealt with in this manner to a minimum.

If the part programmer is confronted with machining situations such as these, he or she will have to resort to drawing the profile, preferably to an enlarged scale, and then imposing appropriate tool paths on the drawing. In the case of milling examples it may be necessary to draw circles indicating the cutter diameter. The milled step referred to above, when dealt with in this way, is shown in Figure 8.61. Having decided on the most suitable tool paths (which may take a number of attempts), the slide movements may be dimensionally determined by carefully scaling the drawing or through mathematical calculation.

An alternative approach is to reproduce the profile on graph paper, as shown in Figure 8.62, in which case the graduated lines on the graph paper can be used to determine the dimensional value of the necessary moves.

Exercises involving the determination of cutter paths are included in subsequent examples.



Material: medium carbon steel 60∅ (2.4∅)

Figure 8.60 *Component requiring excess stock removal. (Inch units are given in parentheses.)*

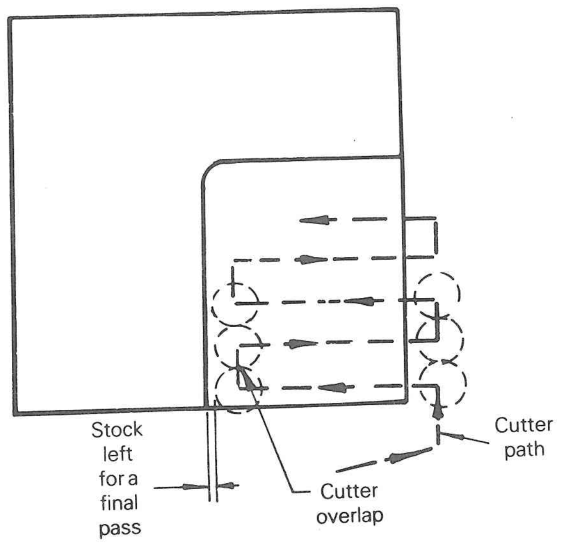


Figure 8.61 *Determination of cutter path to mill a step.*

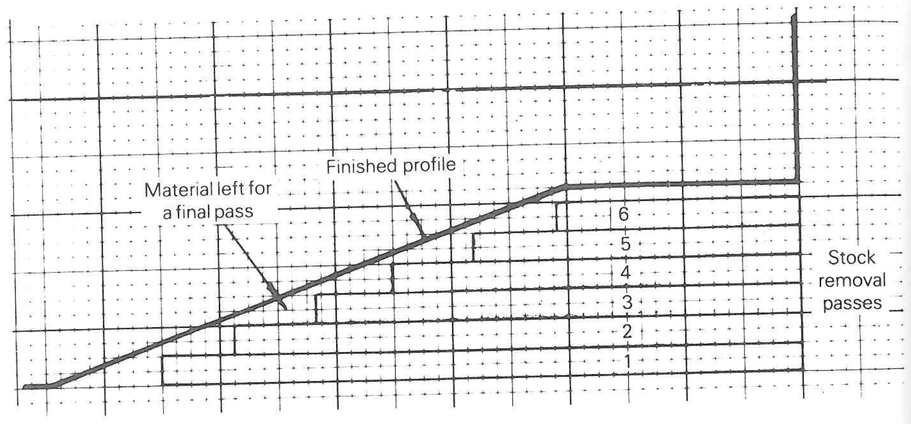


Figure 8.62 *Determination of tool paths when turning.*

PROGRAM LISTING AND PROVING

Before starting to create a part program listing, all the various facets of competent part programming discussed so far should have received due attention.

The sequence of operations, together with the tooling and work-holding techniques to be employed, should be documented. Appropriate speeds and feed rates should have been determined and all the necessary calculations affecting slide movements must be complete.

Having reached this stage the way is clear to list the program, and this requires the programmer to be conversant with the machine programming language.

To become fully proficient with a particular programming system takes time and practice. As with most things it is a case of starting with relatively simple tasks and gradually progressing to more complex examples. If you are a student, or perhaps undergoing training in an industrial establishment, it is almost certain that your course work will be structured in this way.

Competent part programming demands a logical approach and a high degree of concentration and care when actually listing the program. Mistakes are easily made and can have disastrous results, although fortunately most mistakes can be discovered and rectified before machining takes place.

Programs may be listed on appropriate forms, or on plain paper or they can be entered into a computer and listed on the display screen. Programs initially handwritten can, of course, also be entered into a computer and visually displayed.

The use of a computer for program listing is often coupled with the facility to prove the program using animated computer graphics. This involves, in effect, "machining" the component on the screen.

The effectiveness of proving programs in this way will depend on the sophistication of the software available. The simplest software will usually highlight major errors such as movement occurring in the wrong direction or a lathe tool crashing into a chuck, while the more complex will also indicate errors relating to speeds and feeds and even the absence of a coolant supply.

Ultimately, the part program will be entered into the machine control unit, but this may also involve computer graphics. Figure 8.63 shows a controller that includes a built-in visual display. As the program is entered the CRT screen will display the geometric profile of the part and the programmed cutter paths and thus confirm, or otherwise, the validity of the data input. The illustration relates to a program written for the part detailed in Figure 8.64. An enlargement of the CRT display is shown in Figure 8.65 where the component profile can be more readily defined.

A large number of machines currently in use do not have the benefit of built-in computer graphics, and if off-line computer graphics proving facilities are not available, then the proving of the part program must take the form of a test run or a dry run or both.

The test run is basically a check that the data input is valid, that is, that the machine is capable of responding to the data entries included in the program. Data errors are usually indicated by a displayed message. No slide movement takes place during the test run.

The dry run procedure also excludes metal cutting, but, with this checking procedure, slide movements occur at a rapid rate of traverse. This test ensures that the intended machine movements are occurring, and that they will result in the machined features required.

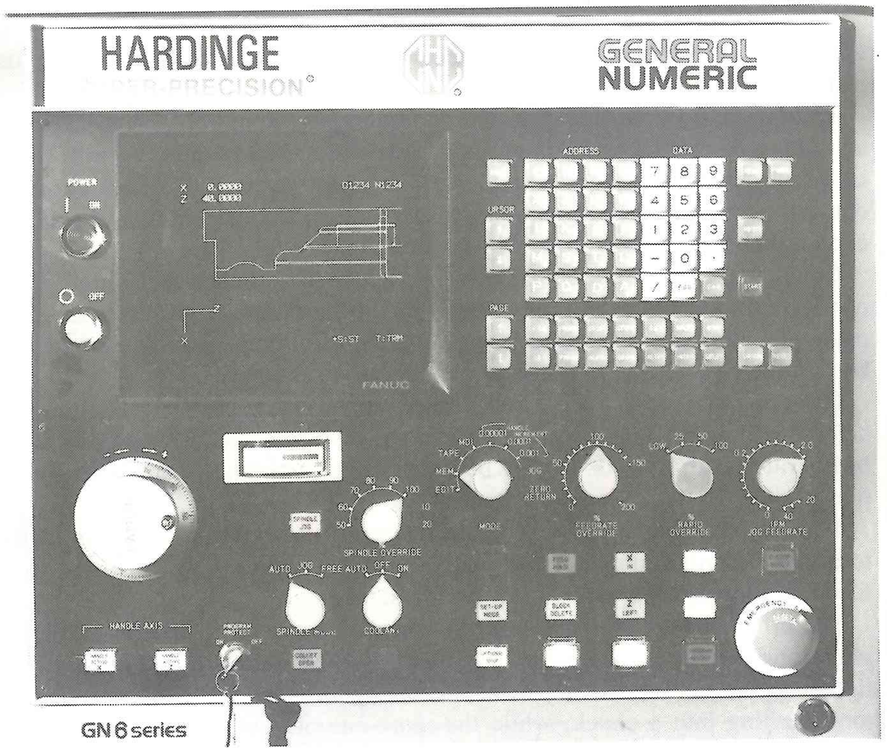
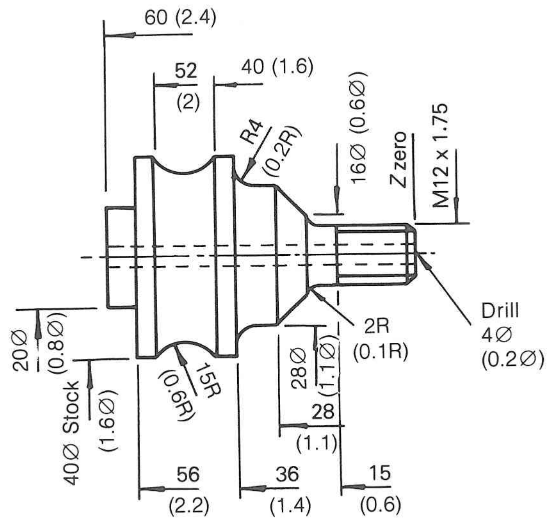


Figure 8.63 *A machine controller with built-in CRT to facilitate program proving.*



Material: aluminum allow 40Ø (1.6Ø)

Figure 8.64 *Component detail. (Inch units are given in parentheses.)*

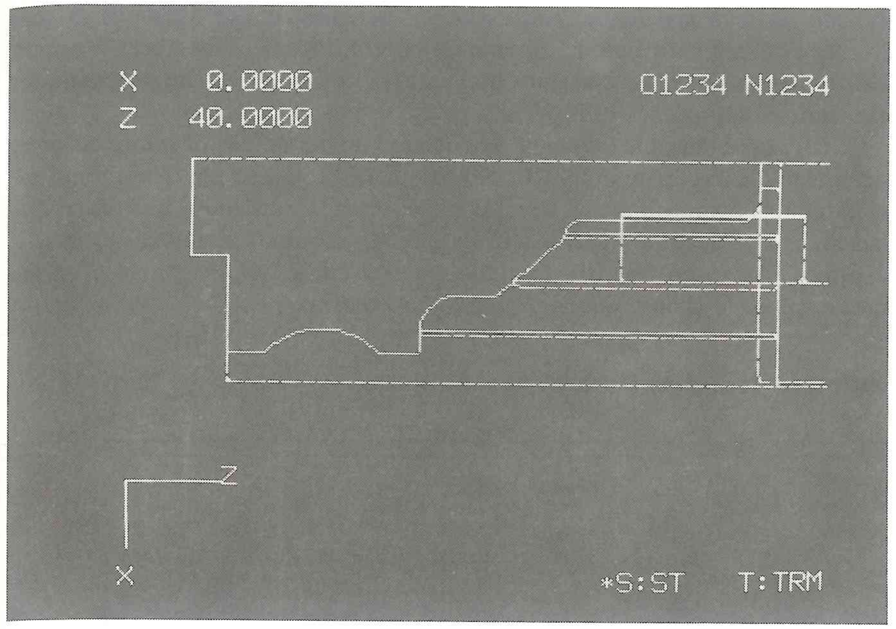


Figure 8.65 *An enlargement of the CRT display shown in Figure 8.63.*

The most common test available, even on the simplest of machines, is to run the program through one block at a time with reduced programmed feed rates, but without the job material being in position so that no machining actually takes place—block by block prove out. This prove out method is usually followed by a block by block run with part material and reduced feed rates before an automatic run is attempted.

Tests of this nature carried out on the machine may or may not be the responsibility of the programmer, although he or she will soon be involved if any errors are indicated.

Finally, there is the need, particularly in industrial situations, to record the program for future use. In its simplest form, storage can be a handwritten version of the proved program. Alternatively, it may be in the form of a perforated tape or be recorded on a magnetic tape or disk.

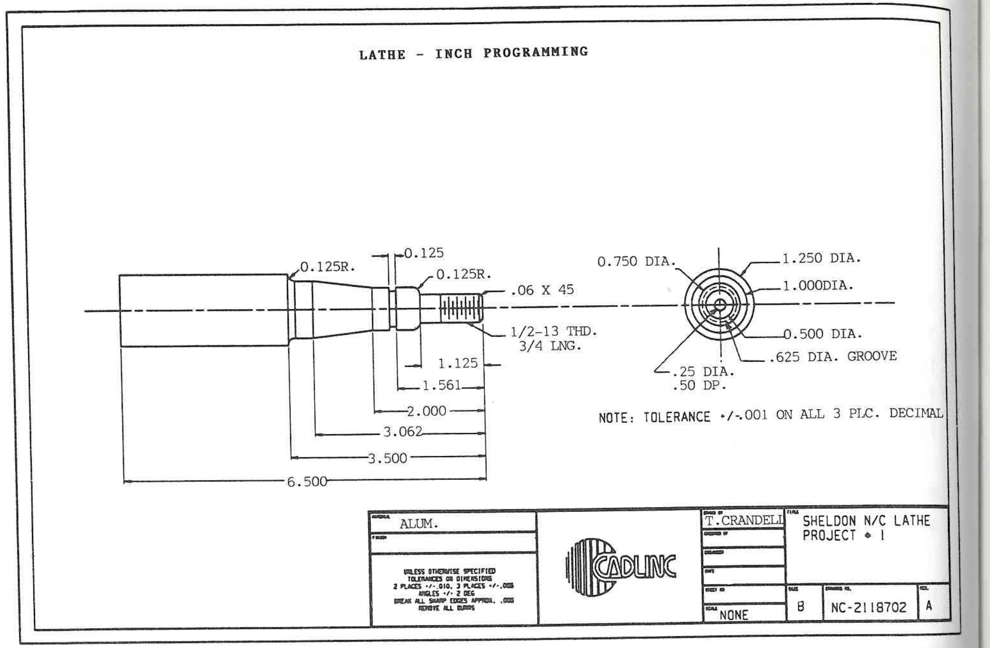
Whatever the storage medium, it must be remembered that people looking to reuse the program at a later date will also need information relating to tooling and workholding. This information is as critical as the part program and must also be carefully filed for future reference.

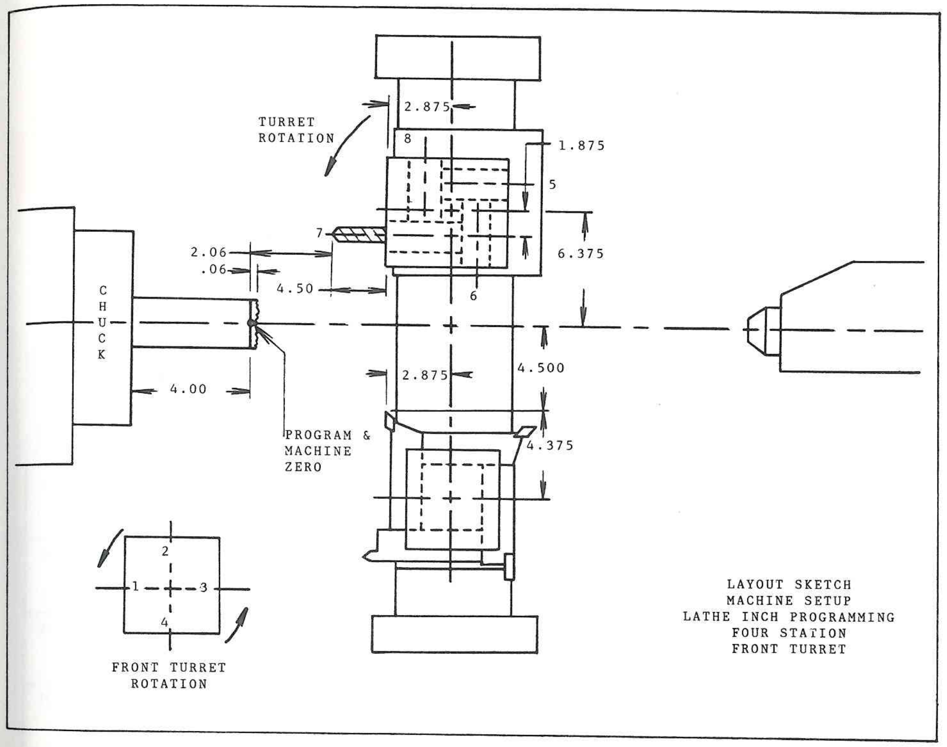
MANUAL PART PROGRAMMING EXAMPLES

The programming examples that follow were prepared to show the calculation and writing of generic manual part programs. Examples follow for both a lathe and mill in inch and metric calculations. The intent of the author is to give you the realization of putting an entire program document together. We must realize though that most machines program slightly different and programmers take various approaches to how they process, tool, and program.

Finally, note that it is common practice to number blocks of information by increments of five or ten: N0010, N0020, N0030, and so on. The reason for adopting this approach is that if, on completion of the program, it is found that something has been omitted, it will be possible to insert additional blocks. It also provides space that will facilitate general editing of the program at the machine control should this be found to be necessary.

Example 1: Lathe Inch Programming



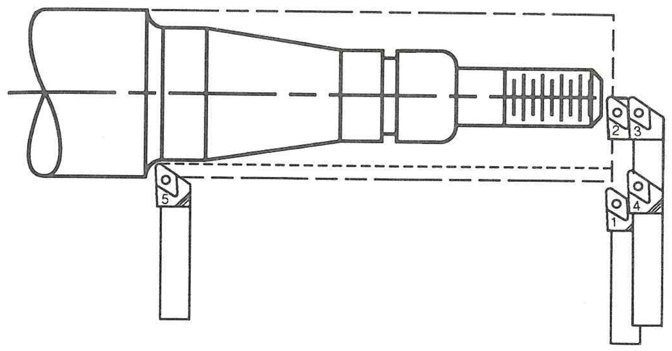


TOOL SHEET

LATHE INCH PROGRAMMING

PART NO. NC-2118702	MATERIAL ALUM.	OPERATION 20	PROGRAMMED BY T. CRANDELL
PART NAME PROJECT 1	B/P CHANGE DATE 9-21-87	DATE 5-5-89	SHEET 1 OF 1

TOOL SEQ.	TOOL NUMBER	TOOL DESCRIPTION	TOOL LENGTH X PROG.	TOOL LENGTH X ACT.	#	TOOL LENGTH Z PROG.	TOOL LENGTH Z ACT.	#	OPERATION DESCRIPTION	SPEED R.P.M.	SPEED CODE	FEED IN/MIN	FEED CODE
	T01	55 DEG. DIAMOND TURNING TOOL (RGH) .031 TOOL NOSE RADIUS Valenite Holder MDJNR-13-4 Insert DNMP432E VN5	4.375			2.875			RGH. FACE & RGH. TURN	960 to 2000		.010 IPR	
	T02	55 DEG. DIAMOND TURNING TOOL (FIN) .031 TOOL NOSE RADIUS Valenite Holder MDJNR-13-4 Insert DNMP432E VN5	4.375			2.875			FIN. FACE & FIN. TURN	2000		.005 IPR	
	T03	.125 GROOVE TOOL Valenite Holder SD-IMR-16-3 Insert TNMC32NG VN5	4.375			2.875			.125 GROOVE	1000		.005 IPR	
	T04	60 DEG. THREAD TOOL Valenite Holder SD-IMR-16-3 Insert TNMC32NV VC7	4.375			2.875			½-13 THREAD	2000		.077 IPR	
	T07	.250 DIA. HSS DRILL	1.875			4.500			DRILL .250 DIA.	2000		.006 IPR	

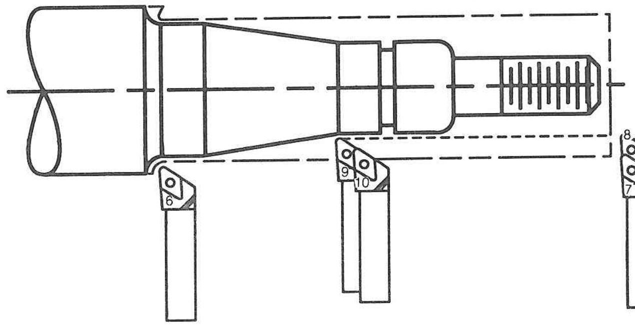


COORDINATE CALCULATIONS

LATHE INCH PROGRAMMING

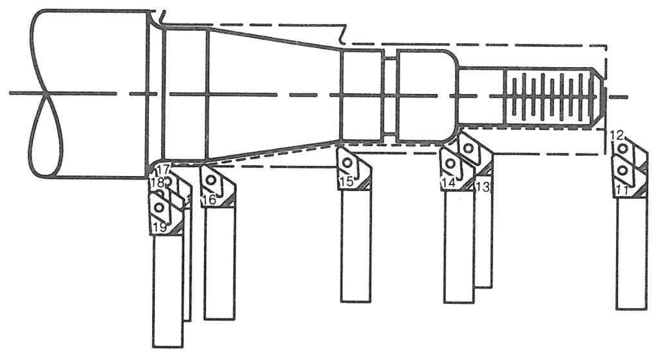
ROUGH FACE & FIRST ROUGH TURN

POINT CODE	X COORDINATE	Z COORDINATE	CALCULATION
P1	X .725	Z .030	X .625 Stock Radius + .100 Clearance Z .030 Finish Stock
P2	X-.050	Z .030	X-.030 Tool Radius - .020 Cut Past Center Z .030 Finish Stock
P3	X-.050	Z .050	X-.030 - .020 Z .030 Finish Stock + .020 Clearance
P4	X. 525	Z .050	X .500 Part Radius + .025 Stock
P5	X .525	Z-3.406	Z-3.375 - $\sqrt{.069^2 - .069^2}$ - .031 Formula #1 Appendix E



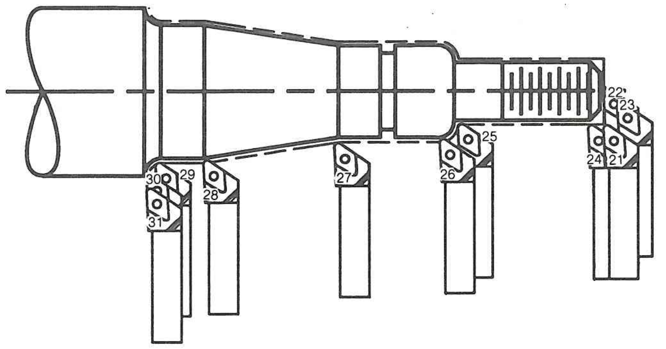
SECOND ROUGH TURN

POINT CODE	X COORDINATE	Z COORDINATE	CALCULATION
P6	X .575	Z-3.356	X .525 Radius + .050 Clearance Z-3.406 Pos."5" + .050 Clearance
P7	X .575	Z .050	Z .030 Finish Stock + .020 Clearance
P8	X .400	Z .050	X .375 Part Radius + .025 Stock
P9	X .400	Z-2.0277	Z-2.0 -[.031-(Tan.$\frac{6.7129^{o}}{2}$ X (.031+.025))] Formula #2 Appendix E
P10	X .450	Z-1.9777	X .400 Radius + .050 Clearance Z-2.0277 Pos."9" + .050 Clearance



SEMI FINISH TURN

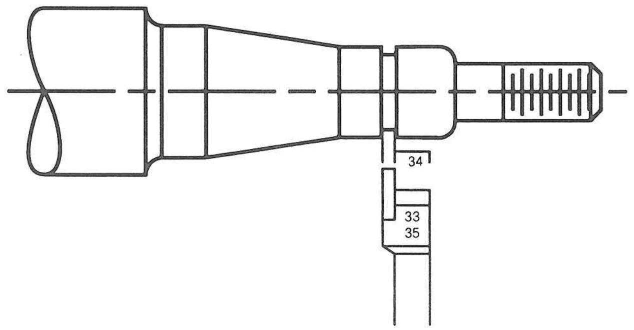
POINT CODE	X COORDINATE	Z COORDINATE	CALCULATION
P11	X .450	Z .050	Z .030 Finish Stock + .020 Clearance
P12	X .265	Z .050	X .250 Part Radius + .015 Finish Stock
P13	X .265	Z-1.1164	Z-1.125 - .125 + .1646 - .031 Formula #3 Appendix E
P14	X .390 I.046	Z-1.281 K.1646	X .375 Part Radius + .015 Finish Stock Z-1.250 Radius Center - .031 Tool Radius
P15	X .390	Z-2.0283	Z-2.0 -[.031-(Tan.$\frac{6.7129^{o}}{2}$ X (.031+.015))] Formula #2 Appendix E
P16	X .515	Z-3.0903	X .500 Part Radius + .015 Finish Stock Z-2.0283 Pos. 15 - 1.062 Taper Length
P17	X .515	Z-3.406	Z-3.500 Part Dim. + .125 Radius - .031 Tool Radius
P18	X .594 I.079	Z-3.485 KO	X .625 Part Radius - .031 Tool Radius Z-3.500 Part Dim. + .015 Finish Stock
P19	X .750	Z-3.485	X .625 Part Radius + .125 Clearance



FINISH FACE AND FINISH TURN

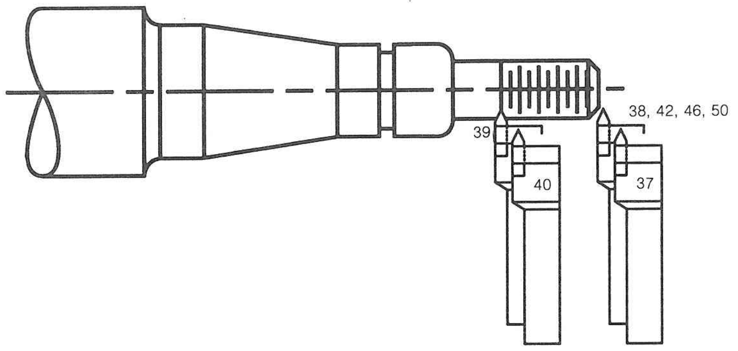
POINT CODE	X COORDINATE	Z COORDINATE	CALCULATION
P20	TOOL	CHANGE	POSITION
P21	X .365	Z 0	X .250 Part Radius + .015 Stock + .100 Clearance Z 0 Finish Datum
P22	X-.050	Z 0	X 0 Datum - .031 Tool Radius - .019 Past Center
P23	X .0718	Z .100	X .250 Part Radius - .06 Cham. - .10 Clearance +.0128 Formula #4 Appendix E -.031 Tool Radius Z 0 Datum + .100 Clearance
P24	X .250	Z .0782	X .250 Part Radius Z-.060 Chamfer + .0128(formula 4) -.031 Tool Radius
P25	X .250	Z-1.1281	Z-1.125 - .125 + .1529 - .031 Formula #3 No Stock Appendix E
P26	X .375 I.031	Z-1.281 K.1529	X .375 Part Radius Z-1.125 - .125 radius - .031 Tool Radius
P27	X .375	Z-2.0292	Z-2.0 -[.031-(Tan.$\frac{6.7129^{\circ}}{2}$ X.031)] Formula #2 No Stock Appendix E
P28	X .500	Z-3.0912	X .500 Part Radius Z-2.0292 Pos. 27 - 1.062 Taper Length

POINT CODE	X COORDINATE	Z COORDINATE	CALCULATION
P29	X.500	Z-3.406	Z-3.500 Part Dim. + .125 Radius - .031 Tool Radius
P30	X .594 I.094	Z-3.500 K0	X .625 Part Radius - .031 Tool Radius Z-3.500 Part Dim.
P31	X .750	Z-3.500	X .625 Part Radius + .125 Clearance
P32	TOOL	CHANGE	POSITION



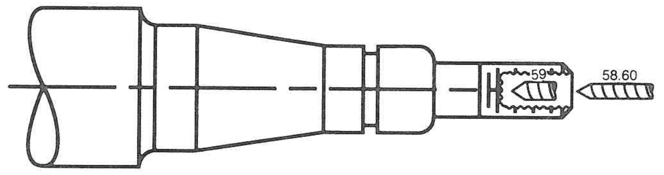
CUT GROOVE

POINT CODE	X COORDINATE	Z COORDINATE	CALCULATION
P33	X .475	Z-1.687	X .375 Part Radius + .100 Clearance Z-1.561 Part Dim. - .125 Groove Width
P34	X .3125	Z-1.687	X .3125 Groove Radius
P35	X .475	Z-1.687	X .375 Part Radius + .100 Clearance
P36	TOOL	CHANGE	POSITION



THREAD CUTTING

POINT CODE	X COORDINATE	Z COORDINATE	CALCULATION
P37	X .350	Z .100	X .250 Part Radius + .100 Clearance Z 0 Finish Datum + .100 clearance
P38	X .2284	Z .0876 K.0769	X .250 Part Radius - .0216 Thread In Feed Z .100 Clearance - .0124 Thread In Feed K 1 ÷ 13 (Thread Pitch)
P39	X .2284	Z-.750	Z-.750 Thread Length Dim.
P40	X .350	Z-.750	X .250 Part Radius + .100 Clearance
P42	X .2166	Z .0808	X .250 Part Radius - .0334 Thread In Feed Z .100 Clearance - .0192 Thread In Feed
P46	X .2076	Z .0756	X .250 Part Radius - .0424 Thread In Feed Z .100 Clearance - .0244 Thread In Feed
P50	X .200	Z .0714	X .250 Part Radius - .050 Thread In Feed Z .100 Clearance - .0286 Thread In Feed
	TOOL	CHANGE	POSITION



DRILL .250 DIAMETER HOLE

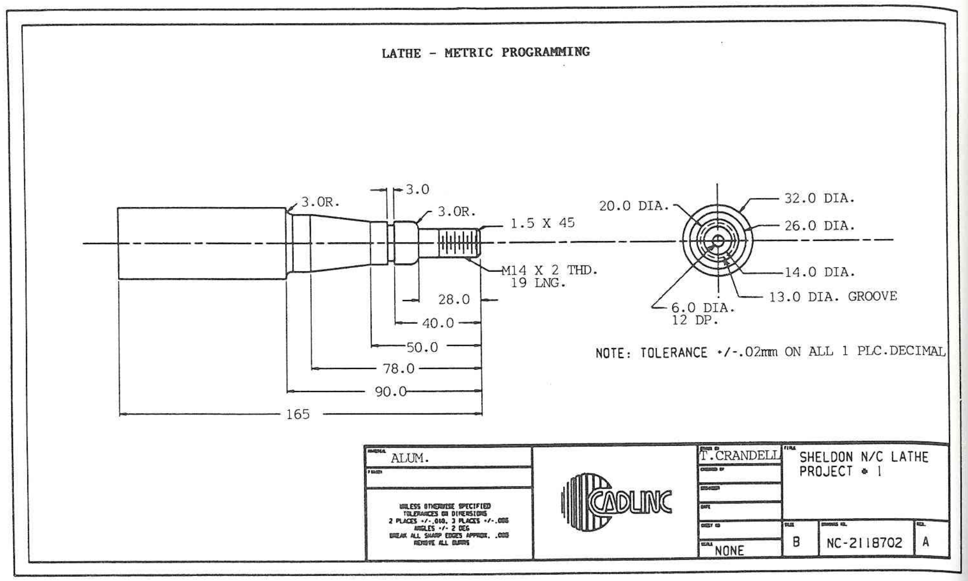
POINT CODE	X COORDINATE	Z COORDINATE	CALCULATION
P58	X 0	Z .100	X 0 Datum Z 0 Datum + .100 Clearance
P59	X 0	Z-.575	Z-.500 Hole Depth - .075 Drill Point .3 X Dia. Drill Point Allowance
P60	X 0	Z .100	X 0 Datum Z 0 Datum + .100 Clearance
P61	TOOL	CHANGE	POSITION

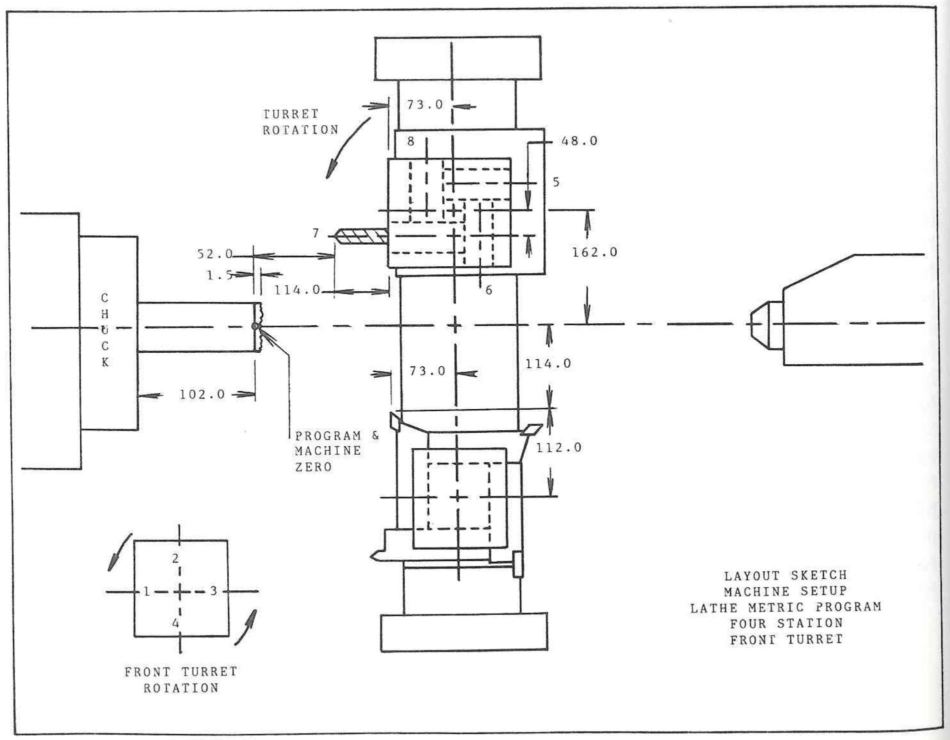
Lathe—Inch Program (Example 1)

```
%          Rewind stop code
N0010 G70     Inch programming
N0020 G97     rpm spindle speed programming
N0030 G95     ipr feedrate programming
N0040 G90     Absolute coordinate programming
N0050 G00 T00     Clear tool offsets
N0060 T0600     Index rear turret to empty station
N0070 T0101 S960 M03     Index tool 1 to cutting position (rough face & turn),
                              start spindle
N0080 G92 X4.5 Z6.56     Preset X & Z axes
N0090 G00 X.725 Z.03 M08     Position tool to rough face, turn coolant on (P1)
N0100 G01 X-.05 F.01     Rough face part leaving 0.030 (P2)
N0110 G00 Z.050     Retract from face (P3)
N0120 X.525     Position X for first rough turn (P4)
N0130 G01 Z-3.406 F.01     Feed for first rough turn (P5)
N0140 G00 X.575 Z-3.356     Retract X & Z axes (P6)
N0150 Z.05 S1132     Rapid to face of part (P7)
N0160 X.4     Position X for second rough turn (P8)
N0170 G01 Z-2.0277 F.01     Feed for second rough turn (P9)
N0180 G00 X.450 Z-1.9777     Retract X & Z axes (P10)
N0190 Z.05 S2000     Rapid to face of part (P11)
N0200 X.265     Position X for semifinish pass (P12)
N0210 G01 Z-1.1164     Feed tangent to convex radius (P13)
N0220 S1538     Change rpm
N0230 G03 X.39 Z-1.281 I.046 K.1647     Contour convex radius (P14)
N0240 G01 Z-2.0283     Feed tangent to taper (P15)
N0250 S1200     Change rpm
N0260 X.515 Z-3.0903     Feed up taper (P16)
N0270 Z-3.406     Feed tangent to concave radius (P17)
N0280 S1010     Change rpm
N0290 G02 X.594 Z-3.485 I.079 K0     Contour concave radius (P18)
N0300 G01 X.750     Feed off part (P19)
N0310 G00 X4.5 Z6.56 T0000     Rapid to tool change position—cancelling tool
                                     offset (P20)
N0320 G92 X0 Z0     Cancel axis presets
N0330 G00 T0202     Index finish facing and turning tool into cutting position
N0340 G90 S2000 M03     Set finishing rpm
N0350 G92 X4.5 Z6.56     Axis presets
N0360 G00 X.365 Z0     Rapid position to start finish face (P21)
N0370 G01 X-.05 F.005     Finish face part (P22)
N0380 G00 X.0718 Z.100     Retract in Z and position for chamfer (P23)
N0390 G01 X.25 Z.0782     Turn chamfer (P24)
N0400 Z-1.1281     Finish turn to convex radius (P25)
N0410 S1600     Change rpm
N0420 G03 X.375 Z-1.281 I.031 K.1529     Contour convex radius (P26)
N0430 G01 Z-2.0292     Finish turn tangent to taper (P27)
N0440 S1200     Change rpm
N0450 X.5 Z-3.0912     Finish turn taper (P28)
N0460 Z-3.406     Finish turn tangent to concave radius (P29)
N0470 S1010     Change rpm
```

```
N0480 G02 X.594 Z-3.5 I.094 K0     Contour concave radius (P30)
N0490 G01 X.750     Feed away from part (P31)
N0500 G00 X4.5 Z6.56 T0000 M09     Rapid to tool change position cancelling tool
                                        offsets and turning off coolant (P32)
N0510 G92 X0 Z0     Cancel axis presets
N0520 G00 T0303     Index grooving tool into cutting position
N0530 G90 S1000 M03     Start spindle
N0540 G92 X4.5 Z6.56     Preset axes
N0550 G00 X.475 Z-1.687 M08     Position in X & Z to cut groove (P33)
N0560 G01 X.3125 F.005     Cut groove to depth (P34)
N0570 G04 F05     Dwell for 5 seconds
N0580 G00 X.475 M09     Retract from groove, turning coolant off (P35)
N0590 X4.5 Z6.56 T0000     Rapid to tool change position cancelling tool offset
                                (P36)
N0600 G92 X0 Z0     Cancel axis presets
N0610 G00 T0404     Index threading tool into cutting position
N0620 G90 S2000 M03     Start spindle
N0630 G92 X4.5 Z6.56     Preset axes
N0640 G00 X.35 Z.1 M08     Position in X & Z to cut thread (P37)
N0650 X.2284 Z.0876     First thread in feed (P38)
N0660 G33 Z-.75 K.0769     Cut first thread pass (P39)
N0670 G00 X.35     Retract X (P40)
N0680 Z.1     Rapid to end of part (P41)
N0690 X.2166 Z.0808     Thread in feed for second pass (P42)
N0700 G33 Z-.75 K.0769     Cut second pass (P43)
N0710 G00 X.35     Retract X (P44)
N0720 Z.1     Rapid to end of part (P45)
N0730 X.2076 Z.0756     Thread in feed for third pass (P46)
N0740 G33 Z-.75 K.0769     Cut third pass (P47)
N0750 G00 X.35     Retract X (P48)
N0760 Z.1     Rapid to end of part (P49)
N0770 X.2 Z.0714     Thread in feed for finish pass (P50)
N0780 G33 Z-.75 K.0769     Cut finish pass (P51)
N0790 G00 X.35     Retract X (P52)
N0800 Z.1     Rapid to end of part (P53)
N0810 X.2 Z.0714     Thread in feed for tool pressure pass (P54)
N0820 G33 Z-.75 K.0769     Cut tool pressure pass (P55)
N0830 G00 X.35 M09     Retract X (P56)
N0840 X4.5 Z6.56 T0000     Rapid to tool change cancelling tool offsets (P57)
N0850 G92 X0 Z0     Cancel axis preset
N0860 G00 T0707     Index 0.250 diameter drill into cutting position
N0870 G90 S2000 M03     Start spindle
N0880 G92 X-4.5 Z2.06     Preset axes
N0890 G00 X0 Z.1 M08     Position to drill (P58)
N0900 G01 Z-.575 F.006     Drill to depth (P59)
N0910 G00 Z.1 M09     Retract drill (P60)
N0920 X-4.5 Z2.06 T0000     Rapid to tool change cancelling tool offsets (P61)
N0930 G92 X0 Z0     Cancel axis presets
N0940 M02     End of program
```

Example 2: Lathe Metric Programming



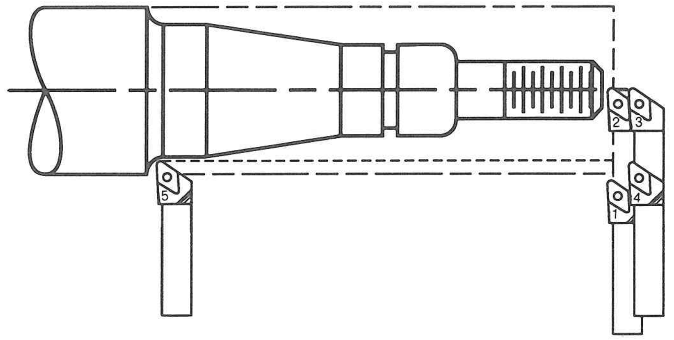


TOOL SHEET

LATHE METRIC PROGRAMMING

PART NO. NC-2118702	MATERIAL ALUM.	OPERATION 20	PROGRAMMED BY T. CRANDELL
PART NAME PROJECT 1	B/P CHANGE DATE		DATE SHEET 1 OF 1

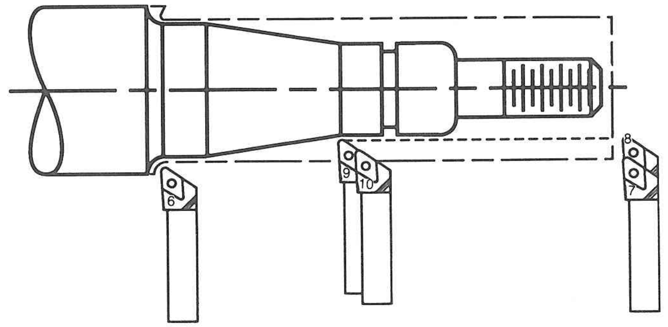
TOOL SEQ.	TOOL NUMBER	TOOL DESCRIPTION	TOOL LENGTH X PROG.	TOOL LENGTH X ACT.	#	TOOL LENGTH Z PROG.	TOOL LENGTH Z ACT.	#	OPERATION DESCRIPTION	SPEED R.P.M.	SPEED CODE	FEED IN/MIN	FEED CODE
	T01	55 DEG. DIAMOND TURNING TOOL (RGH) 0.8 mm TOOL NOSE RADIUS Valenite Holder MDJNR-13-4 Insert DNMP432E VN5	112.0			73.0			RGH. FACE & RGH. TURN	960 to 2000		.25 MMPR	
	T02	55 DEG. DIAMOND TURNING TOOL (FIN) 0.8 mm TOOL NOSE RADIUS Valenite Holder MDJNR-13-4 Insert DNMP432E VN5	112.0			73.0			FIN. FACE & FIN. TURN	2000		.12 MMPR	
	T03	3.0 GROOVE TOOL Valenite Holder SD-TMR-16-3 Insert TNMC32NG VN5	112.0			73.0			3.0 GROOVE	1000		.12 MMPR	
	T04	60 DEG. THREAD TOOL Valenite Holder SD-TMR-16-3 Insert TNMC32NV VC7	112.0			73.0			M14X2 THREAD	1000		2.0 MMPR	
	T07	6.0mm DIA. HSS DRILL	48.0			114.0			DRILL 6.0mm DIA.	2000		.15 MMPR	



COORDINATE CALCULATIONS
LATHE METRIC PROGRAMMING

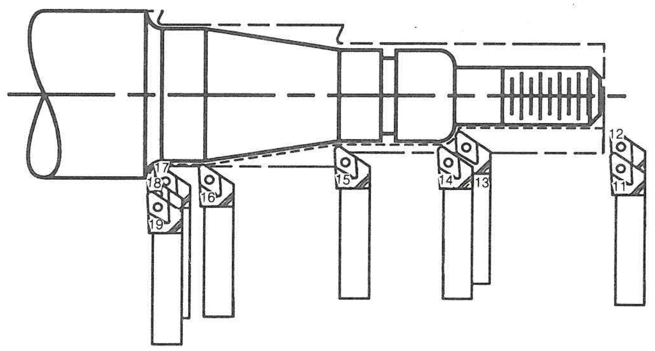
ROUGH FACE & FIRST ROUGH TURN

POINT CODE	X COORDINATE	Z COORDINATE	CALCULATION
P1	X 18.5	Z .75	X 16.0 Stock Radius + 2.5 Clearance Z .75 Finish Stock
P2	X-1.3	Z .75	X-.8 Tool Radius - .5 Cut Past Center Z .75 Finish Stock
P3	X-1.3	Z 3.25	X-.8 - .5 Z .75 Finish Stock + 2.5 Clearance
P4	X 13.6	Z 3.25	X 13.0 Part radius + .6 Stock
P5	X 13.6	Z-87.8	$Z-87.0 - \sqrt{1.6^2 - 1.6^2} - .8$ Formula #1 Appendix E



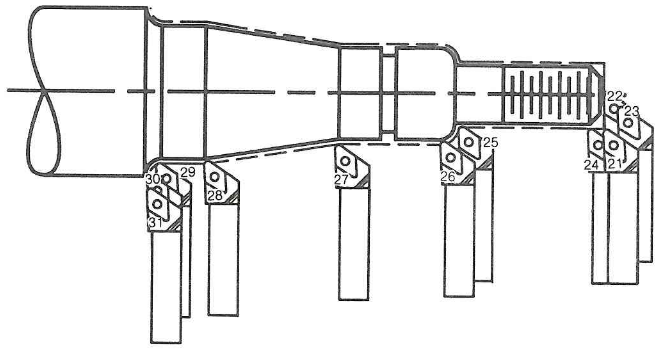
SECOND ROUGH TURN

POINT CODE	X COORDINATE	Z COORDINATE	CALCULATION
P6	X 14.8	Z-86.6	X 13.6 Radius + 1.2 Clearance Z-87.8 Pos."5" + 1.2 Clearance
P7	X 14.8	Z 1.25	Z .75 Finish Stock + .5 Clearance
P8	X 10.6	Z 1.25	X 10.0 Part Radius + .6 Stock
P9	X 10.6	Z-50.72	$Z-50.0 -[.8-(\text{Tan.}\frac{6.7129^\circ}{2}X(.8+.6))]$ Formula #2 Appendix E
P10	X 11.8	Z-49.52	X 10.6 Radius + 1.2 Clearance Z-50.72 Pos."9" + 1.2 Clearance



SEMI FINISH TURN

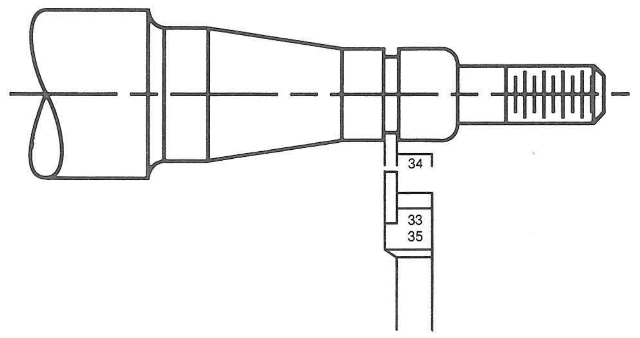
POINT CODE	X COORDINATE	Z COORDINATE	CALCULATION
P11	X 11.8	Z 1.25	Z .75 Finish Stock + .5 Clearance
P12	X 7.4	Z 1.25	X 7.0 Part Radius + .4 Finish Stock
P13	X 7.4	Z-27.78	Z-28.0 - 3.0 + 4.02 - .8 Formula #3 Appendix E
P14	X 10.4 I1.2	Z-31.8 K4.02	X 10.0 Part Radius + .4 Finish Stock Z-31.0 Radius Center - .8 Tool Radius
P15	X 10.4	Z-50.73	Z-50 -[.8-(Tan.$\frac{6.7129^{\circ}}{2}$ X (.8+.4))] Formula #2 Appendix E
P16	X 13.4	Z-78.73	X 13.0 Part Radius + .4 Finish Stock Z-50.73 Pos. 15 - 28.0 Taper Length
P17	X 13.4	Z-87.8	Z-90 Part Dim. + 3.0 Radius - .8 Tool Radius
P18	X 15.2 I1.8	Z-89.6 K0	X 16.0 Part Radius - .8 Tool Radius Z-90 Part Dim. + .4 Finish Stock
P19	X 18.5	Z-89.6	X 16.0 Part Radius + 2.5 Clearance



FINISH FACE AND FINISH TURN

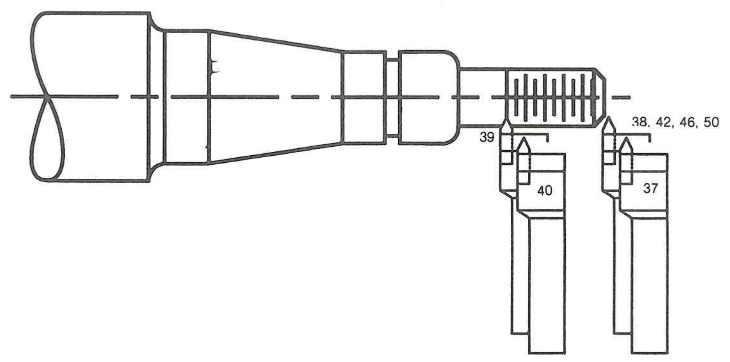
POINT CODE	X COORDINATE	Z COORDINATE	CALCULATION
P20	TOOL	CHANGE	POSITION
P21	X 9.9	Z 0	X 7.0 Part Radius + .4 Stock + 2.5 Clearance Z 0 Finish Datum
P22	X-1.3	Z 0	X 0 Datum - .8 Tool Radius - .5 Past Center
P23	X 2.53	Z 2.5	X 7.0 Part Radius - 1.5 Cham. - 2.5 Clear + .33 Formula #4 Appendix E - .8 Tool Radius Z 0 Datum + 2.5 Clearance
P24	X 7.0	Z-1.97	X 7.0 Part Radius Z-1.5 Chamfer + .33 (formula 4) - .8 Tool Radius
P25	X 7.0	Z-25.39	Z-28.0 - .3 + 3.71 - .8 Formula #3 No Stock Appendix E
P26	X 10.0 I.8	Z-31.8 K3.71	X 10.0 Part Radius Z-28.0 - 3.0 Radius - .8 Tool Radius
P27	X 10.0	Z-50.75	Z-50.0 -[.8-(Tan.$\frac{6.7129^{\circ}}{2}$X .8)] Formula #2 No Stock Appendix E
P28	X 13.0	Z-78.75	X 13.0 Part Radius Z-50.75 Pos. 27 - 28.0 Taper Length

POINT CODE	X COORDINATE	Z COORDINATE	CALCULATION
P29	X 13.0	Z-87.8	Z-90.0 Part Dim. + 3.0 Radius - .8 Tool Radius
P30	X 15.2 I2.2	Z-90.0 K0	X 16.0 Part Radius- .8 Tool Radius Z-90.0 Part Dim.
P31	X 18.5	Z-90.0	X 16.0 Part Radius + 2.5 Clearance
P32	TOOL	CHANGE	POSITION



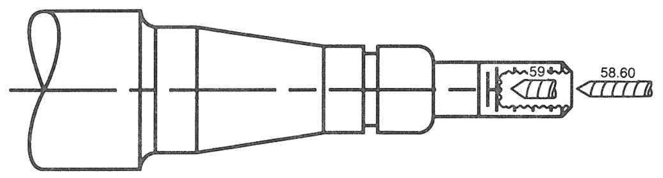
CUT GROOVE

POINT CODE	X COORDINATE	Z COORDINATE	CALCULATION
P33	X 12.5	Z-43.0	X 10.0 Part Radius + 2.5 Clearance Z-40.0 Part Dim. - 3.0 Groove Width
P34	X 6.5	Z-43.0	X 6.5 Groove Radius
P35	X 12.5	Z-43.0	X 10.0 Part Radius + 2.5 Clearance
P36	TOOL	CHANGE	POSITION



THREAD CUTTING

POINT CODE	X COORDINATE	Z COORDINATE	CALCULATION
P37	X 9.5	Z 2.5	X 7.0 Part Radius + 2.5 Clearance Z 0 Finish Datum + 2.5 Clearance
P38	X 6.1	Z 1.98	X 7.0 Part Radius - .9 Thread In Feed Z 2.5 Clearance - .52 Thread In Feed K (Thread Pitch)
P39	X 6.1	Z-19	Z-19 Thread Length Dim.
P40	X 9.5	Z-19	X 7.0 Part Radius + 2.5 Clearance
P42	X 5.4	Z 1.58	X 7.0 Part Radius - 1.6 Thread In Feed Z 2.5 Clearance - .92 Thread In Feed
P46	X 4.9	Z 1.29	X 7.0 Part Radius - 2.1 Thread In Feed Z 2.5 Clearance - 1.21 Thread In Feed
P50	X 4.546	Z 1.09	X 7.0 Part Radius - 2.454 Thread In Feed Z 2.5 Clearance - 1.41 Thread In Feed
	TOOL	CHANGE	POSITION



DRILL 6MM DIAMETER HOLE

POINT CODE	X COORDINATE	Z COORDINATE	CALCULATION
P58	X 0	Z 2.5	X 0 Datum Z 0 Datum + 2.5 Clearance
P59	X 0	Z-13.8	Z-12 Hole Depth - 1.8 Drill Point .3 X Dia. Drill Point Allowance
P60	X 0	Z 2.5	X 0 Datum Z 0 Datum + 2.5 Clearance
P61	TOOL	CHANGE	POSITION

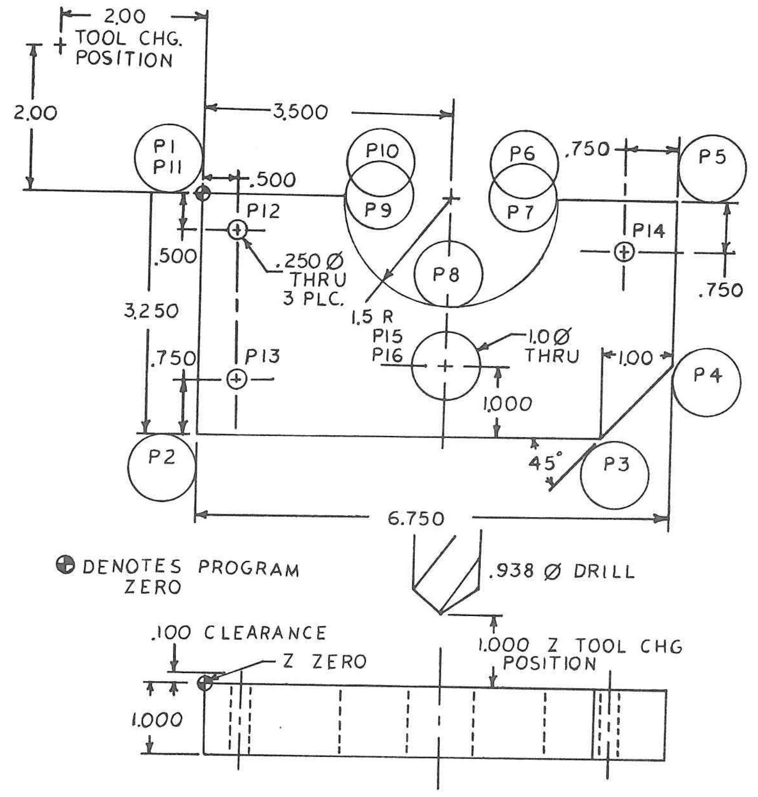
Lathe—Metric Program

```
%       Rewind stop code
N0010 G71       Metric programming
N0020 G97       rpm spindle speed programming
N0030 G95       mmpr feedrate programming
N0040 G90       Absolute coordinate programming
N0050 G00 T00       Clear tool offsets
N0060 T0600       Index rear turret to empty station
N0070 T0101 S960 M03       Index tool 1 to cutting position (rough face and turn),
                             start spindle
N0080 G92 X114.0 Z166.0       Preset X & Z axes
N0090 G00 X18.5 Z.75 M08       Position tool to rough face, turn coolant on (P1)
N0100 G01 X-1.3 F.25       Rough face part leaving 0.5 stock (P2)
N0110 G00 Z3.25       Retract from face (P3)
N0120 X13.6       Position X for first rough turn (P4)
N0130 G01 Z-87.8 F.25       Feed for first rough turn (P5)
N0140 G00 X14.8 Z-86.6       Retract X & Z axes (P6)
N0150 Z1.25 S1132       Rapid to face of part (P7)
N0160 X10.6       Position X for second rough turn (P8)
N0170 G01 Z-50.72 F.25       Feed for second rough turn (P9)
N0180 G00 X11.8 Z-49.52       Retract X & Z axes (P10)
N0190 Z1.25 S2000       Rapid to face of part (P11)
N0200 X7.4       Position X for semifinish pass (P12)
N0210 G01 Z-27.78       Feed tangent to convex radius (P13)
N0220 S1538       Change rpm
N0230 G03 X10.4 Z-31.8 I1.2 K4.02       Contour convex radius (P14)
N0240 G01 Z-50.73       Feed tangent to taper (P15)
N0250 S1200       Change rpm
N0260 X13.4 Z-78.73       Feed up taper (P16)
N0270 Z-87.8       Feed tangent to concave radius (P17)
N0280 S1010       Change rpm
N0290 G02 X15.2 Z-89.6 I1.8 K0       Contour concave radius (P18)
N0300 G01 X18.5       Feed off part (P19)
N0310 G00 X114.0 Z166.0 T0000       Rapid to tool change position—cancelling
                                      tool offset (P20)
N0320 G92 X0 Z0       Cancel axis presets
N0330 G00 T0202       Index finish facing and turning tool into cutting position
N0340 G90 S2000 M03       Set finishing rpm
N0350 G92 X114.0 Z166.0       Axis presets
N0360 G00 X9.9 Z0       Rapid to start finish face (P21)
N0370 G01 X-1.3 F.12       Finish face part (P22)
N0380 G00 X2.53 Z2.5       Retract in Z and position for chamfer (P23)
N0390 G01 X7.0 Z-1.97       Turn chamfer (P24)
N0400 Z-25.39       Finish turn to convex radius (P25)
N0410 S1600       Change rpm
N0420 G03 X10.0 Z-31.8 I.8 K3.71       Contour convex radius (P26)
N0430 G01 Z-50.75       Finish turn tangent to taper (P27)
N0440 S1200       Change rpm
N0450 X13.0 Z-78.75       Finish turn taper (P28)
N0460 Z-87.8       Finish turn tangent to concave radius (P29)
N0470 S1010       Change rpm
```

```
N0480 G02 X15.2 Z-90.0 I2.2 K0      Contour concave radius (P30)
N0490 G01 X18.5      Feed away from part (P31)
N0500 G00 X114.0 Z166.0 T0000 M09      Rapid to tool change position cancelling
                                        tool offsets and turning off coolant (P32)
N0510 G92 X0 Z0      Cancel axis presets
N0520 G00 T0303      Index grooving tool into cutting position
N0530 G90 S1000 M03      Start spindle
N0540 G92 X114.0 Z166.0      Preset axes
N0550 G00 X12.5 Z-43.0 M08      Position in X & Z to cut groove (P33)
N0560 G01 X6.5 F.12      Cut groove to depth (P34)
N0570 G04 F05      Dwell for 5 sec
N0580 G00 X12.5 M09      Retract from groove turning coolant off (P35)
N0590 G00 X114.0 Z166.0 T0000      Rapid to tool change position cancelling tool
                                    offsets (P36)
N0600 G92 X0 Z0      Cancel axis presets
N0610 G00 T0404      Index threading tool into cutting position
N0620 G90 S1000 M03      Start spindle
N0630 G92 X114.0 Z166.0      Preset axes
N0640 G00 X9.5 Z2.5 M08      Position in X & Z to cut thread (P37)
N0650 X6.1 Z1.98      First thread in feed (P38)
N0660 G33 Z-19.0 K2.0      Cut first thread pass (P39)
N0670 G00 X9.5      Retract X (P40)
N0680 Z2.5      Rapid to end of part (P41)
N0690 X5.4 Z1.58      Thread in feed for second pass (P42)
N0700 G33 Z-19.0 K2.0      Cut second pass (P43)
N0710 G00 X9.5      Retract X (P44)
N0720 Z2.5      Rapid to end of part (P45)
N0730 X4.9 Z1.29      Thread in feed for third pass (P46)
N0740 G33 Z-19.0 K2.0      Cut third pass (P47)
N0750 G00 X9.5      Retract X (P48)
N0760 Z2.5      Rapid to end of part (P49)
N0770 X4.546 Z1.09      Thread in feed for finish pass (P50)
N0780 G33 Z-19.0 K2.0      Cut finish pass (P51)
N0790 G00 X9.5      Retract X (P52)
N0800 Z2.5      Rapid to end of part (P53)
N0810 X4.546 Z1.09      Thread in feed for tool pressure pass (P54)
N0820 G33 Z-19.0 K2.0      Cut tool pressure pass (P55)
N0830 G00 X9.5 M09      Retract X (P56)
N0840 X114.0 Z166.0 T0000      Rapid to tool change cancelling tool offsets (P57)
N0850 G92 X0 Z0      Cancel axis preset
N0860 G00 T0707      Index 6.0 mm drill into cutting position
N0870 G90 S2000 M03      Start spindle
N0880 G92 X-114.0 Z52.0      Preset axes
N0890 G00 X0 Z2.5 M08      Position to drill (P58)
N0900 G01 Z-13.8 F.15      Drill to depth (P59)
N0910 G00 Z2.5 M09      Retract drill (P60)
N0920 X-114.0 Z166.0 T0000      Rapid to tool change cancelling tool offsets (P61)
N0930 G92 X0 Z0      Cancel axis presets
N0940 M02      End of program
```

Example 3: Milling, Inch

MILL EXAMPLE INCH



MACHINING CENTER - INCH PROGRAMMING

TOOL SHEET

PART NO. NC-2118906	MATERIAL 1018 M.S.	OPERATION 20	PROGRAMMED BY T. CRANDELL
PART NAME MILL EXAMPLE	B/P CHANGE DATE		DATE / SHEET 1 OF 1

TOOL		TOOL	TOOL DIAMETER			TOOL LENGTH			OPERATION	SPEED		FEED	
SEQ.	NUMBER	DESCRIPTION	PROG.	ACT.	#	PROG.	ACT.	#	DESCRIPTION	R. P. M.	CODE	IN/MIN	CODE
	01	1.0 DIA. FOUR FLUTE ENDMILL #56779 HSS National Twist HOLDER # QC40-EM100-338 Kennametal	1.000			5.500 PROJECTION 2.00 MIN.			MILL PERIPHERY OF PART LOC. 1 THRU 10	400 RPM		11.2	IPM
	02	.250 DIA. DRILL HSS #49016 National Twist HOLDER # QC40-DA300-163 Kennametal COLLECT # 300DA-0250	.250			4.500 PROJECTION 2.5 MIN.			DRILL .(3) .250 DIA. HOLES THRU LOC. 11 THRU 13	1600 RPM		9.6	IPM
	03	15/16 DIA. DRILL HSS #10472 National Twist HOLDER #2-60-209-303 Kennametal	.938			7.000 PROJECTION 3.5 MIN.			DRILL .937 DIA. HOLE THRU LOC. 14	425 RPM		8.5	IPM
	04	1.000 DIA. BORING TOOL CARB. BORING BAR # 2-80-076-026 Kennametal Insert #E32ACP HOLDER # QC40-TG150-450 Kennametal	1.000			6.000 PROJECTION 2.5 MIN.			FIN. BORE 1.0 DIA. HOLE THRU LOC. 15	1200 RPM		7.2	IPM

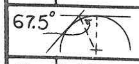
INCH
MACHINING CENTER

COORDINATE SHEET

COMPANY FERRIS STATE UNIVERSITY

DATE

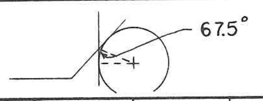
PART NO. NC-2118906	PART NAME MILL EXAMPLE	OPERATION NO. 20	BY CRANDELL SHEET 1 OF 5

POINT CODE	X ABSOLUTE COORDINATE	Y ABSOLUTE COORDINATE	TABLE DEG. ABS.	TOOL DESCRIPTION	T SETTING LENGTH	D DEPTH BELOW WORK SUR.	S WORK SUR.	Z TOOL CHG.	Z CLEAR POS.	Z DEPTH POS.
								SET TOOL – PRESENT + TOOL CHG. CLEARANCE	WORK SUR. + CLEARANCE	WORK SUR. – FEATURE DP. – TOOL POINT
TOOL CHG.	-2.000	+2.000		T01 1.0 END MILL	7.000	0	0	1.000	1.000	-
	X 0 Datum - 2.000 Cutter Center Position Y 0 Datum + 2.000 Cutter Center Position				Tool Chg. 7.000 - 7.000 + 1.000 = 1.000 Clear Pos. 0 + 1.000 = 1.00					
P1	-.500	+.500			5.500	1.100	0	2.500	.100	-1.100
	X 0 Datum - .500 Cutter Radius Y 0 Datum + .500 Cutter Radius				Tool Chg. 7.000 - 5.500 + 1.000 = 2.500 Clear Pos. 0 + .100 = .100 Z Depth 0-1.0 Part Thickness-.100 Clear=-1.100					
P2	-.500	-3.750								
	Y 0 Datum - 3.750 Part Width - .500 Cutter Radius									
P3	5.9413	-3.750								
	X 0 + 6.75 Part Length - 1.00 Angle Length + COTAN 67.5° * .5 Cutter Center = 5.75 +.2071 = 5.9571									
P4	7.250	-2.4571								
	X 0 Datum + 6.75 Part Length + .500 Cutter Radius = 7.250 Y 0 Datum - 3.25 Part Width + 1.00 Angle Width Length - COTAN 67.5° * .5 Cutter Center = -2.25 - .2071 = -2.4571 Continued									

INCH
MACHINING CENTER

COORDINATE SHEET

COMPANY FERRIS STATE UNIVERSITY — DATE

PART NO. NC-2118906				PART NAME MILL EXAMPLE	OPERATION NO. 20			BY CRANDELL	SHEET 2 OF 5	
POINT CODE	X ABSOLUTE COORDINATE	Y ABSOLUTE COORDINATE	TABLE DEG. ABS.	TOOL DESCRIPTION	T SETTING LENGTH	D DEPTH BELOW WORK SUR.	S WORK SUR.	Z TOOL CHG.	Z CLEAR POS.	Z DEPTH POS.
				Note: The longest tool (.938 dia. drill 7.00 gage length) is used for setting the Z tool chg.				SET TOOL − PRESENT + TOOL CHG. CLEAR-ANCE	WORK SUR. + CLEAR-ANCE	WORK SUR. − FEATURE DP. − TOOL POINT
P5	7.250	.500								
	X 0 Datum + 6.750 Part Length + .500 Cutter Radius Y 0 Datum + .500 Cutter Radius									
P6	4.500	.500								
	X 0 Datum + 3.500 Part Radius Center + 1.500 Part Radius - .500 Cutter Radius									
P7	4.500	0								
	Y 0 Datum									
P8	3.500	-1.000		I 1.0 J0						
	X 0 Datum + 3.500 Part Radius Center Y 0 - 1.500 Part Radius + .500 Cutter Radius				I 1.500 Part radius - .500 Cutter Radius					
P9	2.500	0		I 0 J1.0						
	X 0 Datum + 3.500 Part Radius Center - 1.500 Part Radius + .500 Tool Radius Y 0 Datum				J 1.500 Part Radius - .500 Cutter Radius					

INCH
MACHINING CENTER

COORDINATE SHEET

COMPANY FERRIS STATE UNIVERSITY — DATE

PART NO. NC-2118906				PART NAME MILL EXAMPLE	OPERATION NO. 20			BY CRANDELL	SHEET 3 OF 5	
POINT CODE	X ABSOLUTE COORDINATE	Y ABSOLUTE COORDINATE	TABLE DEG. ABS.	TOOL DESCRIPTION	T SETTING LENGTH	D DEPTH BELOW WORK SUR.	S WORK SUR.	Z TOOL CHG.	Z CLEAR POS.	Z DEPTH POS.
								SET TOOL − PRESENT + TOOL CHG. CLEAR-ANCE	WORK SUR. + CLEAR-ANCE	WORK SUR. − FEATURE DP. − TOOL POINT
P10	2.500	.500								
	Y 0 Datum + .500 Cutter Radius									
P11	-.500	.500								
	X 0 Datum - .500 Cutter Radius Y 0 Datum + .500 Cutter Radius									
TOOL CHG.	-2.000	2.000		T02 .250 DIA. DRILL	5.500 TL.#1	0	0	2.500		
	X 0 Datum - 2.000 Tool Change Position Y 0 Datum + 2.000 Tool Change Position				Z 7.000 Set Tl. - 5.500 Present Tl.#1 + 1.000 Clearance = 2.500					
P12	.500	-.500			4.500 TL.#2	1.175	0	3.500	.100	-1.175
	X 0 Datum + .500 Hole Center Dim. Y 0 Datum - .500 Hole Center Dim.				Tool Chg. 7.000 - 4.500 + 1.000 = 3.500 Clear Pos. 0 + .100 = .100 Z Depth 0 - 1.00 - .175 = -1.175					
P13	.500	-2.500								
	Y 0 Datum - 3.250 Part Width + .750 Hole Center Dim.									

INCH
MACHINING CENTER
COORDINATE SHEET

COMPANY FERRIS STATE UNIVERSITY | DATE

PART NO. NC-2118906				PART NAME MILL EXAMPLE	OPERATION NO. 20			BY CRANDELL	SHEET 4	OF 5
POINT CODE	X ABSOLUTE COORDINATE	Y ABSOLUTE COORDINATE	TABLE DEG. ABS.	TOOL DESCRIPTION	T SETTING LENGTH	D DEPTH BELOW WORK SUR.	S WORK SUR.	Z TOOL CHG.	Z CLEAR POS.	Z DEPTH POS.
								SET TOOL – PRESENT + TOOL CHG. CLEAR-ANCE	WORK SUR. + CLEAR-ANCE	WORK SUR. – FEATURE DP. – TOOL POINT
P14	6.000	-.750								
	X 0 Datum + 6.750 Part Length - .75 Hole Center Dim. Y 0 Datum - .750 Hole Center Dim.									
TOOL CHG.	-2.000	2.000		T03 .938 DIA. DRILL	4.500 TL.#2	0	0	3.500		
	X 0 Datum - 2.000 Tool Change Position Y 0 Datum + 2.000 Tool Change Position							Z 7.000 Set TL. - 4.500 Present Tl.#2 + 1.000 Clearance = 3.500		
P15	3.500	-2.250			7.000 TL.#3	1.3814	0	1.000	.100	-1.3814
	X 0 Datum + 3.500 Hole Center Dim. Y 0 Datum - 3.250 Part Width + 1.000 Hole Center Dim.							Tool Chg. 7.00 - 7.00 + 1.00 = 1.000 Clear Pos. 0 + .100 = .100 Z Depth 0 - 1.00 - .2814 - .100 = -1.3814		
TOOL CHG.	-2.000	2.000		T04 1.000 DIA. BORING BAR	7.000 TL#3	0	0	1.000		
	X 0 Datum - 2.000 Tool Change Position Y 0 Datum 2.000 Tool Change Position							Z 7.000 Set TI. -7.000 Present TI.#3 + 1.000 Clearance = 1.000		
P16	3.500	-2.250			6.000 TL#4	1.100	0	2.000	.100	-1.100
	X 0 Datum + 3.500 Hole Center Dim. Y 0 Datum - 3.250 Part Width + 1.000 Hole Center Dim.							Tool Chg. 7.00 - 6.00 + 1.00 = 2.000 Clear Pos. 0 + .100 = .100 Z Depth 0 - 1.00 - .100 = -1.100		

INCH
MACHINING CENTER
COORDINATE SHEET

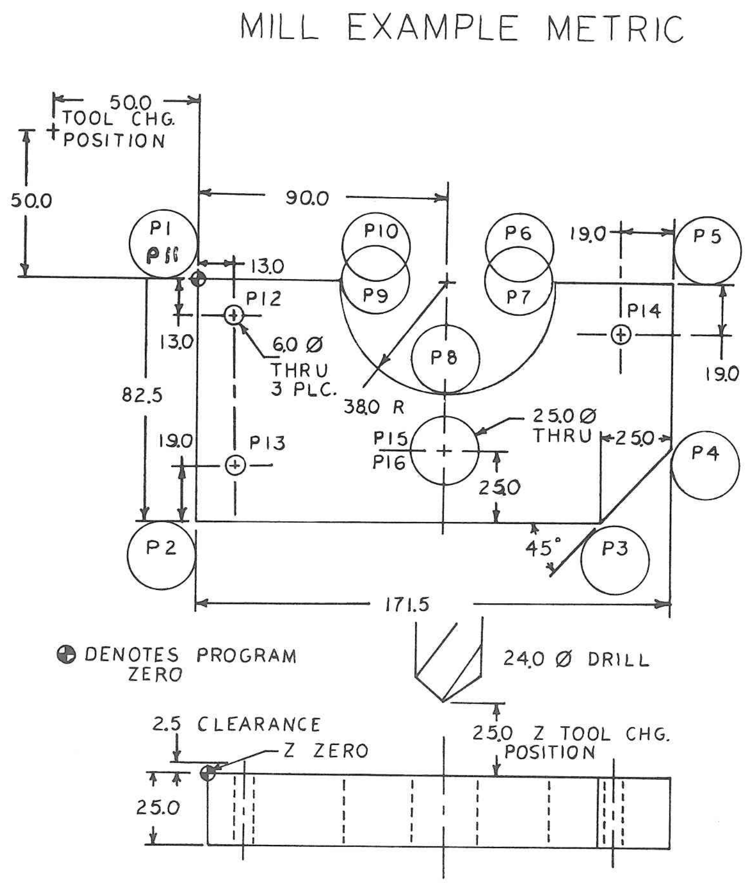
COMPANY FERRIS STATE UNIVERSITY | DATE

PART NO. NC-2118906				PART NAME MILL EXAMPLE	OPERATION NO. 20			BY CRANDELL	SHEET 5	OF 5
POINT CODE	X ABSOLUTE COORDINATE	Y ABSOLUTE COORDINATE	TABLE DEG. ABS.	TOOL DESCRIPTION	T SETTING LENGTH	D DEPTH BELOW WORK SUR.	S WORK SUR.	Z TOOL CHG.	Z CLEAR POS.	Z DEPTH POS.
								SET TOOL – PRESENT + TOOL CHG. CLEAR-ANCE	WORK SUR. + CLEAR-ANCE	WORK SUR. – FEATURE DP. – TOOL POINT
TOOL CHG.	-2.000	2.000			6.000 TL#4	0	0	2.000		
	X 0 Datum - 2.000 Tool Change Position Y 0 Datum - 2.000 Tool Change Position							Z 7.000 Set Tl. - 6.000 Present Tl.#4 + 1.000 Clearance = 2.000		

Inch Machining Center Example Program

```
N0010 G90       Absolute programming
N0020 G70       Inch programming
N0030 G94       Inch per minute feedrate
N0040 G17       X-Y circular interpolation plane
N0050 G40 T00       Cancel cutter diameter compensation and tool length com-
                    pensation
N0060 G80 Z1.0      Retract Z axis
N0070 G00 X-2.0 Y2.0 T01 M06      Tool change "1.0 end mill"
N0080 X-.5 Y.5 Z.1 S400 M03     Rapid to position 1, start spindle
N0090 Z-1.1 M08      Rapid to Z depth, turn coolant on
N0100 G01 Y-3.75 F11.2      Feed to position 2
N0110 X5.9413      Feed to position 3
N0120 X7.25 Y-2.4571      Feed to position 4
N0130 Y.5      Feed to position 5
N0140 X4.5      Feed to position 6
N0150 Y0      Feed to position 7
N0160 G02 X3.5 Y-1.0 I1.0 J0      Circular interpolate to position 8
N0170 X2.5 Y0 I0 J1.0      Circular interpolate to position 9
N0180 G01 Y.5      Feed to position 10
N0190 X-.5      Feed to position 11
N0200 G00 Z.1 M09      Clear part
N0210 G80 X-2.0 Y2.0 Z2.5 T02 M06      Tool change ".250 dia. drill"
N0220 X.5 Y-.5 Z.1 S1600 M03      Rapid to position 12 turn, spindle on
N0230 G81 Z-1.175 R.1 F9.6 M08      Drill position 12, turn coolant on
N0240 Y-2.5      Drill position 13
N0250 X6.0 Y-.75 M09      Drill position 14, turn coolant off
N0260 G80 X-2.0 Y2.0 Z3.5 T03 M06      Tool change ".938 dia drill"
N0270 X3.5 Y-2.25 Z.1 S425 M03      Rapid to position 15, turn spindle on
N0280 G81 Z-1.3814 R.1 F8.5 M08      Drill position 15, turn coolant on
N0290 M09      Turn coolant off
N0300 G80 X-2.0 Y2.0 Z1.0 T04 M06      Tool change "1.0 dia. boring bar"
N0310 X3.5 Y-2.25 Z.1 S1200 M03      Rapid to position 16, turn spindle on
N0320 G81 Z-1.1 R.1 F7.2 M08      Bore position 16, turn coolant on
N0330 M09      Turn coolant off
N0340 G80 X-2.0 Y2.0 Z2.0 M02      Rapid to tool change position and end pro-
                                   gram
```

Example 4: Milling, Metric



MACHINING CENTER - METRIC PROGRAMMING

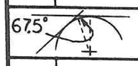
TOOL SHEET

PART NO. NC-2118906			MATERIAL 1018 M.S.			OPERATION 20			PROGRAMMED BY T. CRANDELL			
PART NAME MILL EXAMPLE			B/P CHANGE DATE						DATE		SHEET 1 OF 1	
TOOL		TOOL	TOOL DIAMETER			TOOL LENGTH			OPERATION	SPEED		FEED	
SEQ.	NUMBER	DESCRIPTION	PROG.	ACT.	#	PROG.	ACT.	#	DESCRIPTION	R. P. M.	CODE	IN/MIN	CODE
	01	24.0 DIA. FOUR FLUTE ENDMILL # D3242-EDP244449 HSS Vern Wheeler Co. HOLDER # QC40-EM100-338 Kennametal	24.0MM			140.0MM PROJECTION 50.0MM MIN.			MILL PERIPHERY OF PART LOC. 1 THRU 10	400 RPM		284	MM/ MIN.
	02	6.0 DIA. DRILL HSS #12695 National Twist HOLDER # QC40-DA300-163 Kennametal COLLECT #300DA-0250	6.0MM			114.0MM PROJECTION 64MM MIN.			DRILL (3) 6.0 DIA. HOLES THRU LOC. 11 THRU 13	1600 RPM		244	MM/ MIN.
	03	24.0 DIA. DRILL HSS # HOLDER #	24.0MM			178.0MM PROJECTION 90MM MIN.			DRILL 24.0 DIA. HOLE LOC. 14	425 RPM		216	MM/ MIN.
	04	25.0 DIA BORING TOOL CARB. BORING BAR # 2-80-076-026 Kennametal INSERT #E32ACP HOLDER # QC40-TG150-450 Kennametal	25.0MM			152.0MM PROJECTION 64MM MIN.			FIN. BORE 25.0 DIA. HOLE THRU LOC. 15	1200 RPM		183	MM/ MIN.

METRIC
MACHINING CENTER

COORDINATE SHEET

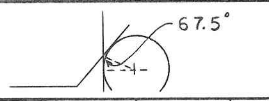
COMPANY FERRIS STATE UNIVERSITY DATE

PART NO. NC-2118906			PART NAME MILL EXAMPLE		OPERATION NO. 20			BY CRANDELL	SHEET 1	OF 5
POINT CODE	X ABSOLUTE COORDINATE	Y ABSOLUTE COORDINATE	TABLE DEG. ABS.	TOOL DESCRIPTION	T SETTING LENGTH	D DEPTH BELOW WORK SUR.	S WORK SUR.	Z TOOL CHG.	Z CLEAR POS.	Z DEPTH POS.
								SET TOOL – PRESENT + TOOL CHG. CLEAR-ANCE	WORK SUR. + CLEAR-ANCE	WORK SUR. – FEATURE DP. – TOOL POINT
TOOL CHG.	-50.0	+50.0		T01 25.0 ENDMILL	178.0 SET TOOL	0	0	25.0	25.0	
	X 0 Datum - 50.0 Cutter Center Position Y 0 Datum + 50.0 Cutter Center Position				Tool Chg. 178.0 - 178.0 + 25.0 = 25.0 Clear Pos. 0 + 25.0 = 25.0					
P1	-12.5	+12.5			140.0	27.5	0	63.0	2.5	-27.5
	X 0 Datum - 12.5 Cutter Radius Y 0 Datum - 12.5 Cutter Radius				Tool Chg. 178 - 140 + 25.0 = 63.0 Clear Pos. 0 + 2.5 = 2.5 Z Depth 0 - 25.0 Part Thickness -2.5 Clear=-27.5					
P2	-12.5	-95.0								
	Y 0 Datum - 82.5 Part Width - 12.5 Cutter Radius									
P3	151.68	-95.0								
 67.5°	X 0 + 171.5 Part Length - 25.0 Angle Length + COTAN 67.5° * 12.5 Cutter Center = 146.5 + 5.18 = 151.68									
P4	184.0	-62.68								
	X 0 Datum + 171.5 Part Length + 12.5 Cutter Radius = 184.0 Y 0 Datum - 82.5 Part Width + 25.0 Angle Length - COTAN 67.5° * 12.5 Cutter Center = -57.5 - 5.18 = -62.68 Continued									

METRIC
MACHINING CENTER

COORDINATE SHEET

COMPANY FERRIS STATE UNIVERSITY | DATE

PART NO. NC-2118906				PART NAME MILL EXAMPLE	OPERATION NO. 20			BY CRANDELL	SHEET 2	OF 5
POINT CODE	X ABSOLUTE COORDINATE	Y ABSOLUTE COORDINATE	TABLE DEG. ABS.	TOOL DESCRIPTION	T SETTING LENGTH	D DEPTH BELOW WORK SUR.	S WORK SUR.	Z TOOL CHG.	Z CLEAR POS.	Z DEPTH POS.
 67.5°				Note: The longest tool (24.0 dia. drill 178.0 gage length) is used for setting the Z tool chg.				SET TOOL − PRESENT + TOOL CHG. CLEARANCE	WORK SUR. + CLEARANCE	WORK SUR. − FEATURE DP. − TOOL POINT
P5	184.0	12.5								
X 0 Datum + 171.5 Part Length + 12.5 Cutter Radius Y 0 Datum + 12.5 Cutter Radius										
P6	115.5	12.5								
X 0 Datum + 90.0 Part Radius Center + 38.0 Part Radius - 12.5 Cutter Radius										
P7	115.5	0								
Y 0 Datum										
P8	90.0	-25.5		I 25.5 J0						
X 0 Datum + 90.0 Part Radius Center Y 0 Datum - 38.0 Part Radius + 12.5 Cutter Radius					I 38.0 Part Radius - 12.5 Cutter Radius					
P9	64.5	0		I0 J 25.5						
X 0 Datum + 90.0 Part Radius Center - 38.0 Part Radius + 12.5 Cutter Radius Y 0 Datum					J 38.0 Part Radius - 12.5 Cutter Radius					

METRIC
MACHINING CENTER

COORDINATE SHEET

COMPANY FERRIS STATE UNIVERSITY | DATE

PART NO. NC-2118906				PART NAME MILL EXAMPLE	OPERATION NO. 20			BY CRANDELL	SHEET 3	OF 5
POINT CODE	X ABSOLUTE COORDINATE	Y ABSOLUTE COORDINATE	TABLE DEG. ABS.	TOOL DESCRIPTION	T SETTING LENGTH	D DEPTH BELOW WORK SUR.	S WORK SUR.	Z TOOL CHG.	Z CLEAR POS.	Z DEPTH POS.
								SET TOOL − PRESENT + TOOL CHG. CLEARANCE	WORK SUR. + CLEARANCE	WORK SUR. − FEATURE DP. − TOOL POINT
P10	64.5	12.5								
Y 0 Datum + 12.5 Cutter Radius										
P11	-12.5	12.5								
X 0 Datum - 12.5 Cutter Radius Y 0 Datum + 12.5 Cutter Radius										
TOOL CHG.	-50.0	50.0		T02 6.0 DIA. DRILL	140.0 TL. #1	0	0	63.0		
X 0 Datum - 50.0 Tool Chg. Position Y 0 Datum + 50.0 Tool Chg. Position					Z 178.0 Set TL. - 140.0 Present TL.#1 + 25.0 Clearance = 63.0					
P12	13.0	-13.0			114.0 TL.#2	29.3	0	89.0	2.5	-29.3
X 0 Datum + 13.0 Hole Center Dim. Y 0 Datum + 13.0 Hole Center Dim.					Tool Chg. 178.0 -114.0 + 25.0 = 89.0 Clear Pos. 0 + 2.5 = 2.5 Z Depth 0 - 25 - 4.3 = -29.3					
P13	13.0	-63.5								
Y 0 Datum - 82.5 Part Width + 19.0 Hole Center Dim.										

METRIC
MACHINING CENTER

COORDINATE SHEET

COMPANY FERRIS STATE UNIVERSITY DATE

PART NO. NC-2118906				PART NAME MILL EXAMPLE	OPERATION NO. 20			BY CRANDELL	SHEET 4 OF 5	
POINT CODE	X	Y	TABLE	TOOL	T	D	S	Z	Z	Z
	ABSOLUTE COORDINATE	ABSOLUTE COORDINATE	DEG. ABS.	DESCRIPTION	SETTING LENGTH	DEPTH BELOW WORK SUR.	WORK SUR.	TOOL CHG.	CLEAR POS.	DEPTH POS.
								SET TOOL – PRESENT + TOOL CHG. CLEARANCE	WORK SUR. + CLEARANCE	WORK SUR. — FEATURE DP. — TOOL POINT
P14	152.5	-19.0								
	X 0 Datum + 171.5 Part Length - 19.0 Hole Center Dim. Y 0 Datum - 19.0 Hole Center Dim.									
TOOL CHG.	-50.0	50.0		T03 24.0 DIA. DRILL	114.0 TL.#2	0	0	89.0		
	X 0 Datum - 50.0 Tool Chg. Position Y 0 Datum + 50.0 Tool Chg. Position							Z 178.0 Set TL. - 114.0 Present TL. #2 + 25.0 Clearance = 89.0		
P15	90.0	-57.5			178.0 TL.#3	34.7	0	25.0	2.5	-34.7
	X 0 Datum + 90.0 Hole Center Dim. Y 0 Datum - 82.5 Part Width + 25.0 Hole Center Dim.							Tool Chg. 178.0 - 178.0 + 25.0 = 25.0 Clear Pos. 0 + 2.5 = 2.5 Z Depth 0 - 25.0 - 7.2 - 2.5 = -34.7		
TOOL CHG.	-50.0	50.0		T04 25.0 DIA. BORING BAR	178.0 TL.#3	0	0	25.0		
	X 0 Datum - 50.0 Tool Change Position Y 0 Datum+50.0 Tool Change Position							Z 178.0 Set TL. -178.0 Present TL.#3 + 25.0 Clearance = 25.0		
P16	90.0	-57.5			152.0 TL.#4	27.5	0	51.0	2.5	-27.5
	X 0 Datum + 90.0 Hole Center Dim. Y 0 Datum - 82.5 Part Width + 25.0 Hole Center Dim.							Tool Chg. 178.0 - 152.0 + 25.0 =51.0 Clear Pos. 0 + 2.5 = 2.5 Z Depth 0 - 25.0 - 2.5 = -27.5		

METRIC
MACHINING CENTER

COORDINATE SHEET

COMPANY FERRIS STATE UNIVERSITY DATE

PART NO. NC-2118906				PART NAME MILL EXAMPLE	OPERATION NO. 20			BY CRANDELL	SHEET 5 OF 5	
POINT CODE	X	Y	TABLE	TOOL	T	D	S	Z	Z	Z
	ABSOLUTE COORDINATE	ABSOLUTE COORDINATE	DEG. ABS.	DESCRIPTION	SETTING LENGTH	DEPTH BELOW WORK SUR.	WORK SUR.	TOOL CHG.	CLEAR POS.	DEPTH POS.
								SET TOOL – PRESENT + TOOL CHG. CLEARANCE	WORK SUR. + CLEARANCE	WORK SUR. — FEATURE DP. — TOOL POINT
TOOL CHG.	-50.0	50.0			152.0 TL.#4	0	0	51.0		
	X 0 Datum - 50.0 Tool Change Position Y 0 Datum+50.0 Tool Change Position							Z 178.0 Set TL. - 152.0 Present TL.#4 + 25.0 Clearance = 51.0		

Metric Machining Center Example Program

```
N0010 G90     Absolute programming
N0020 G71     Metric programming
N0030 G94     Millimeter per minute feedrate
N0040 G17     X-Y circular interpolation plane
N0050 G40 T00     Cancel cutter diameter compensation and tool length com-
                      pensation
N0060 G80 Z25.0     Retract Z axis
N0070 G00 X-50.0 Y50.0 T01 M06     Tool change "25.0 endmill"
N0080 X-12.5 Y12.5 Z2.5 S400 M03     Rapid to position 1, start spindle
N0090 Z-27.5 M08     Rapid to Z depth turn coolant on
N0100 G01 Y-95.0 F284     Feed to position 2
N0110 X151.68     Feed to position 3
N0120 X184.0 Y-62.68     Feed to position 4
N0130 Y12.5     Feed to position 5
N0140 X115.5     Feed to position 6
N0150 Y0     Feed to position 7
N0160 G02 X90.0 Y-25.5 I25.5 J0     Circular interpolate to position 8
N0170 X64.5 Y0 I0 J25.5     Circular interpolate to position 9
N0180 G01 Y12.5     Feed to position 10
N0190 X-12.5     Feed to position 11
N0200 G00 Z2.5 M09     Clear part
N0210 G80 X-50.0 Y50.0 Z63.0 T02 M06     Tool change "6.0 dia. drill"
N0220 X13.0 Y-13.0 Z2.5 S1600 M03     Rapid to position 12, turn spindle on
N0230 G81 Z-29.3 R2.5 F244 M08     Drill position 12, turn coolant on
N0240 Y-63.5     Drill position 13
N0250 X152.0 Y-19.0     Drill position 14, turn coolant off
N0260 G80 X-50.0 Y50.0 Z89.0 T03 M06     Tool change "24.0 dia drill"
N0270 X90.0 Y-57.5 Z2.5 S425 M03     Rapid to position 15, turn spindle on
N0280 G81 Z-34.7 R2.5 F216 M08     Drill position 15, turn coolant on
N0290 M09     Turn coolant off
N0300 G80 X-50.0 Y50.0 Z25.0 T04 M06     Tool change, "25.0 dia boring bar"
N0310 X90.0 Y-57.5 Z2.5 S1200 M03     Rapid to position 16, turn spindle on
N0320 G81 Z-27.5 R2.5 F183 M08     Bore position 16, turn coolant on
N0330 M09     Turn coolant off
N0340 G80 X-50.0 Y50.0 Z51.0 M02     Rapid to tool change position and end
                                         program
```

CONVERSATIONAL PART PROGRAMMING

Conversational part programming requires the programmer to respond to a set of questions that is a built-in feature of the machine control system and displayed on the CRT screen of the control unit. As each response is made, further options are presented and responses made until that particular group, or "block" of related data, is complete. The programmer then moves on to the next block.

For example, assume that, as part of a milling program, a relative tool movement of −39.786 mm (−1.568 in.) is required in a certain direction identified as the *X* axis. An appropriate feedrate has already been programmed.

With conversational programming the programmer will establish the appropriate operating mode, that is, linear motion at a controlled feedrate, by making a selection from a list of alternatives pushing the appropriate button. That selection having been made, the next prompt will ask for the dimensional value of the intended move in the *X* axis to be keyed in by the operator.

If listed an example of the data for the above move could read as follows: Note different controls will use varying formats.

```
N260 MILL or LIN X-39.786
```

where the entry MILL or LIN (linear) indicates the type of slide movement required and N260 is the data block number.

Similarly, consider a program entry to achieve combined slide movements that will result in a cutter path passing through an arc of 90° and having a radius 8 mm (0.3 in.).

First, the appropriate operating mode will be selected, such as CIRC—an abbreviation for circular interpolation. The prompts that follow will ask for the target position in the respective axes followed by the value of the radius and the direction of movement, whether clockwise or counterclockwise.

A listed program block containing these data would read

```
N350 CIRC X43.765 Z-75.000 R8 CW
```

The conversational concept can be extended to include machining requirements other than slide movement control.

Consider the turning of a bar of metal on a turning center. Before any thought can be given to slide movements, the basic metal-cutting data would have to be ascertained. For example, the correct spindle speed and feedrate are of vital importance. The spindle speed is affected by the work diameter and the cutting speed. The cutting speed is related to the material being machined. The feed rate would depend on the depth of cut, tool type, and surface finish required.

From this it can be seen that the necessary data to machine the metal successfully can be related to four factors:

(a) the material being cut;
(b) the material diameter;
(c) the surface finish required;
(d) the tool type.

Some computerized controls can be programmed to select the appropriate spindle speed and feedrate from an input of information relating to these factors. To assist the input of information there will be a material file and surface roughness file within the computer memory, as shown in Figure 8.66. Cutting tool types will also be numerically identified. This type of input could be found on a machine control unit, but is more likely to be encountered on offline programming systems.

MATERIAL STOCK FILE	
CODE	MATERIAL
1	MILD STEEL
2	MED. CARBON STEEL
3	STAINLESS STEEL
4	CAST IRON
5	ALUMINUM

SURFACE ROUGHNESS FILE	
CODE	Ra (μin.)
1	100 (250)
2	50 (125)
3	25 (63)
4	12.5 (32)
5	6.3 (16)

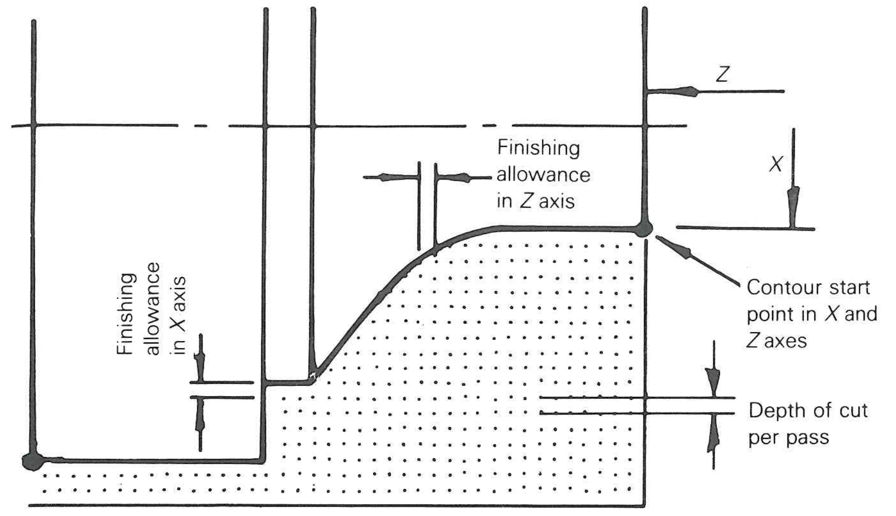
Figure 8.66 *Material file and surface roughness file.*

A simple question-and-answer routine will extract from the computer memory all the necessary data to give the correct cutting conditions, making calculations or judgments on the part of the programmer quite unnecessary. An example of a question-and-answer routine is as follows:

Prompt	Response
Material?	5
Material diameter?	50
Surface code?	4
Tool number?	8

There is no standard conversational part programming language. The systems are very individualistic. It should also be noted that while conversational MDI programs can be prepared away from the machine and then instantly entered into the control, usually by the use of magnetic tape, floppy disk, or direct numerical control, they are commonly entered by the machine setup person/operator pressing appropriate buttons as described previously. Unless the machine control unit is of the less common type that permits a second program to be entered while the first is being activated, shop floor data entry involves the machine being nonproductive while the program is being entered, and this does have its drawbacks. On the other hand, the technique is favored by some companies since total control of the machining operation by workshop personnel (as opposed to programs being prepared away from the shop floor and then subsequently passed on to them), makes use of valuable practical skills and, equally important, has the effect of improving job satisfaction.

Conversational controls are also unique in their programming in that they have many specialized routines in order to make part material removal and repetitive operations easier. Figure 8.67 shows a very useful stock removal


Z
X
Finishing allowance in Z axis
Finishing allowance in X axis
Contour start point in X and Z axes
Depth of cut per pass

Further data required:

(1) Profile data (added at end of program);

(2) Profile data start block number;

(3) Profile data end block number.

A final profile trace is optional.

Figure 8.67 *Stock removal to a defined contour cycle. (Refer to your programming manual for specific examples.)*

cycle. From one data block, plus blocks defining the component profile, the controller will automatically determine the number of passes necessary to remove the excess material, constantly varying the length of travel in the *Z* axis if need be, and finally taking a finishing cut to reduce the component profile precisely to size. The definition of the component profile in program data is added at the end of the program and is automatically activated via the stock removal cycle call. The profile definition can be achieved by the inclusion of appropriate minor cycles such as those described above.

This stock removal cycle is for application along the *Z* axis. Similar cycles cater for stock removal along the face or *X* axis of a workpiece.

Figures 8.68 and 8.69 illustrate cycles that may be used for reducing the diameter at any position along the length of a part and for grooving. Figure 8.70 illustrates a screw-cutting cycle, an essential feature of any control system devoted to turning. This particular version of a screw-cutting cycle is particularly easy to use. From just one block of data the control automatically de-

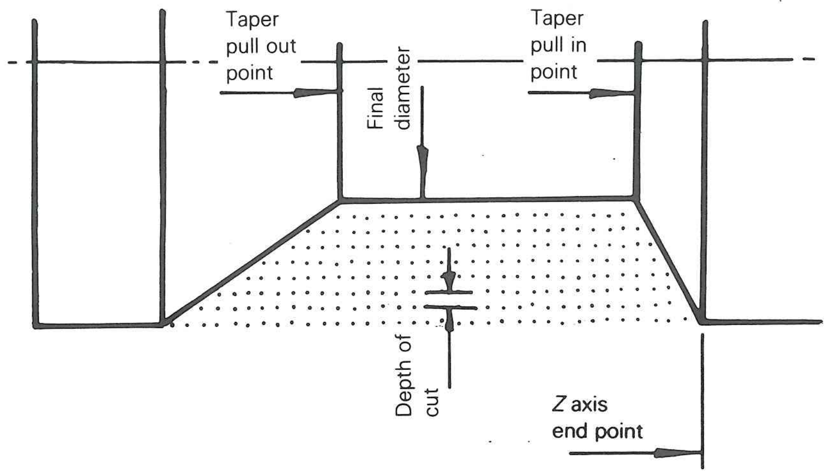


Figure 8.68 *Stock removal cycle with optional tapered entry and exit.*

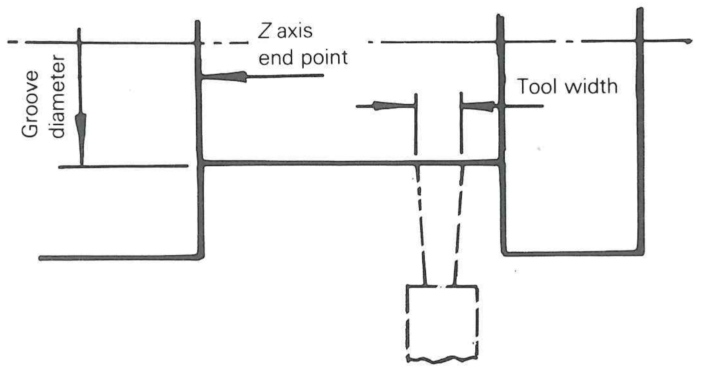
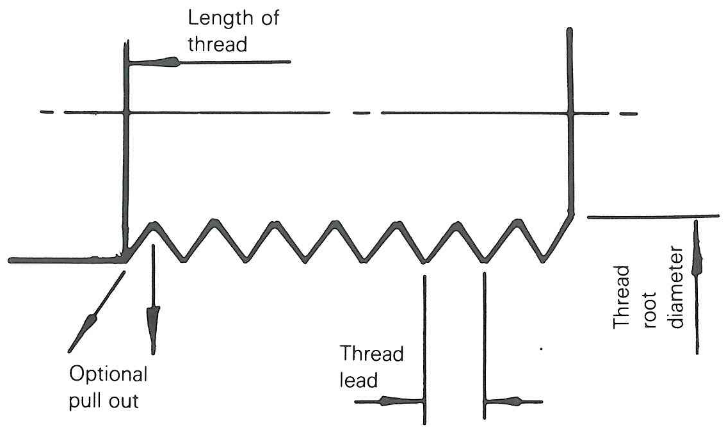


Figure 8.69 *Stock removal or grooving cycle.*

termines the number of passes necessary to achieve the required thread depth. In the block of data shown G84 is the cycle code, X specifies the thread root diameter, Z specifies the thread length, the P_1 value is the depth of the first pass, and P_2 is the lead. The control automatically effects a progressive reduction in the depth of cut of each pass that results in improved surface finish and prolonged tool life.

On less sophisticated screw-cutting cycles the X diameter for each pass along the thread length has to be predetermined by the part programmer, and each pass is programmed in a separate data block.

The range of special cycles used in milling machine control systems is equally helpful to the part programmer.



Data example: G84X8.168Z-20P_1 1P_2 1.5

Figure 8.70 *Automatic threading cycle (millimeter example).*

Fairly common is the provision of a face milling cycle such as that illustrated in Figure 8.71, where a data input specifies the dimensions of the face to be milled. From this information the control unit will determine the number of passes required while taking into consideration a stated cutter overlap that will ensure the face is evenly machined.

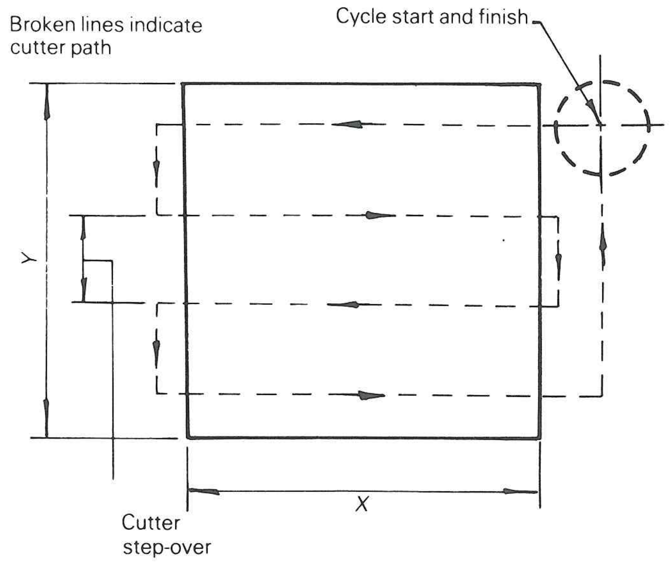


Figure 8.71 *Face milling cycle.*

Figure 8.72 illustrates a slot milling cycle. Here again the overall dimensions are programmed. The first pass made by the cutting tool goes through the center and then returns to the start. Further passes are made until the correct depth is achieved, the number of passes necessary being determined from the programmed movement to be made in the Z axis before each cut commences. When the correct depth is reached, the cutter path will be a series of loops increasing in size with each pass. As with the face milling cycle, the control unit will determine the number of loops necessary to machine the slot to size, again taking into consideration the need for each cutter pass to overlap to provide a completely clean surface. Another variation of this cycle will complete the looping sequence before dropping to the next Z depth.

Similar to the slot milling cycle is the pocket milling cycle. This cycle commences at the center of the pocket, the cutter feeding in the Z axis to a programmed depth. There follows a series of loops until the programmed X and Y dimensions are reached, again with a cutter overlap on each pass. If the pocket depth is such that more than one increment in the Z axis is necessary, the cutter is returned to the center of the pocket, Z increments down, and the cycle is repeated. Some systems provide for a cycle that roughs out the main pocket and then machines it to size with a small finishing cut. A pocket milling cycle is illustrated in Figure 8.73.

Figure 8.74 shows another widely used cycle referred to as a "bolt hole circle." This is for drilling a series of equally spaced holes on a pitch circle diameter. Given that the cutter has been brought to the pole position indicated, the other dimensional data required are the position of the first hole, the Z axis movement, the pitch diameter or radius depending on the control system, and the number of holes required. The control will make all the necessary calculations to convert the polar coordinates to linear coordinates and will effect slide movements accordingly.

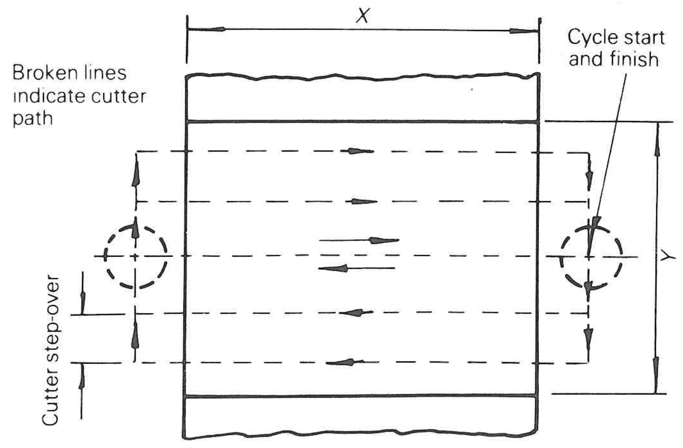


Figure 8.72 *Slotting cycle.*

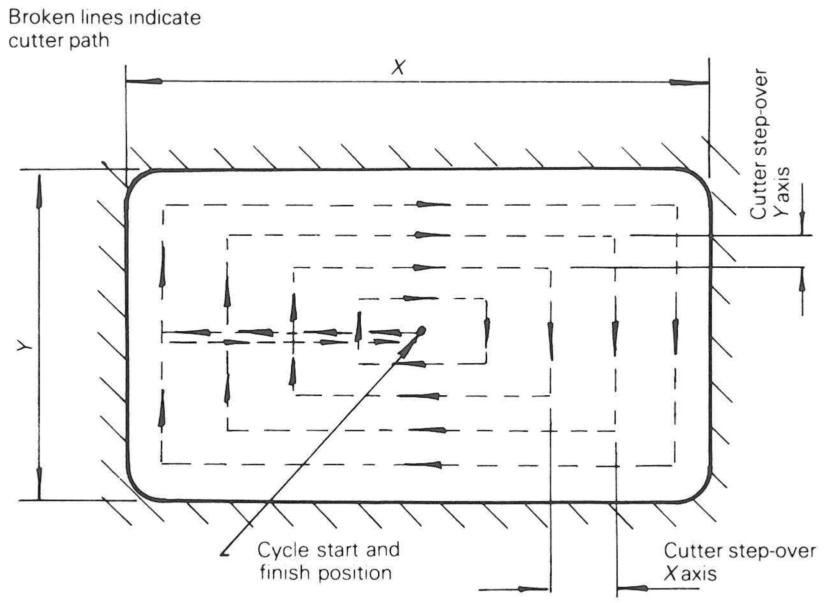

Broken lines indicate cutter path
X
Y
Cutter step-over Y axis
Cycle start and finish position
Cutter step-over X axis

Figure 8.73 *Pocket milling cycle.*

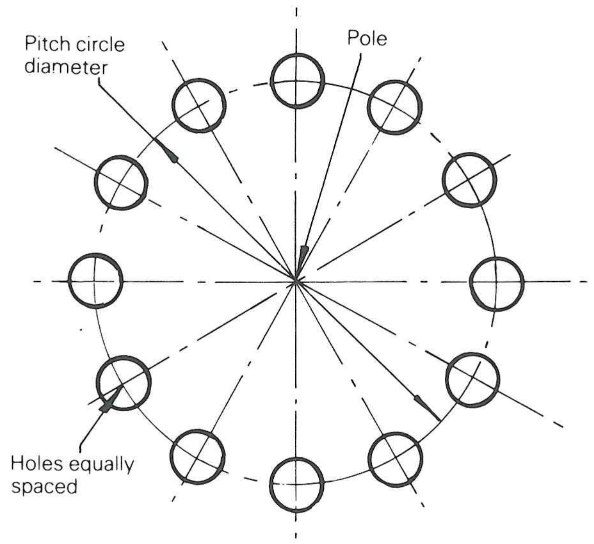


Figure 8.74 *Bolt hole circle.*

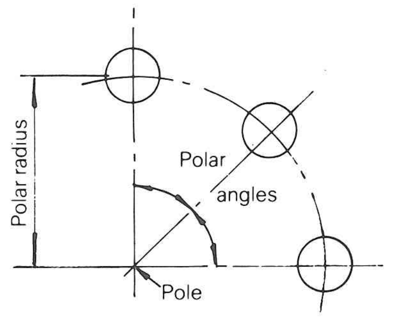


Figure 8.75 *Positioning using polar coordinates.*

A variation of this cycle will cater for just two or three holes positioned in an angular relationship to each other. An example is detailed in Figure 8.75. Again, the pole position is programmed and the cutter will be in this position when the cycle commences. The additional data that will be required are the *Z* axis movement, the polar radius and the polar angle(s); the controller will convert this information to slide movement in the relative axes.

Further milling cycles include those for boring, threading, elliptical profiles and even for the machining of helical arcs which simply requires data defining an arc in the *X* and *Y* axes and the change in the *Z* axis dimension between the start and end point. A moment's thought about the mathematical complexity of programming a cutter path such as this should be more than sufficient to emphasize how valuable canned cycles are as an aid to simplifying part programming procedures.

It should be noted that specialized cycles are normally meant for uniform or symmetrical shapes and as programming becomes more complex it is usually necessary to return to "G" code programming. Many conversational controls now have the ability to be programmed in two of three dimensional mode and conversational or "G" code mode.

ROTATION AND TRANSLATION

The positions of holes in angular relationship to each other was discussed previously on bolt circles. The machined feature—the hole—is rotated about a polar position.

The ability to rotate the positions of holes in this way is generally associated with the bolt hole circle facility and is commonly available. Many control systems also have the ability to rotate more complex features. The principle of rotation is illustrated in Figure 8.76.

A programming facility closely allied to rotation is translation. This permits the programmer to reposition a feature about a defined pole or center and then

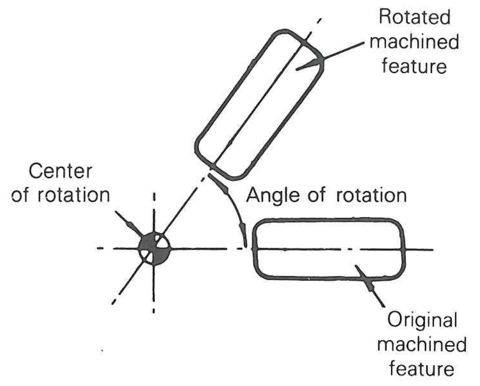


Figure 8.76 *Rotation of machined feature.*

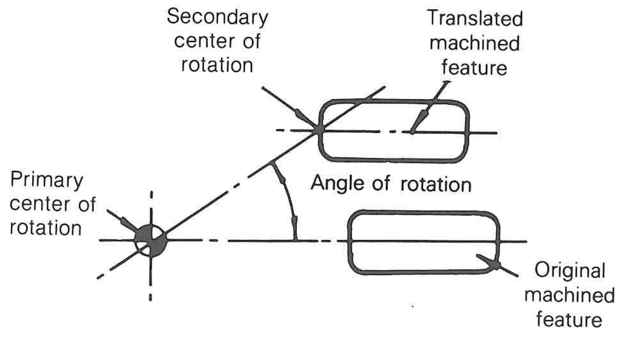

Secondary center of rotation
Translated machined feature
Primary center of rotation
Angle of rotation
Original machined feature

Figure 8.77 *Polar translation.*

to rotate the feature about a predetermined point on the feature itself. The principle is illustrated in Figure 8.77.

A bonus of the translation facility is that features of complex shape that are required in an angular location, which thus present some fairly complex programming calculations, can be programmed as though they lay on a true *XY* axis, simplifying the calculations required. They can then be repositioned using translation.

Translation may also be defined in linear terms and is, in effect, a datum shift facility. The shift may be in the *X* or *Y* axis or in both. Translation defined in this manner is illustrated in Figure 8.78.

SCALING

The scaling facility available on some control systems enables components that are geometrically identical but uniformly variable dimensionally to be produced from the same program data.

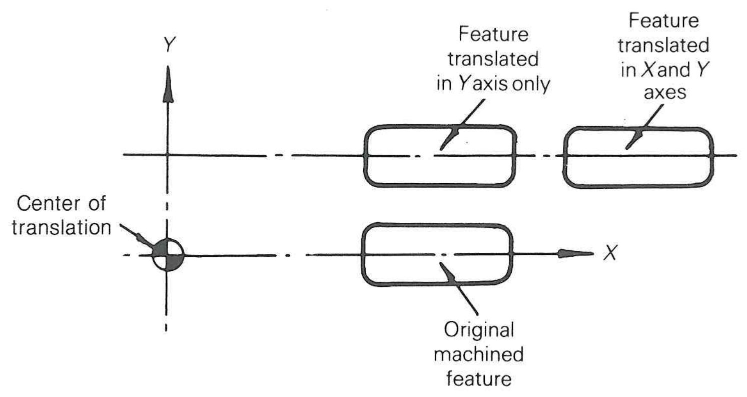

Y
Feature translated in Y axis only
Feature translated in X and Y axes
Center of translation
X
Original machined feature

Figure 8.78 *Linear translation.*

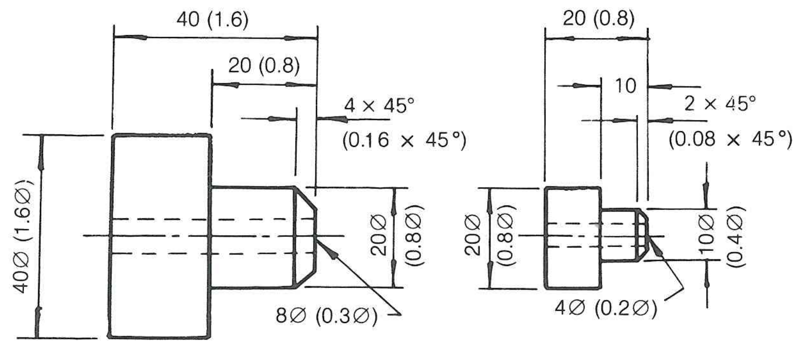


Figure 8.79 *Geometrically identical components suitable for production by scaling. (Inch units are given in parentheses.)*

Figure 8.79 illustrates two components, the production of which could be accommodated by scaling. Scaling is also available for milling operations. It can be applied to complete components or to one feature of a component.

An example of a scaling factor range, available on a widely used vertical machining center, is from 0.002 to 250. With such an extensive range the desired reduction or increase in size could well involve a machining requirement outside the capabilities of the machine, in which case an error message would be indicated. In practice components likely to be considered for production by scaling would rarely involve the use of widely varying scaling factors, but even a small scale factor, say 2, could produce an unacceptable result if the original data was for a fairly large component.

Some feel it is proper to use the lower end of a scaling factor range to minutely increase or decrease the machined size in one or more axes to maintain a dimensional tolerance that may be being lost owing, for example, to the effect of clamping or distortion of the workpiece. This is not normally done, though, and is usually the task of the tool compensation option to be discussed subsequently.

MIRROR IMAGE

Mirror image is the term used to describe a programming facility used to machine components, or features of components, that are dimensionally identical but geometrically opposite either in two axes or one axis. By using the mirror image facility, such components can be machined from just one set of data.

In Figure 8.80 an original component feature is indicated in the bottom left-hand corner of the diagram. A complete mirror image of that feature is shown in the top right-hand corner, while mirror images in the *X* axis and the *Y* axis are shown, respectively, in the bottom right- and top left-hand corners of the diagram.

To produce a complete mirror image both the *X* axis and *Y* axis dimensional values will change from negative to positive. For half mirror images the dimensional values will change from negative to positive in one axis only.

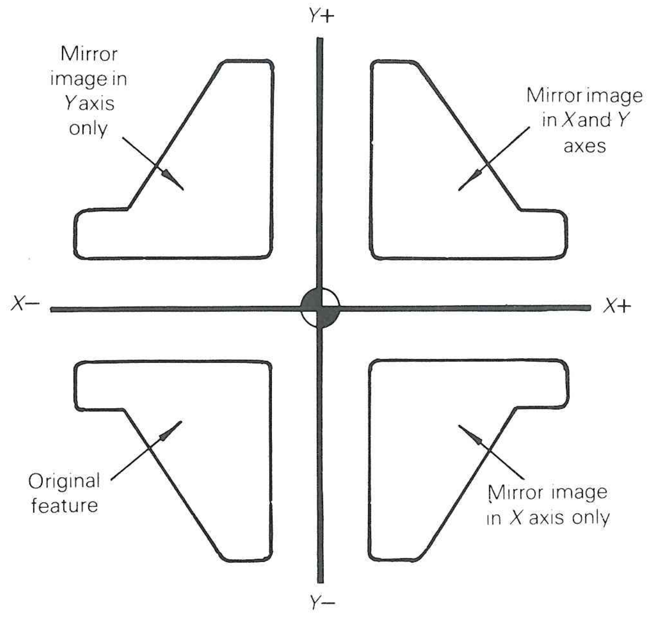

Y+
Mirror image in Y axis only
Mirror image in X and Y axes
X−
X+
Original feature
Mirror image in X axis only
Y−

Figure 8.80 *Mirror image.*

CONVERSATIONAL PART PROGRAMMING EXAMPLES

The programming examples that follow were prepared to show the vast differences in systems as well as their similarities. Both examples are from vertical CNC milling machines. The positional calculations are not shown because they are computed in the same manner as in the manual part program examples given earlier in the chapter.

The first example (Figure 8.83) is that of a Wells Index 320 mill (Figure) 8.81) with a Heidenhain TNC145 control (Figure 8.82). The example indicates some screen commands as well as operator responses and then a print out of the program. This example is intended to give a feel of what it is like to program this type of conversational control. (These examples are given in inch units only.)

Example 1

Operation description:

Using 0.500 in. diameter endmill cut a 0.100 in. deep step around the part maintaining a 0.250 in. width (Figures 8.83 and 8.84).

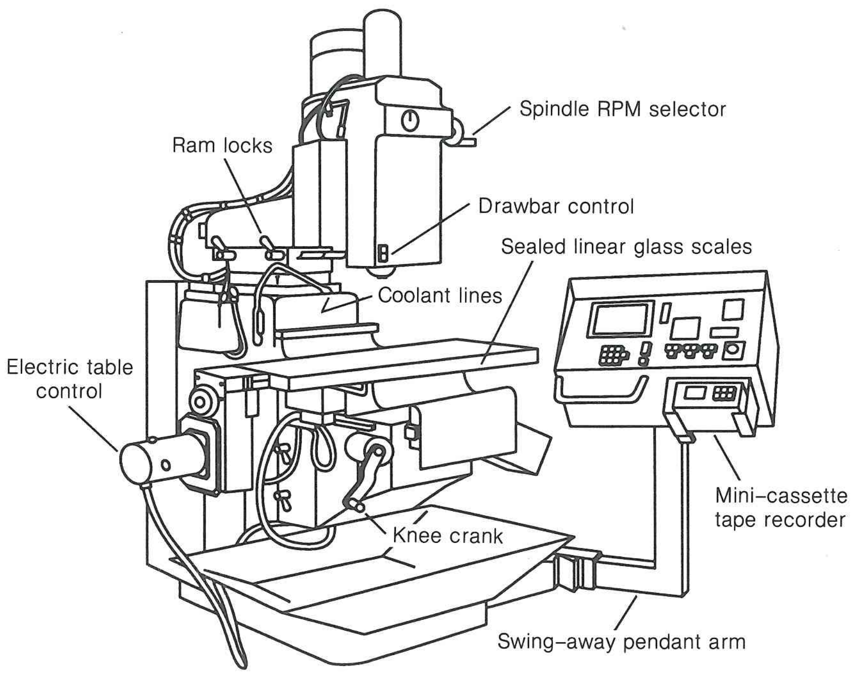


Figure 8.81 *Wells-Index System 3 CNC mill with Heidenhain 145C control.*

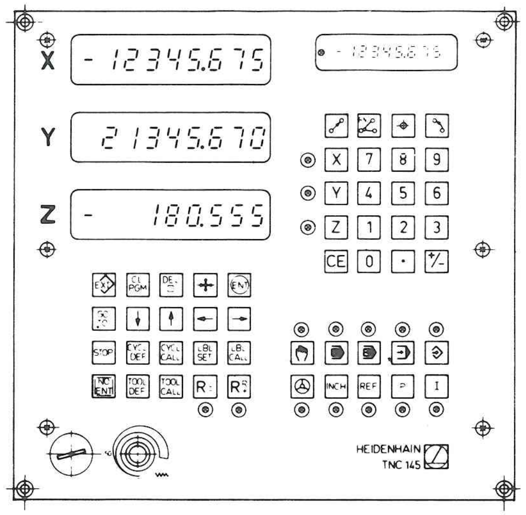


Figure 8.82 *TNC 145C console keyboard.*

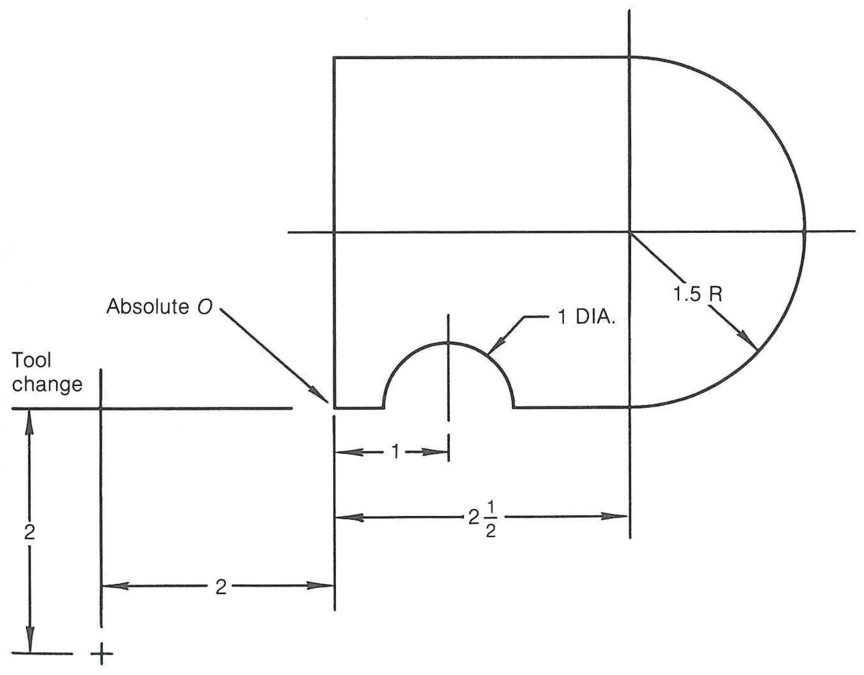


Figure 8.83 *Wells example.*