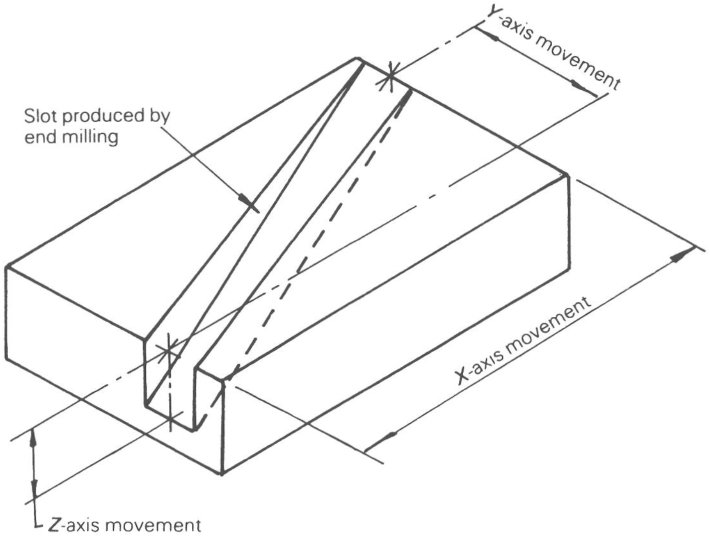


Figure 6.3 *Resulting tool path when three slides move simultaneously.*

Contouring

Contouring also involves two or more controlled slide movements resulting from program data that specify the next position required and the required feed rates to reach that position, so there is some overlap between linear interpolation and contouring. However, contouring can also be much more complex, involving combinations of angular movement and curves with one feature moving without interruption in the cutting process into another. This type of movement gives rise to the expression continuous path machining, which is often used to describe contouring.

Machining of the elliptical profile shown in Figure 6.4 would involve continuous path movement. Likewise, the radii shown on the components in Figure 6.5 would be produced in a similar manner. The elliptical shape is not readily defined in numerical terms, and to produce the necessary cutter path would present an interesting, although not insurmountable, problem to the part programmer unless the control system was specially equipped with a canned cycle to deal with such a situation. On the other hand, the two radii shown in Figure 6.5 are an everyday occurrence and most control systems can readily accommodate the production of a radius, or a combination of radii. Such a facility is referred to as circular interpolation.

Circular arcs may be programmed in the *XY*, *XZ*, and *YZ* planes. In exceptional cases three axes may be involved, resulting, in effect, in a helical tool path.

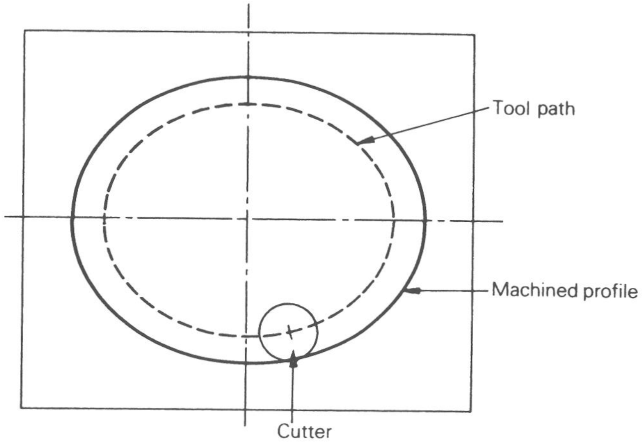


Figure 6.4 *Component profile produced by contouring.*

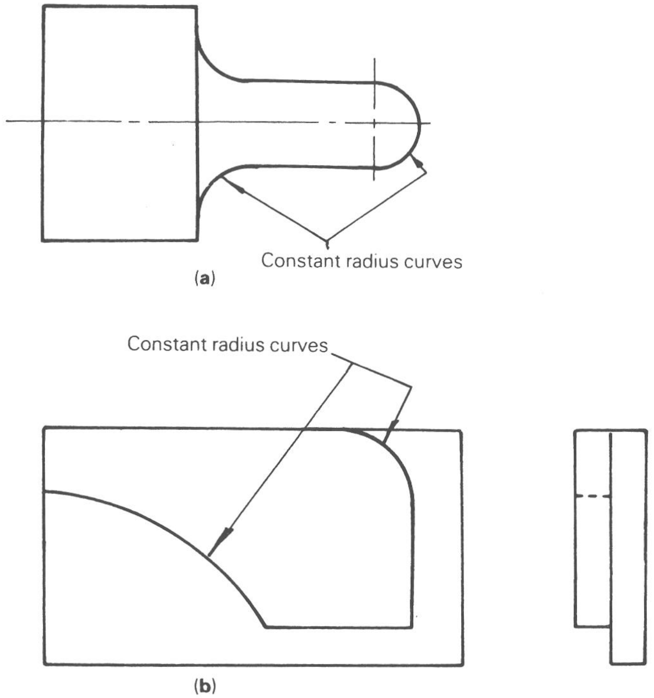


Figure 6.5 *Components with radial features requiring circular interpolation:* (a) *turned component and* (b) *milled profile.*

PROGRAMMING POSITIONAL MOVES

In practice the three types of positioning referred to previously are rarely isolated. The production of the majority of components will involve a combination of the techniques. However, it will be necessary to clearly identify in the part program the type of positioning required at each stage of the machining process.

Manual data input systems will vary from one control system to another. For example, a widely used training machine specifies all linear movement as linear interpolation and differentiates by linking the movement to an appropriate feed rate. The program entry is reduced to pressing a linear interpolation key followed by the dimensional detail and the feed rate. Similarly, a radius is simply defined by pressing a circular interpolation key, followed by a data entry of the dimensional value of the target position, the radius, and the direction of rotation as either clockwise or counterclockwise.

Control systems using the recommendations contained in EIA RS-274-D or BS 3635:1972 will specify the type of positioning involved by using the appropriate preparatory function or G code, the common ones being as follows:

G00	Point-to-point
G01	Linear interpolation
G02	Circular interpolation clockwise
G03	Circular interpolation counterclockwise

Having defined the type of positioning in this way the instruction is completed by including dimensional details of the move together with the feed rate for G01, G02, and G03. G00 moves are usually made at the maximum slide traverse rate for the machine.

DIMENSIONAL DEFINITIONS OF SLIDE MOVEMENT

In Chapter 1 it was explained that the axes in which slide movement can take place are designated by a letter and either a plus (+) or minus (−) sign to indicate the direction of movement. Unfortunately, these designated slide movements, owing to the different design configurations of machine tools, do not always coincide with the movement of the tool in relation to the work, and as a result this can cause some confusion when slide movements are being determined. In the case of a turning center with a conventional tool post there is no problem, since the slide movement and the tool movement in relation to the work are identical. But on a vertical machining center, for example, to achieve a positive (+) movement of the tool in relation to the work, the table, not the cutter, has to move, and this movement is in the opposite direction. Since a move in the wrong direction, especially at a rapid feed rate, could have disastrous results, this fact should be clearly understood.

A sound technique when determining slide movements is to program the tool movement in relation to the work. In other words, on all types of machines, imagine it is the tool moving and not, as is sometimes the case, the workpiece. To do this it is necessary to redefine some, but not all, of the machine movements. A simple diagram such as the one alongside the components shown in Figures 6.6 and 6.9 is usually very helpful.

Once the direction of movement has been established it will need to be dimensionally defined. There are two methods used, and they are referred to as:

(a) absolute;
(b) incremental.

Figure 6.6 shows the profile of a component to be machined on a turning center using the machine spindle center line and the face of the workpiece as datums in the *X* and *Z* axes respectively. Assume the sequence of machining is to commence with the 1.4 in. (35 mm) diameter, followed by the 1.2 in. (30 mm) diameter and finishing with the 1 in. (25 mm) diameter.

To machine the profile using absolute dimensions, it is necessary to relate all the slide movements to a preestablished datum. The movements required in absolute terms are indicated in Figure 6.7. Note that all position commands are the actual distance that the tool tip is from the datum point.

Incremental positioning involves relating the slide movement to the final position of the previous move. The slide movements, expressed in incremental terms, which would be necessary to machine the profile are indicated in Figure 6.8. Note position commands indicate the direction and the exact amount of slide motion required.

Note that each dimension in the *X* axis in Figure 6.7 is equal to the work

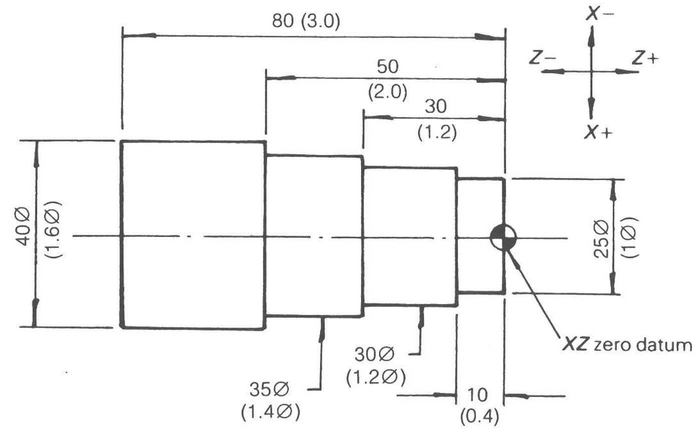
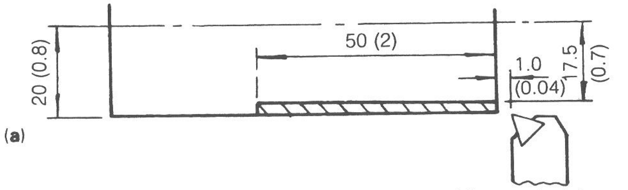


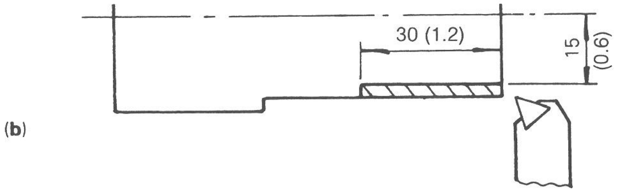
Figure 6.6 *Component detail. (Inch units are given in parentheses.)*



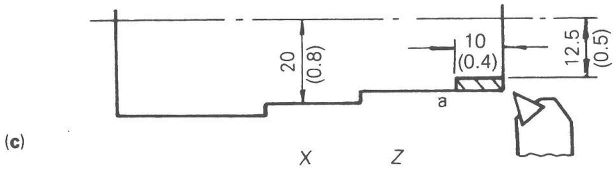
All cuts start 1 mm (0.04 in.) from face

Axis movements

	X	*Z*
Feed in	17.5 (0.7)	1.0 (0.04)
Turn 35∅ (1.4∅)	17.5 (0.7)	−50.0 (−2)
Withdraw	18.0 (0.72)	−50.0 (−2)
Return	18.0 (0.72)	1.0 (0.04)



	X	*Z*
Feed in	15.0 (0.6)	1.0 (0.04)
Turn 30∅ (1.2∅)	15.0 (0.6)	−30.0 (−1.2)
Withdraw	15.5 (0.62)	−30.0 (−1.2)
Return	15.5 (0.62)	1.0 (0.04)



	X	*Z*
Feed in	12.5 (0.5)	1.0 (0.04)
Turn 25∅ (1∅)	12.5 (0.5)	−10.0 (−0.4)
Withdraw	20.0 (0.8)	−10.0 (−0.4)
Return	20.0 (0.8)	1.0 (0.04)

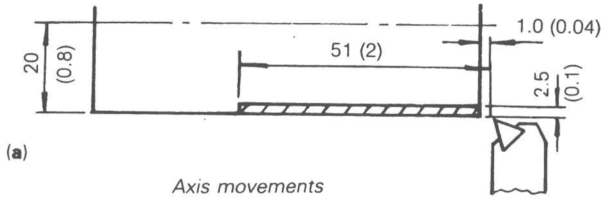
Figure 6.7 *Turning using absolute positioning. (Inch units are given in parentheses.)*

radius. When turning, some control systems will require dimensions in the *X* axis to be stated as a diameter, other machines may allow the programmer to select radius or diameter programming.

Figure 6.9 shows a component that is to be milled in the sequence A to C on a vertical machining center using datums as indicated. Assume that the movement in the *Z* axis to give a slot depth of 0.4 in (10 mm) has already

been made. The necessary slide movements in the *X* and *Y* axes in absolute and incremental terms are indicated in Figures 6.10 and 6.11, respectively.

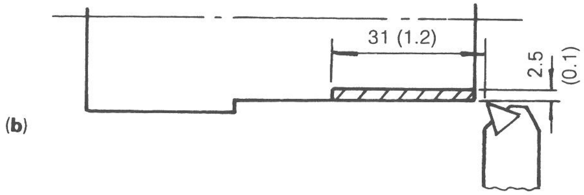
On the more sophisticated control systems, it is possible to use absolute and incremental dimensional definition within the same program, the distinction being achieved by using the G91 preparatory function code when the switch from absolute (G90) to incremental (G91) is to be made.



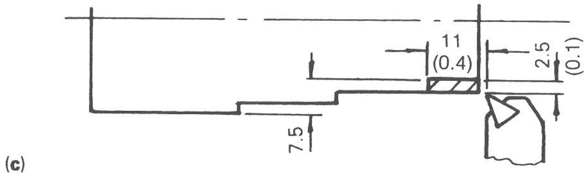
Axis movements

All cuts start 1 mm (0.04 in.) from face

	X	*Z*
Feed in	−2.5 (−0.1)	0
Turn 35∅ (1.4∅)	0	−51.0 (−2)
Withdraw	0.5 (0.02)	0
Return	0	51.0 (2)



	X	*Z*
Feed in	−3.0 (−0.12)	0
Turn 30∅ (1.2∅)	0	−31.0 (−1.2)
Withdraw	−0.5 (0.02)	0
Return	0	31.0 (−1.2)



	X	*Z*
Feed in	−3.0 (−0.12)	0
Turn 25∅ (1∅)	0	−11.0 (−0.4)
Withdraw	7.5 (0.3)	0
Return	0	11.0 (0.4)

Figure 6.8 *Turning using incremental positioning. (Inch units are given in parentheses.)*

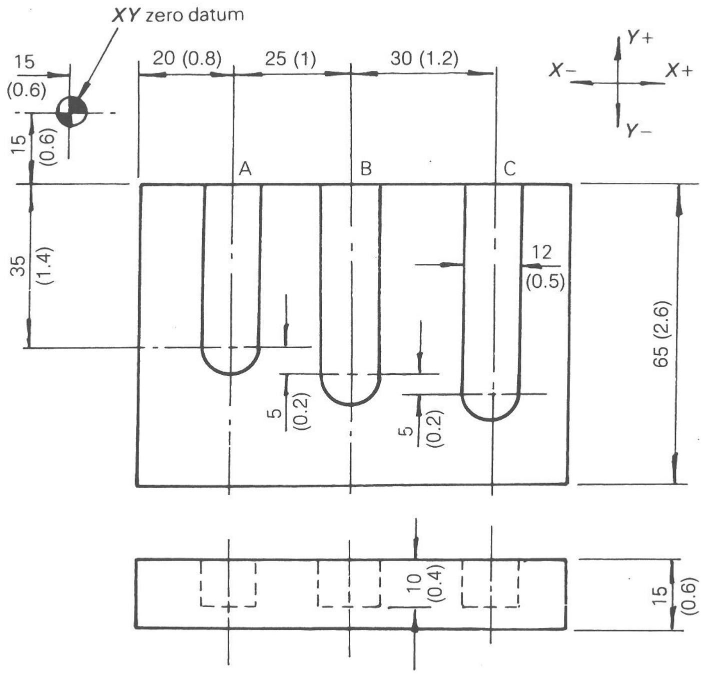


Figure 6.9 *Component detail. (Inch units given in parentheses.)*

CIRCULAR INTERPOLATION

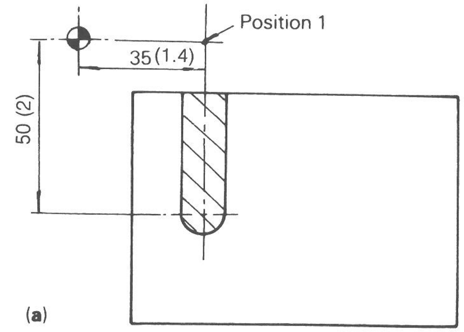
It was stated earlier that circular arc programming, particularly on conversational data input systems, has been reduced to simply dimensionally defining the target position, the radius, and the direction in which movement is to take place. On control systems using the word address format, it is rather more complex and there are slight variations in approach. Two of these variations will be considered later.

Common to all systems used to program circular movement is the need to determine whether the relative tool travel is in a clockwise (CW) or counterclockwise (CCW) direction. The following approach is usually helpful.

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

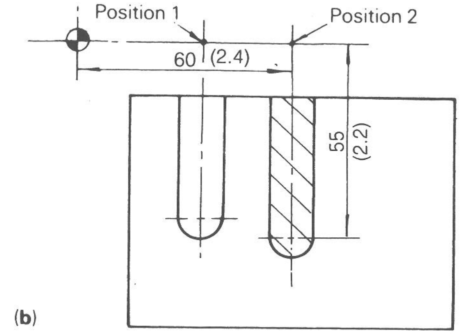
The standard G codes for circular interpolation are G02 (CW) and G03 (CCW). However, not all systems adopt this recommendation and there is at least one widely used system in which they are reversed, that is, G02 is CCW and G03

Figure 6.10 *Milling using absolute positioning. (Inch units are given in parentheses.)*

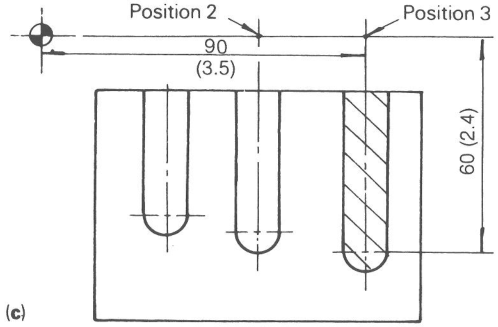


Axis movements

	X	*Y*
Move from datum to position 1	35.00 (1.4)	0
Mill to length	35.00 (1.4)	−50.00 (−2.00)
Return to position 1	35.00 (1.4)	0

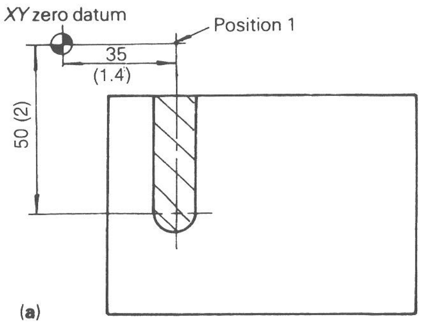


	X	*Y*
Move from position 1 to position 2	60.00 (2.4)	0
Mill to length	60.00 (2.4)	−55.00 (−2.2)
Return to position 2	60.00 (2.4)	0



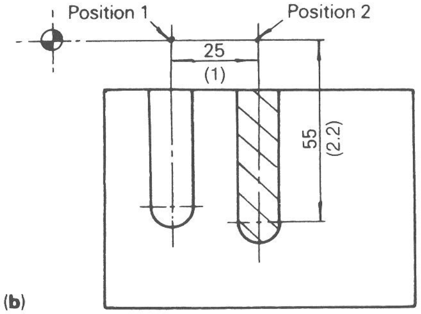
	X	*Y*
Move from position 2 to position 3	90.00 (3.5)	0
Mill to length	90.00 (3.5)	−60.00 (−2.4)
Return to position 3	90.00 (3.5)	0
Return to datum	0	0

Figure 6.11 *Milling using incremental positioning. (Inch units are given in parentheses.)*

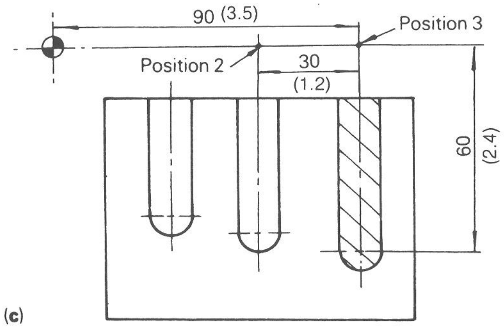


Axis movements

	X	Y
Move from datum to position 1	35.00 (1.4)	0
Mill to length	0	−50.00 (−2.00)
Return to position 1	0	50.00 (2.00)



	X	Y
Move from position 1 to position 2	25.00 (1.00)	0
Mill to length	0	−55.00 (−2.2)
Return to position 2	0	55.00 (2.2)



	X	Y
Move from position 2 to position 3	30.00 (1.2)	0
Mill to length	0	−60.00 (−2.4)
Return to position 3	0	60.00 (2.4)
Return to datum	−90.00 (−3.5)	0

is CW. (In this case it is advisable to refer to the machine tool programming manual.)

The three variations in arc programming referred to above are as follows. Note: That machines will normally not have all three methods of circular arc programming.

Method 1

Assuming that the last programmed move brought the cutting tool to the start point, the arc is defined in the following manner:

1. The finish or target point of the arc is dimensionally defined in relation to the start point using the appropriate combination of X, Y, and Z dimensional values stated in absolute or incremental terms.
2. The center of the arc is dimensionally defined in relation to the start point using I, J, and K values measured along the corresponding X, Y, and Z axes respectively.

Thus the arc shown in Figure 6.12 would be programmed as follows. In absolute terms using diameter programming:

		X	*Z*	*I*	*K*
Inch	G02	1.6	2.0	0	0.8
Metric	G02	40	50	0	20

In incremental terms:

		X	*Z*	*I*	*K*
Inch	G02	0.8	−0.8	0	0.8
Metric	G02	20	−20	0	20

The variation in the X values in these two examples is because the absolute program assumes that X values are programmed as a diameter rather than a radius.

I has no value because the center and start point of the arc are in line with each other in relationship to the X axis. In practice, when a value is zero, it is not entered in the program.

The I, J, and K values are always positive, with I related to X, J related to Y, and K related to Z.

Complete circles and semicircles are programmed as a series of 90° quadrants in many cases. Thus a complete circle would require four lines of program entry. New pieces of equipment can now complete full circles in one line of program entry.

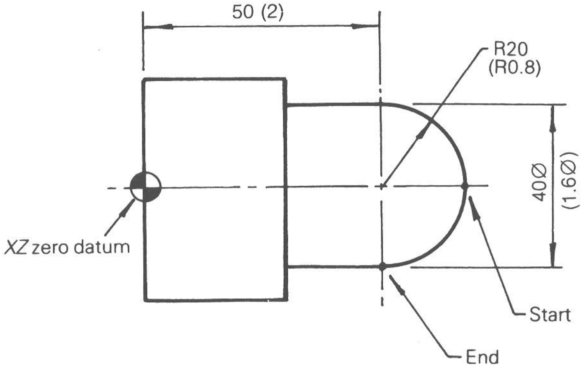


Figure 6.12 *Turned component detail involving arc programming. (Inch units are given in parentheses.)*

Figure 6.13 shows the program for a milled profile. The cutter radius has been ignored.

In absolute terms:

Inch	G	X	Y	I	J
	03	2	−1.2	1.2	0
	02	3.5	−1.2	0	1.6
Metric	G	X	Y	I	J
	03	50	−30	30	0
	02	90	−70	0	40

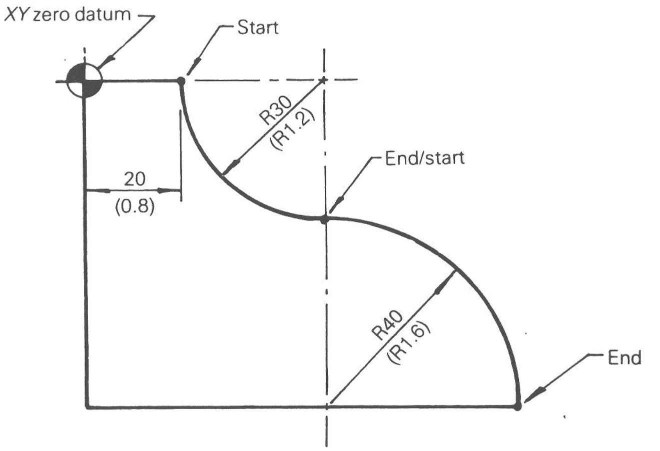


Figure 6.13 *Milled component involving arc programming. (Inch units are given in parentheses.)*

In incremental terms:

Inch	G	X	Y	I	J
	03	1.2	−1.2	1.2	0
	02	1.6	−1.6	0	1.6
Metric	G	X	Y	I	J
	03	30	−30	30	0
	02	40	−40	0	40

There are often situations where the start and/or stop points do not coincide with an X, Y, or Z axis, and it is then necessary to make a series of calculations. Such a situation is shown in Fig. 6.14. Dimensional values for X, Y, I, and J have to be determined. The necessary trigonometry is indicated in Fig. 6.15.

From A to B the magnitude of the X move is

Inch $1 \times \cos 30^\circ - 1 \times \cos 75^\circ = 0.866 - 0.259 = 0.607$
Metric $25.00 \cos 30^\circ - 25.00 \cos 75^\circ = 21.65 - 6.47 = 15.18$

From A to B the magnitude of the Y move is

Inch $1 \times \sin 75^\circ - 1 \times \sin 30^\circ = 0.966 - 0.500 = 0.466$
Metric $25.00 \sin 75^\circ - 25 \sin 30^\circ = 24.15 - 12.50 = 11.65$

The magnitude of the I dimension in the X axis is

Inch $1 \times \cos 75^\circ = 0.259$
Metric $25 \cos 75^\circ = 6.47$

The magnitude of J in the Y axis is

Inch $1 \times \sin 75^\circ = 0.966$
Metric $25.00 \sin 75^\circ = 24.15$

Once the dimensions have been incorporated, they are incorporated in the program as before.

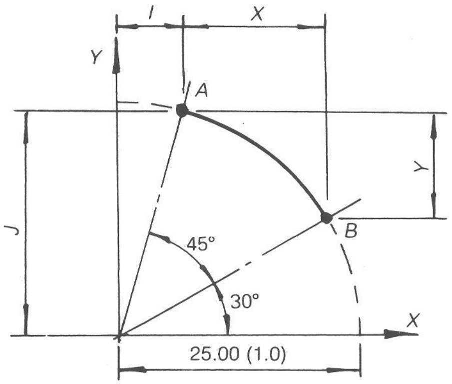


Figure 6.14 *Partial arc programming. (Inch units are given in parentheses.)*

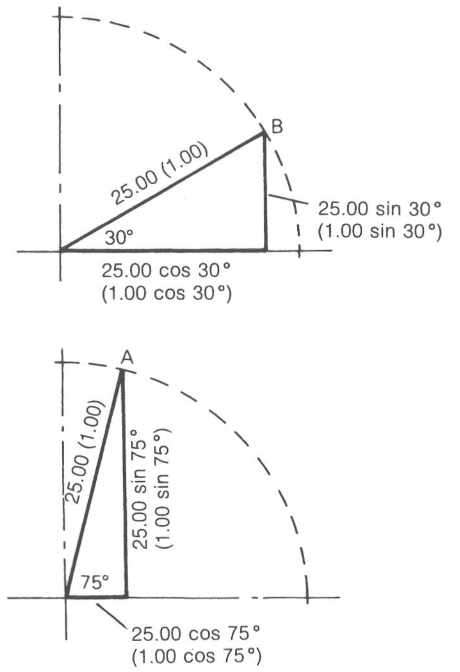


Figure 6.15 *Trigonometry required to program a partial arc. (Inch units are given in parentheses.)*

Method 2

The second method of arc programming varies from the one previously described in the way in which the arc center is defined. As in the previous method, it will be assumed that the cutting tool has arrived at the start point of the curve. To continue, the following data are required.

1. The finish or target point of the arc is dimensionally defined in relation to the start point using the appropriate combination of *X*, *Y*, and *Z* values stated in absolute or incremental terms.
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.

Using this method the arc shown in Fig. 6.12 would be programmed as follows.

In absolute terms:

	G	X	Z	I	K
Inch	02	1.6	2.0	0	2.0
Metric	02	40	50	0	50

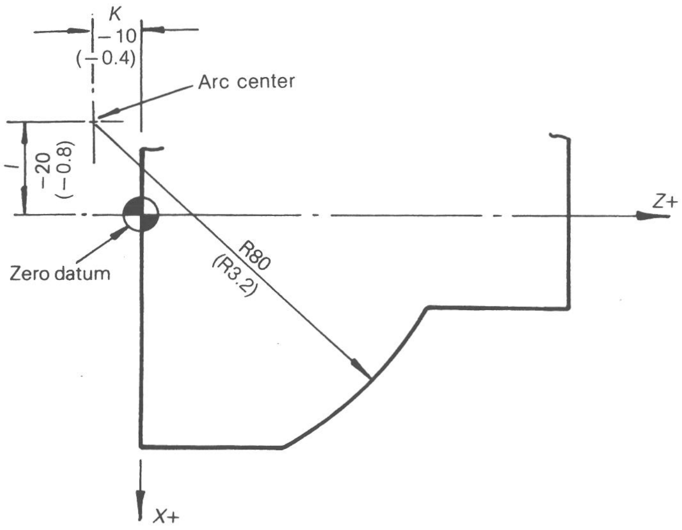


Figure 6.16 *Negative I and K values. (Inch units are given in parentheses.)*

In incremental terms:

	G	X	Z	I	K
Inch	02	0.8	−0.8	0	2.0
Metric	02	20	−20	0	50

Note that in this example it is *I* that has no value, since the center of the arc lies on the *X* datum and therefore *I* would be omitted from the program.

When the arc center is related to the program datum it is possible for the *I*, *J*, and *K* values to be a negative quantity, as illustrated in Figure 6.16.

The programming methods referred to above concern arcs of up to 90°. Some of the more modern control systems permit programming of arcs in excess of 90° in one data block, a facility referred to as 'multi-quadrant' programming or 360° circular interpolation.

Method 3

The third method of arc programming on some controls is to use absolute or incremental polar coordinates. It varies from the previous methods in that it does not use *I* and *J* values. With this method the circle center point has been defined previously with *X*, *Y*, or *Z* values. The arc is then programmed with a radius dimension and an angular amount of tool path from the circle center. A positive or negative angle will establish the direction of the cutter path:

1. The circle center is established with absolute or incremental dimensions.

2. The tool will have been moved to the arc start point.
3. The degrees of arc and radius are then programmed, with the sign (+ or −) on the degrees of arc establishing direction of cut.

Using the above terms the arc in Figure 6.12 would be programmed as follows:

In absolute terms:

Inch	CC *X*0 *Z*2	Define circle center
	G1 *X*0 *Z*2.8	Position cutter to starting point
	C Polar radius 0.8 polar angle 90°	Cut circle
Metric	CC *X*0 *Z*50	Define circle center
	G1 *X*0 *Z*70	Position cutter to start point
	C Polar radius 20 polar angle 90°	Cut circle

Positive angles denote clockwise motion; negative angles denote counterclockwise motion. Note: Most machines with polar coordinate circular interpolation capabilities are conversationally controlled.

In incremental terms:

Inch	CC *X*0 *Z*2	Circle center from datum
	G1 *X*0 *Z*2.8	Position tool to start from circle center
	C Polar radius 0.8 polar angle 90°	Circular movement
Metric	CC *X*0 *Z*50	Circle center from datum
	G1 *X*0 *Z*20	Position tool to start from circle center
	C Polar radius 20 polar angle 90°	Circular movement

RAMP

The starting and stopping of slide servo motors appear to be instantaneous. In fact there is, of course, a brief period of acceleration at the start of a move and a brief period of deceleration at the end of a move. This is shown graphically in Figure 6.17.

The period of acceleration is known as "ramp up" and the period of deceleration as "ramp down." The ramp is a carefully designed feature of the servo motor.

From a metal-cutting point of view, the quicker a slide attains its correct feed rate the better, and ideally this should be maintained throughout the cut. The ramp period therefore is kept as brief as possible, but consideration has to be given to ensuring that at the end of the movement there is no motor overrun or oscillation, both of which could affect the dimensional accuracy of the component.

For linear interpolation the ramp effect is rarely of concern, but for circular interpolation, and particularly where one curve runs into another, it is preferable that there is no speed variation of the servo motor, and thus of the feed rate of the slide, however small this might be. Any such variation would not only affect the metal-removal rate but may also affect the dimensional accuracy and surface finish of the component. Because of this, many control units are equipped with a *ramp inhibit* or *ramp suppression* facility, which means there is no slowing down or acceleration of the slide movement as one programmed movement leads into a second. G codes allocated to ramp are usually G08 and G09.

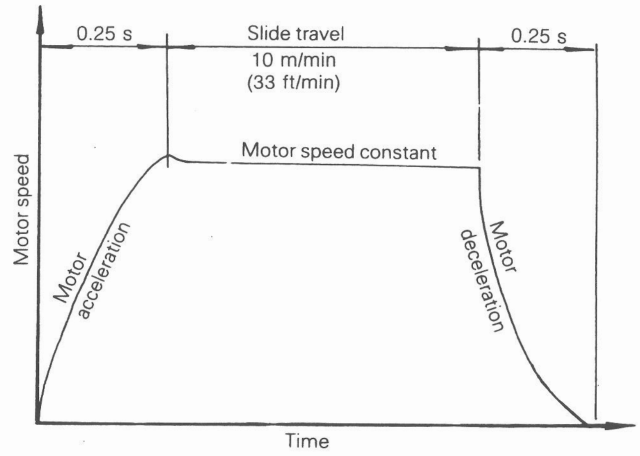


Figure 6.17 *Servo motor speed/feed rate relationship.*

REPETITIVE MACHINING SEQUENCES

There are a number of machining sequences that are commonly used when machining a variety of components. Other less common sequences may be repetitive, but only on one particular component. It is helpful, since it reduces the program length, if such a sequence can be programmed just once and given an identity so that it can be called back into the main program as and when required. Such sequences are referred to in a variety of ways, for example, as cycles, subroutines, loops, patterns, and macros. Although this can be slightly confusing, there are instances when one particular title appears to be more appropriate than the others. Various types of repeat machining sequences are discussed here.

Standardized Fixed Cycles

A number of the basic machining sequences, or cycles, commonly used were initially standardized (ANSI/EIA RS-274-D:1979; BS 3635:1972). The rec-

2. Th
3. Th

Usi
lows:
In a

Inch

Metric

Positi
wise
capab
In i

Inch

Metric

The s
fact
and a
icall
Th
lerati
moto
Fr
feed
The
to be
run
com
Fc

ommendations were commonly adopted and continue to be employed today. The machining cycles are identified by assigned G codes, and when they are incorporated into a control system, they are referred to as "fixed" or "canned" cycles. Perhaps the most commonly used fixed cycle is that of drilling a hole. Consider the hole shown in Figure 6.18(a). The sequence of machine movements involved in drilling the hole would be:

1. Position to hole location.
2. Lower the spindle at a programmed feed rate.
3. Lift the spindle rapidly to the start position.

Now consider the process of drilling the hole shown in Figure 6.18(b). The same sequence of spindle movements is necessary; the only variation is in the depth of travel. To program such a sequence of moves is quite simple, but if there were a large number of holes to be drilled, apart from the boredom of repeating the necessary data when writing the program, the program itself would be very long. In addition, the fewer data commands that have to be handled the less likely it is that errors will be made. By standardizing the sequence of moves the only additional data requirements are the new hole location, depth of cut, feed rate, and spindle speed. This information, with the appropriate G code, is entered only once. Each time the slide moves to bring the spindle to a new position in relation to the work another hole is drilled to the programmed depth.

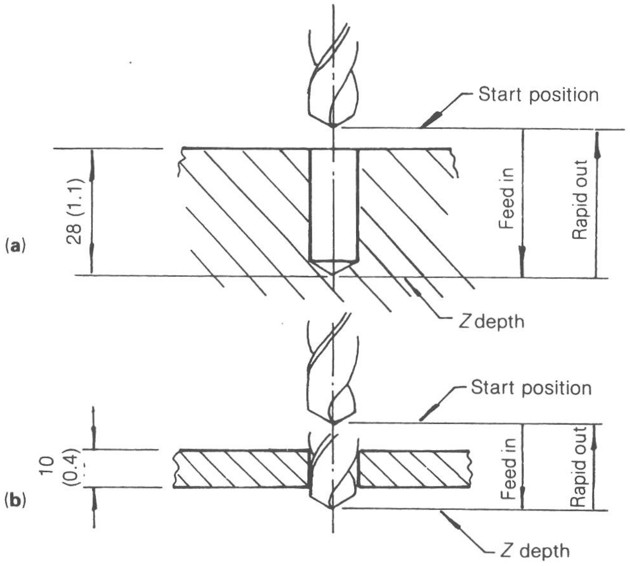


Figure 6.18 *Movements required to drill holes. (Inch units are given in parentheses.)*

Nonstandardized Fixed Cycles

It is often the case that manufacturers of machine control units wish to include in their systems cycles that are not necessarily widely applicable and therefore do not fit into the "standardized" category, but the inclusion of which considerably enhances their control system. The cycles they choose to include will depend on the machine type to which the control is to be fitted. Some of the more common cycles of this nature are discussed below.

Loops The term "loop" is particularly relevant when reducing raw material to size by making a series of roughing cuts. Consider the component shown in Figure 6.19, which is to be reduced from 50 mm (2 in.) to 26 mm (1 in.) diameter by a series of cuts each of 2 mm (0.08 in.) depth. Assuming that the starting point for the tool is as shown, the tool will first move in a distance 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, thereby completing the 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" command in the main program as many

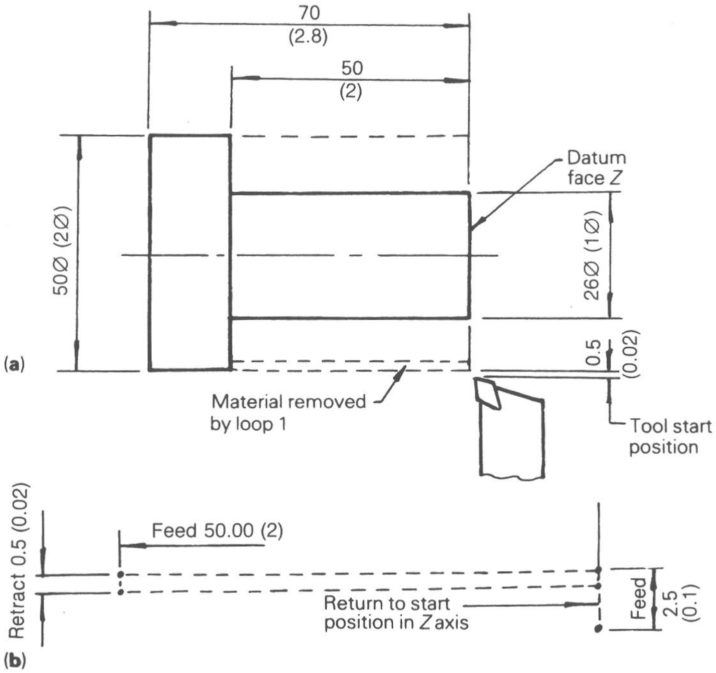


Figure 6.19 *Looping or roughing cycle:* (a) *component and* (b) *loop details, repeated six times. (Inch units are given in parentheses.)*

times as necessary to reduce the work to the required diameter. Note: some controls will do this with special "G" codes, while other controls will use special command codes, but the results are the same.

Face Milling Cycle Figure 6.20 shows details of a face milling cycle. After programming the appropriate G code, together with spindle speed and feed rate, the only other information required are the *X* and *Y* dimensions of the face to be milled. The control unit computer will determine the number of passes necessary and the appropriate cutter step-over to machine the face. The cutter diameter will be picked up automatically from previously entered information. This type of cycle is very commonly found on conversationally programmed controls.

Slot Milling Cycle Figure 6.21 illustrates a slot milling routine. As with face milling, the programmer has to state spindle speed, feed rate, and slot dimensions in the *X* and *Y* axes. The first pass made by the cutter passes through the middle of the slot and then returns to the start. Further passes are made until the correct depth is achieved, the number of passes necessary being determined by the axis increment depth programmed in the cycle. When the correct depth is reached, the cutter path is that of a series of cycles increasing in size with each pass. Some controls vary this process by cutting the entire slot at each depth except the finish pass. Again, as with the face milling, the computer will determine the step-over and the number of cycles necessary to machine the slot to size.

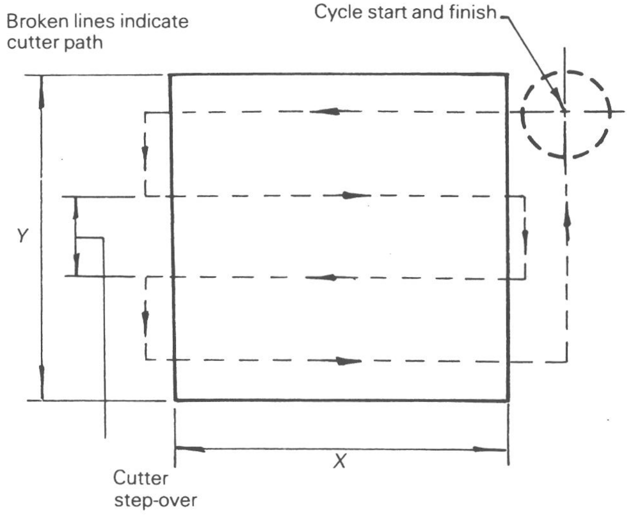


Figure 6.20 *Face milling cycle.*

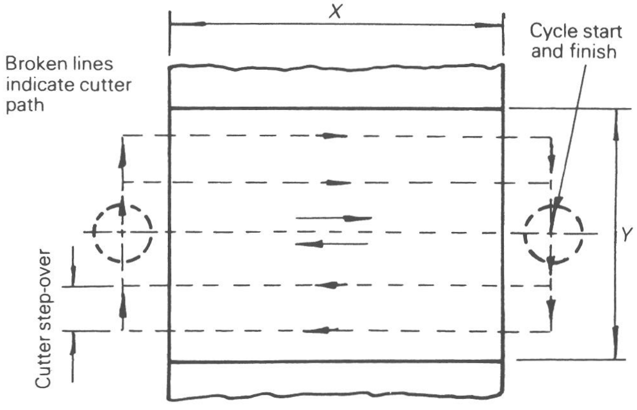


Figure 6.21 *Slot milling cycle.*

Pocket Milling Figure 6.22 illustrates the pocket milling cycle. This cycle starts at the center of the pocket, the cutter feeding in the *Z* axis to a programmed depth. There follows a series of cycles until the programmed *X* and *Y* dimensions are reached, the step-over of up to 80% of the cutter diameter will ensure that a flat surface is produced by providing overlap of passes. Some systems provide for a cycle that roughs out the main pocket and then machines to size with a small finishing cut. If the pocket depth is such that more than one increment in the *Z* axis is necessary, the slide movement returns the cutter to the center of the pocket and the cycle is repeated at the next depth.

Bolt Hole Circles The term "bolt hole circle" means that a number of holes are required equally spaced on a stated pitch circle diameter as illustrated in Figure 6.23. Given that the program has brought the cutter to the pole position,

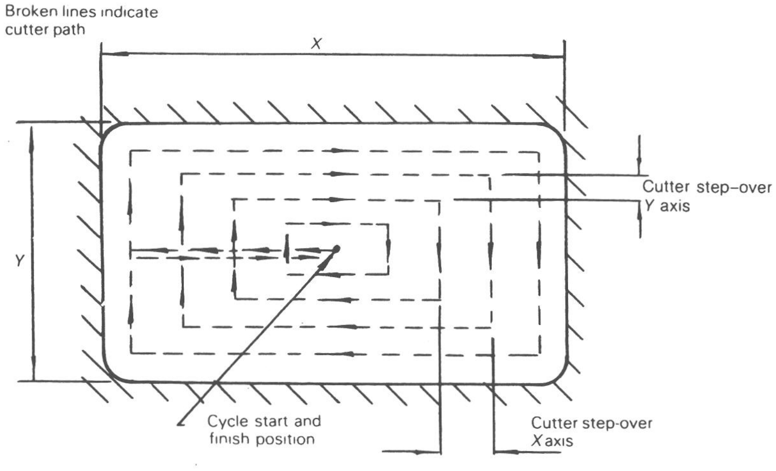


Figure 6.22 *Pocket milling cycle.*

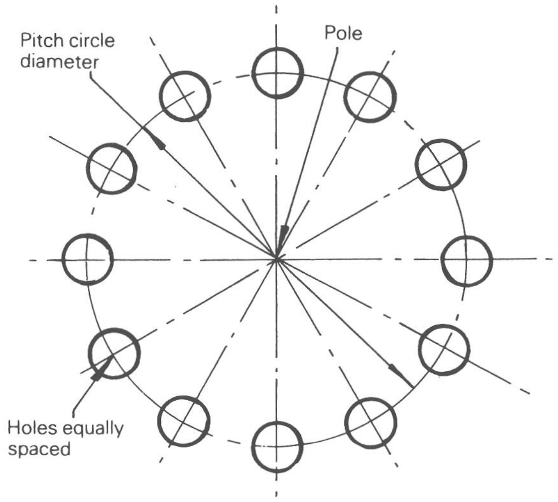


Figure 6.23 *Bolt hole circle.*

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 computer makes all the necessary calculations to convert the polar coordinates to linear coordinates and to move the slides accordingly.

A variation of this cycle will cater for just two or three holes positioned in an angular relationship to one another. An example is detailed in Figure 6.24. Again, the pole position is programmed and the cutter will be at this point when the cycle commences. The additional dimensional data that have to be supplied are the *Z* axis movement, the polar radius and the polar angle(s), and the number of holes required, the computer then converts this information to slide movement in the appropriate axes.

On some control systems it is possible to "rotate" more complex loop programmed features such as the example shown in Figure 6.25.

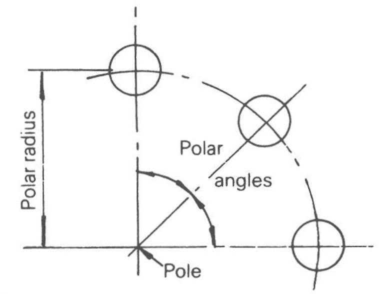


Figure 6.24 *Polar coordinates.*

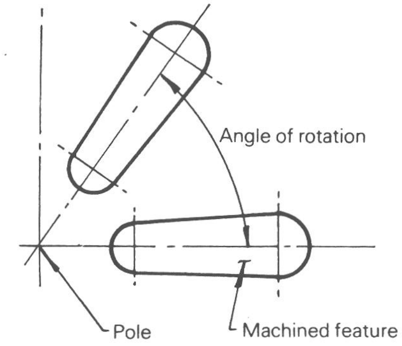


Figure 6.25 *Feature rotation.*

Cycles Devised by the Part Programmer

Cycles devised by the part programmer may be defined as follows. First, there are cycles that are devised specifically for one particular machining task. Second, there are those that may be used when machining a range of components.

Consider the component shown in Figure 6.26, which has a repetitive feature, namely, the recess. When writing a program for machining this particular component, the programmer would devise a cycle, in situations such as this being referred to as a "routine," for producing just one recess. Via an appropriate call the blocks of data defining the routine can be activated as and when required within the main machining program at new locations.

The construction of a routine may include subroutines also specifically constructed by the part programmer and may also utilize any fixed or canned cycles that are considered appropriate. The technique of programming cycles or routines within routines is referred to as "nesting" and is further described subsequently.

Assume the component shown in Figure 6.26 is quite large so that within each recess there were also a number of holes arranged in three groups, as

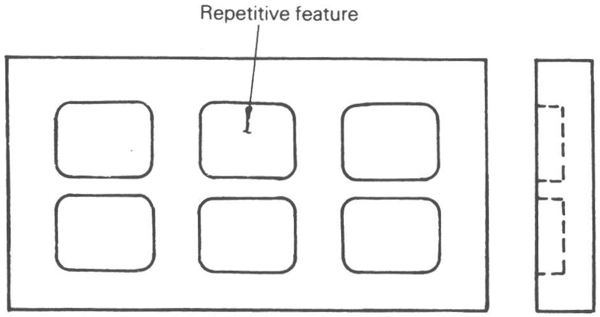


Figure 6.26 *Component with repetitive feature.*

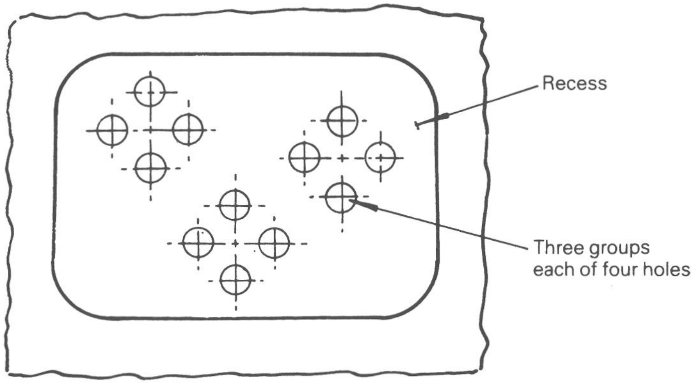


Figure 6.27 *Enlarged detail of component in Figure 6.26.*

shown in 6.27. The main routine would be the data necessary for the production of the recess, as explained above. The subroutine would be the data necessary to produce a group of four holes. The subroutine would be nested within the main routine and called into the main program on three occasions.

However, the production of the four holes is repetitive, and thus it is possible to program to produce just one hole, but to repeat the sequence four times. The complete sequence for producing the component is illustrated diagrammatically in Figure 6.28. On some control systems it is possible to program cycles within cycles as many as eight deep.

Programmer-devised cycles of the second type, to which reference was made above, are useful when a machined feature commonly occurs within the production schedule of a particular company, that is, a machined feature (possibly of unusual design) is required over a range of components. To accommodate this situation some control systems permit routines that are "user defined" to be prepared and "stored within the control system," so that they may be recalled and utilized as and when required as part of a more comprehensive machining program. A routine of this nature is also referred to as a "macro."

A macro may have fixed dimensions or it may have parametric variables, that is, the dimensions may be varied to produce different versions of the same basic feature or component. This technique is referred to as "parametric programming" and is described in more detail in Chapter 9.

MIRROR IMAGE

A commonly occurring aspect of mechanical engineering design is the need for components, or features of components, that are dimensionally identical but geometrically opposite either in two axes or in one axis. By using the mirror-image facility such components or features can be machined from just

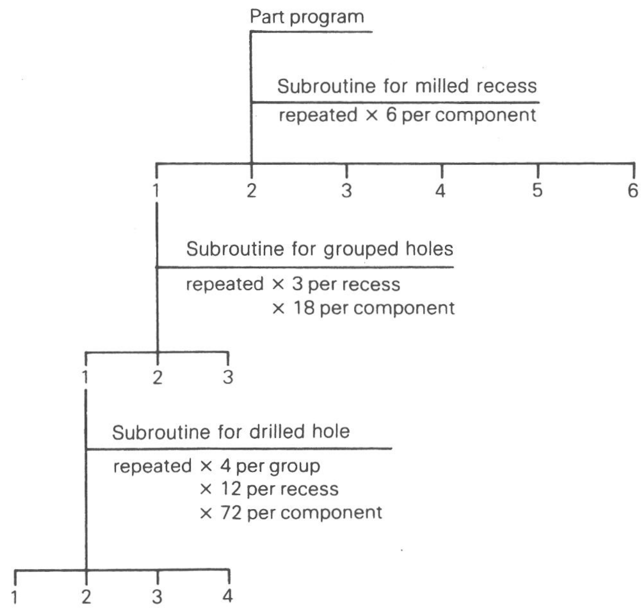


Figure 6.28 *'Nesting' three deep.*

one set of data. The component shown in Figure 6.29 has a feature that is mirrored in two axes. Note that, to produce the second profile, the positive incremental values become negative and the negative incremental values become positive. To produce a feature of the opposite hand, as shown in Figure 6.30, the direction of slide movement changes in one axis only.

SCALING

Another common requirement in mechanical engineering design is components with the same geometrical shape but varying dimensionally. Figure 6.31 illustrates two such components. When a control system is fitted with a scaling facility, it is possible to produce a range of components, varying in size, from one set of program data. The facility can also be used to produce geometrically identical features of components that may be required to be reproduced to different sizes.

SLASH DELETE

The slash or block delete facility enables part, or parts, of a program to be omitted. It is particularly useful when producing components that have slight

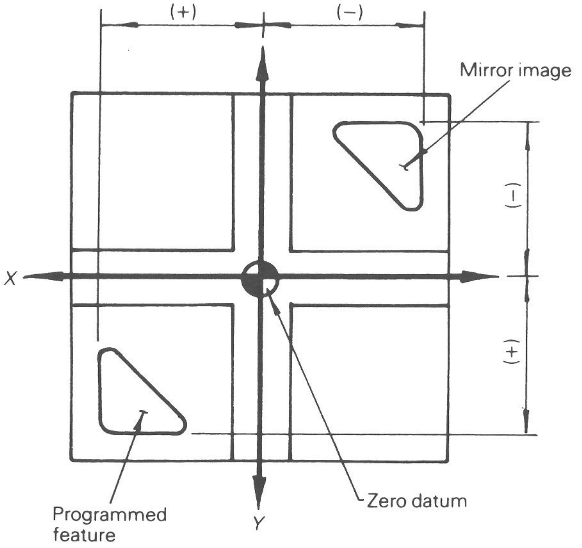


Figure 6.29 *Mirror image in two axes.*

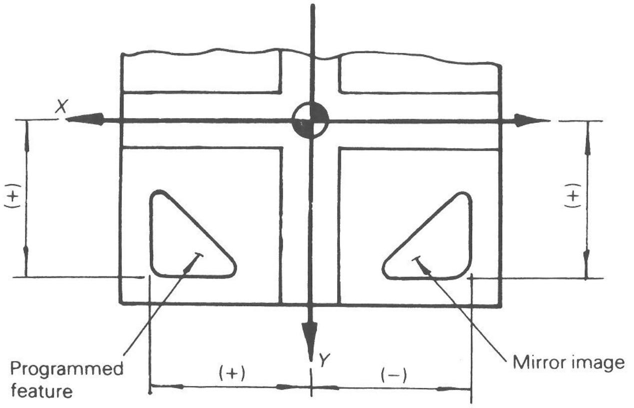


Figure 6.30 *Mirror image in one axis.*

dimensional variations. For example, a hole may be required in one version of a component but not in another, although all other details may be identical. The program data relating to the production of the hole are contained within the programmed symbols/, one at the start of each block concerned. An example is shown below. See manufacturers' programming manual for possible variations in format.

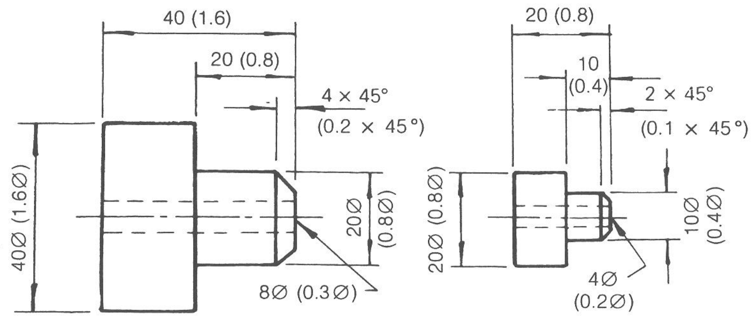


Figure 6.31 *Geometrically identical components suitable for production by scaling. (Inch units are given in parentheses.)*

```
/N05   G01     Z1000 F 150*
/N06   G00     Z-1000 *
```

To make a component *with* the hole, the operator need not take any action. To produce a component *without* the hole, the operator will have to activate the slash delete switch on the control console at the start of the program. When the slash is reached, the control unit will ignore the data that follow. On some systems, if the slash delete is not activated, the program will stop when the slash is reached and the operator then has to make a positive response either to activate the data or to delete them.

This facility is particularly useful when machining castings or forgings, where stock removal requirements may vary, the operator being given the option to include an extra cut or delete it as necessary.

JOG

The jog facility enables the machine operator to move the machine slides manually via the control console. This may need to be done for a variety of reasons, the most obvious one being when establishing datums at the initial setting of the machine. There are also two standard "G" code boring cycles that call for jog retracts after the cycle before returning to automatic operation. It may also be necessary to stop an automatic sequence and move the machine slides to facilitate work measurement, tool changing due to breakage, and so on. Whatever the reason it is desirable that the automatic program is restarted at the point at which it was interrupted, and most control systems have a *return from jog* facility that returns the machine slides to their original positions, this facility being activated manually via a button on the control console.

PROGRAM STOP

Stops in a machining sequence can be predetermined and included in the part program as a miscellaneous function (M00). Scheduled stops for measurement, tool changing (on manual tool change machines), etc., have to be notified to the machine operator so that he or she will be aware of his or her duties at this point.

Program stops can also be optional, that is, the sequence does not have to stop. Optional stops are also included in the part program as a miscellaneous function (M01) and the control will ignore the command unless the operator has previously activated a switch on the control console.

DATUMS

Machine Datum

The machine datum, also referred to as "zero datum" or simply as "zero," is a set position for the machine slides, having a numerical identity within the control system of zero. All slide movements are made in dimensional relationship to this datum as indicated earlier in the chapter, when absolute and incremental positioning moves were discussed.

On some machines the zero datum may be a permanent position that cannot be altered. On other machines a new zero is readily established by moving the slides so that the cutting tool is placed in the desired position in relation to the workpiece and then pressing the appropriate zero button on the control console. The facility to establish a datum in this manner is referred to as a floating zero or zero shift. The location of the original zero is not retained within the control memory.

A fixed machine datum may be helpful to the part programmer, especially when the programming is carried out remote from the machining facility, since the position can be taken into consideration when writing the program. It will be necessary, however, for the programmer to specify the exact location of the component in relation to the machine datum if the program is to achieve the desired results.

A floating zero affords greater flexibility when machine setting, since the work can be positioned anywhere within the range of slide movement and the zero established to suit. But this can be time-consuming and, if incorrectly carried out, may result in machining errors.

Program Datum

The program datum or zero is established by the part programmer when writing the part program, and the program will require all slide movements to be made in relation to that point.

In practice, the machine zero and the program zero are often synchronized by either accurately positioning the work or, when possible, resetting the machine zero. Any unavoidable variations between the two positions can be accommodated by using the zero offset facility, if available, as described below.

ZERO OFFSET

The zero offset facility enables a machine zero datum to be readily repositioned on a temporary basis. Once it has been repositioned, the slide movements that follow will be made in dimensional relationship to the new datum. It is particularly useful when the original machine datum does not coincide with the part program datum, a situation that can arise, for instance, when a part program has been prepared without regard to the normally fixed position of the machine datum and difficulties are encountered in positioning the workpiece to suit the part program. A simple example of this is when a part program for a turned component has been prepared using the forward face of the workpiece as a zero datum when the machine zero is, as is often the case, located at the back face of the chuck. Use of the zero offset facility will reestablish the zero at the work face. The zero offset facility also enables two or more components to be machined at one setting from the same part program. In Figure 6.32 component 1 would be machined with slide movements made in relation to datum 1. On completion of the machining sequence, the machine table would be caused to move, via the part program or by manual intervention at the keyboard to datum 2 where the offset feature would be activated to the predeter-

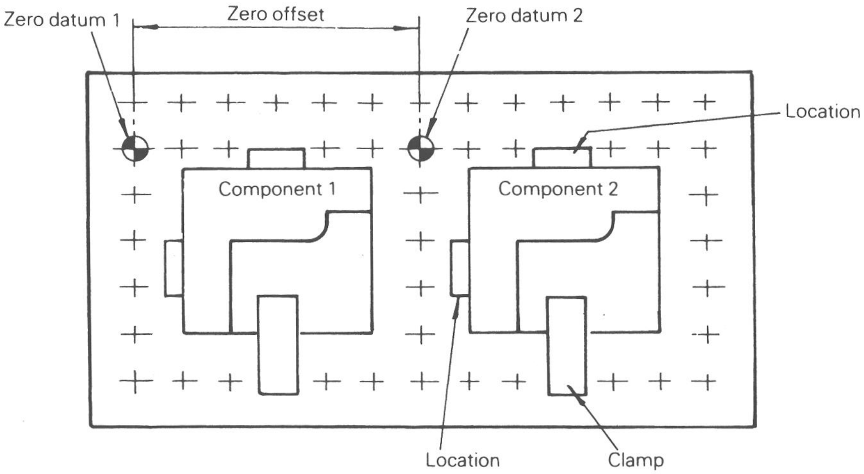


Figure 6.32 *The use of zero offset facility when machining components located on a grid plate.*

mined offset dimension and the machining sequence would be repeated for component 2, with all slide movements being made in relation to datum 2. The control would retain information regarding the location of the original datum, which remains a permanent feature of the part program, and the machine slide can be caused to return to that position.

The facility may also be used to machine identical features on components of different lengths. A simple example would be to cut a screw thread of particular dimensions on the ends of two bars the overall lengths of which are not the same. Similarly, the facility can also be used when turned workpieces are reversed for secondary operations and the second setting in relation to the Z axis zero differs from the first.

On the more sophisticated controls it is possible to establish a new zero, or zeros, at various stages throughout the program. All subsequent moves will be made in relation to the new datum, but these moves are not necessarily a repeat of the moves made before the new zero was established. This facility enables the features of complex or very long components to be machined by relating slide movement to more than one datum, thus simplifying programming and possibly reducing machining time by limiting the length of slide travel.

QUESTIONS

1 What is a preparatory function and how is it designated in word address programming?

2 What is a miscellaneous function and how is it designated in word address programming?

3 What is meant by the term "modal"?

4 Name and describe the three types of positioning control used on computer numerically controlled machine tools.

5 Explain, with the aid of a simple sketch, the difference between absolute and incremental dimension definition.

6 When are the letters *I*, *J*, and *K* used in a word address program?

7 What is meant by the term "ramp suppression"?

8 Describe what happens during a peck drill cycle.

9 What is a looping cycle and when is it used?

10 Describe what happens during a pocket milling cycle.

11 What is a bolt hole circle?

12 What is meant by the "rotation" of a machined feature?

13 With the aid of a simple sketch describe the effect of reproducing a machine feature using the mirror image programming facility in (a) two axes; (b) one axis.

14 What is the programming function that permits the production from one set of data, components geometrically identical but with proportional dimensional variations?

15 Explain the meaning of the term "nesting" as applied to machining cycles.

16 When is the block or slash delete facility likely to be used and how is it generally invoked?

17 What is the jog facility on a machine control system and when is it likely to be used?

18 Give two reasons for including an optional stop in a program.

19 Why is it necessary to inform a machine operator of the scheduled stops in a machining program?

20 With the aid of simple sketches describe the meaning of zero offset.

8

PART PROGRAMMING FOR COMPUTER NUMERICALLY CONTROLLED MACHINING

THE PART PROGRAM

The term part program is used to describe a set of instructions that, when entered into a machine control unit, will cause the machine to function in the manner necessary to produce a particular component or part. Manual part programming is the term used to describe the preparation of a part program without recourse to computing facilities to determine cutter paths, profile intersecting points, speeds and feeds, etc.

The program may be prepared manually and expressed in a coded language that is applicable to the machine controller being used. Alternatively, it may be written in another language or compiled by the use of computer graphics. The result is then post-processed, or translated, to suit the machine controller.

Included in the part program will be the necessary dimensional data relating to the features of the component itself, together with control data that will result in the machine making the slide movements required to produce the component. These data will be supplemented by instruction data that will activate and control the appropriate supporting functions.

Programs as entered into machine control units involve either of two programming concepts:

(a) word address
(b) conversational manual data input (MDI)

There are considerable variations between the two methods.

Whether a production scene incorporates total automation or merely one or two numerically controlled machines positioned among traditional machines, at the heart of successful numerical control is efficient competent part programming. The practical *skill* level requirement on the shop floor is, without doubt, in decline, but a high level of practical *knowledge* is essential if part programmers are to use costly equipment at their disposal to the best advantage. The selection of a correct sequence of operations, together with efficient cutting speeds and feeds, tooling and work holding, and the ability to express these requirements in the correct format are of paramount importance.

Unfortunately, programming methods differ and even when the basic approach is similar (for example, with word address), there are still variations and peculiarities, and conversational manual data input is very individual.

Thus the reader should appreciate that the ability to program with one control system, although there is much carryover, rarely means that knowledge can be used *in total* elsewhere. Specialist training is essential, and most machine-tool manufacturers respond to this by offering training courses as part of the overall package to customers buying their equipment. However, once the basic concepts involved in part programming are understood, the change from one system to another does not appear to be a major problem. Indeed, the variations encountered can be a source of much interest, while the mastery of yet another system can give considerable personal satisfaction.

PROCEDURE

Taking as a starting point the detail drawing of the component to be manufactured, the tasks that confront the part programmer may be listed as follows:

1. Select a machine capable of handling the required work.
2. Determine the machine process to be used.
3. Determine work holding and location techniques.
4. Determine tooling requirements and their identity.
5. Document, or otherwise record, instructions relating to work holding, work location, and tooling.
6. Calculate suitable cutting speeds and feed rates.
7. Calculate profile intersecting points, arc centers, etc.
8. Determine appropriate tool paths including the use of canned cycles and subroutines.
9. Prepare the part program.
10. Prove the part program and edit as necessary.
11. Record the part program for future use.

Although these stages have been given a separate identity, they are very much interrelated and cannot be treated in isolation. A diagrammatic impression of the approach to be adopted is given in Figure 8.1.

MACHINE SELECTION

In selecting the machine to be used the first consideration is the type of work that has to be carried out. The tolerance and surface finishes required on the part will determine the type of machine and process to be used. Even when the type of machine is established, its specifications will need to be reviewed to ensure that part accuracy can be maintained.

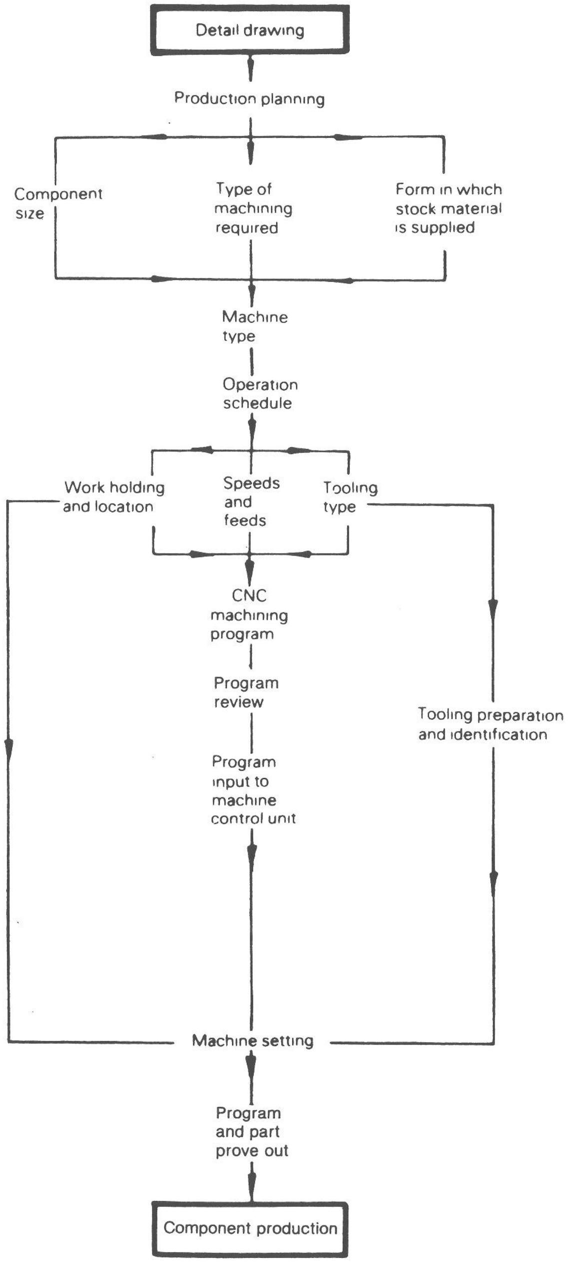


Figure 8.1 *Procedures associated with part programming.*

For relatively simple components the choice will be obvious and is likely to involve just one machine. On the other hand, more complex designs may require machining to be carried out on a second or perhaps third machine. It may be necessary to move from one machine to a second before returning to the original machine for further work, and so on.

Such transfers, and the stage at which they will take place, need to be determined clearly, since they will have a direct bearing on the preparation of appropriate machining programs.

Machine selection will also be influenced by component size, and the programmer must ensure that any machine used has the necessary physical capacity to accommodate the workpiece.

Decisions made at the component design stage relating to materials—whether the form of supply will be a casting or a solid bar, for instance—will also need to be considered, since this may have some bearing on work-holding and machine-loading arrangements.

PROCESSING OF MACHINING OPERATIONS

Having selected a machine capable of handling the required work, the next task confronting the part programmer is to decide on a suitable sequence of operations.

In order to do this effectively the programmer should ideally have a thorough understanding of the capabilities and operating procedures associated with the particular machine to be used, and adequate knowledge of the work-holding equipment and tooling that can be employed.

It is often the case that, giving due regard to safety requirements, a machining task can be tackled in more than one way with equally good results in terms of dimensional accuracy and surface finish. But the programmer must always bear in mind one objective is to complete the machining as quickly and efficiently as possible. There are two basic planning techniques that, when carefully considered, can make a significant contribution to achieving this objective.

The first is to carry out as much machining as possible at one work setting and to avoid unnecessary repositioning of the work, since this can be a very time-consuming business. The second is to carry out as much machining as possible with each cutting tool called, and to avoid unnecessary tool changing or indexing. The programmer should bear these points firmly in mind when listing the sequence of operations to be adopted.

The compilation of the process of operations to be used will not only be an aid to logical thinking throughout the rest of the part programming process, but it is also likely to be of value to the machine operator and may be required as a record for future reference. The more complex the component, the more vital the compilation of the process becomes.

It is likely that the operations process will form just part of the general documentation relating to a particular job, which will also contain information relating to work-holding, tooling, speeds, and feeds. The documentation relating to these aspects of part programming are discussed subsequently.

WORK-HOLDING AND LOCATION

The part programmer's responsibilities regarding work-holding and location are as follows:

(a) determine the work-holding device or devices to be used;
(b) determine if there will be a need to use supplementary support at any stage during a machining sequence;
(c) determine the means of ensuring accurate location of the workpiece prior to machining;
(d) document all matters relating to work setting that will have a direct effect on the validity of the part program and that will, therefore, be of importance to the machine set-up person.

Decisions made in relation to these factors are greatly influenced by component shape and size. Components of regular shape are usually accommodated in standard work-holding devices such as chucks, collets, and vises. Components of irregular shape often require special work-holding arrangements, and as a result demand extra attention from the programmer. He or she may find it necessary to include special slide movements in the program, solely to avoid collisions between the cutting tools and the clamping.

Similarly, the programmer will need to give special attention to components requiring supplementary support—the use of a center support or steady rest, for example—and may well have to include control of these features within the part program.

Multicomponent settings will also have a direct effect on the approach adopted when preparing the part program.

A special characteristic of CNC machining involving very high rates of metal removal is that considerble cutting forces may be exerted in a number of directions during the production of a single component, with very rapid change from one direction to another, possibly occurring without the safeguard of manual observation or intervention. This variation in cutting force direction means that the prime objective in work location, that of ensuring that the cutting forces are directed against an immovable feature in the work-holding arrangement, may not always be met when using standard equipment. For example, work held in a conventional machine vise is only positively located when the cutting forces are directed against the vise jaw. If the cutting force changes direction so that it is at 90° to the fixed jaw, there will be a frictional hold only, which is not foolproof.

When confronted with the problem of multidirectional cutting forces, the programmer should give full consideration to the alternative approaches available. Devices such as the grid plate will provide for positive location in several directions, but it may be necessary to use a specially devised fixture. A number of the project components included in Appendix C will require this approach.

It is possible that the work-holding equipment available is very limited in range, such as a machine vise. In this situation the programmer will have to make the best of arrangements such as the frictional hold described previously. For example, a reduction in metal removal rates will reduce the cutting forces exerted on the workpiece. Each problem encountered will require individual assessment, and the methods used to overcome the problem should be selected with reference to the high safety standards that are so essential in CNC machining.

Another factor that must be considered is that of geometric tolerances, as listed in Appendix D. When any of these are encountered on a part drawing, the programmer must ensure that the work-holding and location arrangements being used will enable them to be achieved. It is a further area of part programming that requires the programmer to be well versed in the practical side of CNC machining, and to have a full understanding of the capabilities and limitations of the work-holding devices that may be used.

In order that specified geometric requirements are satisfied, it may be necessary to adopt a special approach to work setting, or, as is more likely, work resetting before carrying out further operations. In such cases it is imperative that the part programmer indicates to the machine set-up person or operator his or her reasons for doing so. Such information is included in the general documentation relating to that particular workpiece.

The importance of positive location of the workpiece to absorb the forces exerted by the metal-cutting action has already been stressed. There is, however, another reason why the part programmer is concerned about precise location of the work. He or she will program the slide movements in relation to a datum that will be determined when the part program is prepared, and unless the part to be machined is precisely positioned in relation to that datum the intended machining features will not be achieved. Subsequent parts must also be positioned in exactly the same way to ensure uniformity of the product.

When establishing a program zero datum, the programmer will have to take into consideration the reference zero position that is an incorporated feature of the machine control system. The machine zero may or may not be in a fixed position. If it is fixed, it may be capable of being shifted on a temporary basis via the part program using the G92 preset code. It may be capable of being established anywhere within the operating range of the machine, or there may be limitations on repositioning. Whatever the circumstances, the programmer will need to understand them completely.

Consider first a control system that permits a machine zero to be established anywhere the programmer chooses. In this situation it may be considered that the correct programming approach is to establish a machine zero that will correspond with the chosen program zero. So for a component such as the one illustrated in Figure 8.2 the programmer selects the corner of the workpiece as zero for all programmed moves in the *X* and *Y* axes, and a 2 mm (0.1 in.) clearance between the top of the work and the *Z* axis zero. By selecting the upper left-hand corner of the part as program zero and machine zero, the programmer can use many part print dimensions in the part program. To ensure that there is correlation between the two zero positions the following machine setting approach will be necessary.

1. Set the corner of the vise jaw to zero in the *X* and *Y* axes (achieved by using a center locator or wiggler, or possibly an electronic probe).
2. Set the *Z* axis zero 2 mm (0.1 in.) above the work surface (achieved by touching on to a suitable worksetting block and calibrating the tool length offset accordingly).
3. Locate all workpieces using the corner of the fixed jaw of the vise as a reference position. (A plate attached to the vise jaw may be used to simplify this process).

The setting arrangement that accommodates the *X* and *Y* axes requirements is illustrated in Figure 8.3.

Consider now a situation involving a turned component such as that illustrated in Figure 8.4, and assume that the programmer has chosen to establish the face of the part as the *Z* datum zero and that the machine spindle center line is the *X* axis zero, as is normal. All that is required of the programmer is to ensure that the machine set-up person or operator is aware that the program datum is at the face of the work. The set-up person or operator will be required to establish the *Z* axis zero at the machine in the manner appropriate to that particular machine, and then ensure that all workpieces are all set to a measured overhang or to a stop as illustrated in Figure 8.5.

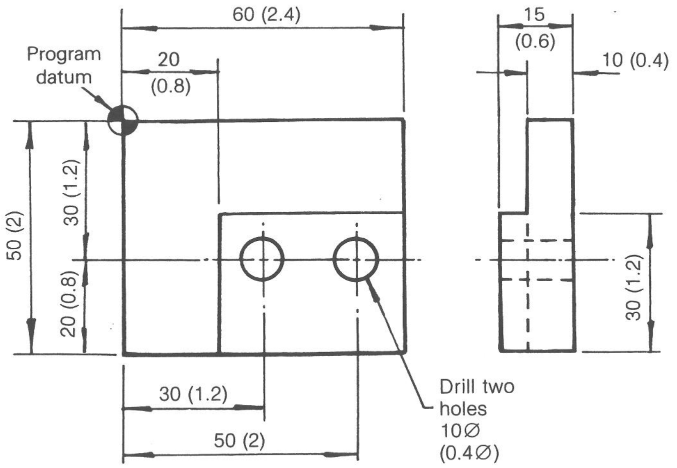


Figure 8.2 *Component detail. (Inch units are given in parentheses.)*

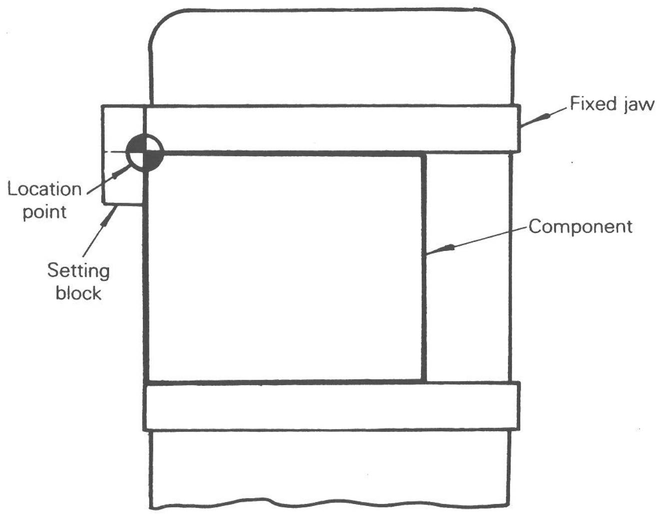
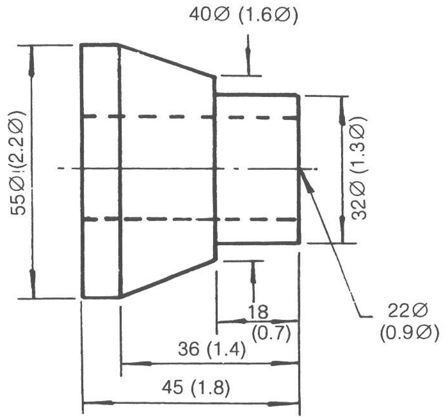


Figure 8.3 *Use of stop block on fixed jaw for component location.*



Material: medium carbon steel

Figure 8.4 *Component detail. (Inch units are given in parentheses.)*

It is often the case that turning centers have a set zero datum for the machine, usually at the back face of the chuck or a reference surface on the spindle nose.

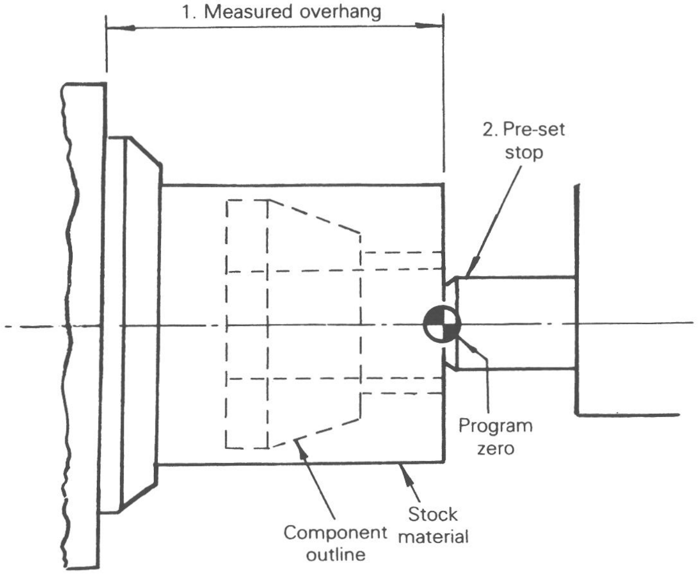


Figure 8.5 *Alternative work setting techniques to establish a datum for turned work.*

This type of zero cannot be changed but can be shifted on a temporary basis using the G92 preset axis code.

The programmer may choose to use the back face of the component as the program zero in the *Z* axis, a technique often applied when work is being produced from prepared billets. Work location is simple, and simply involves ensuring the material is firmly placed against the reference face. A further bonus is that all programmed slide movements will be positive.

To use the facility of repositioning the zero on a temporary basis—so that it corresponds to a program zero established at the workpiece face for instance—it will be necessary for the programmer to determine the amount of shift required to accommodate all the programmed movements in the *Z* axis. The dimensional value of the shift required, that is, the work overhang, must be documented. Eventually it will be entered into the program through the use of the G92 code followed by *X*, *Z*, and/or *Y* axis positional data as to what current slide positions should be. Some older machines may still use another method of establishing zero shifts through special offset tables, which can be activated by assigned codes like E, F, or H.

To ensure that the programmed machine movements achieve the desired effect, the work material has to be positioned accurately, either manually or automatically against stops, and this function is the responsibility of the machine set-up person or operator. The accuracy of this method of work setting can be improved if the overhang is slightly larger than the actual work requires, allowing a facing cut to be used early in the machining sequence to establish the new zero precisely.

It may be necessary to provide for more than one zero shift within the same turning program. A common situation is when the component length is such that, to ensure adequate support and to avoid chatter, part of the machining is carried out with a reduced overhang. After a programmed stop in the machining cycle, the operator repositions the work to suit the second zero position. Alternatively, the repositioning of the work may be achieved automatically through the program. This is particularly appropriate when a bar feed is utilized, the bar feeding to appropriate stops. The provision of a center support may also be a feature of such an arrangement. After the second zero shift all subsequent moves will be made in relation to that datum.

An example of a component which would involve two zero shifts during machining is shown in Figure 8.6. Because the diameter of the component is relatively small in proportion to its length, it would be advisable to use two settings and a center support for the second sequence of machining operations. The first setting involving the shift of the machine zero to the work face is illustrated in Figure 8.7(a), while the second setting requiring shifting the zero for a second time is shown in Figure 8.7(b).

The use of a second program zero is also applied to milling operations. An initial program zero is established and some machine movements will be made in relation to that datum. Then, via an appropriate program call the zero will be reestablished and all subsequent moves will be made in relation to the second datum.

One milling situation where the zero shift facility is particularly useful is when more than one component is to be machined at one setting, as illustrated in Figure 8.8. In this example a grid plate is used as a work-holding device. The advantage of the grid plate is that all the clamping and location points can be identified using a letter/number grid reference, like using a map reference

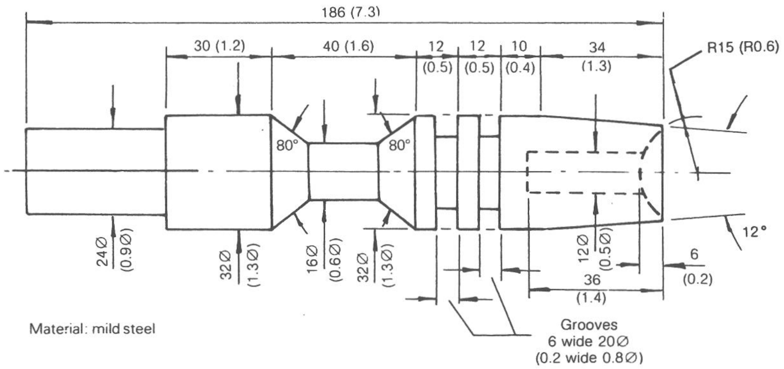


Figure 8.6 *Component detail. (Inch units are given in parentheses.)*