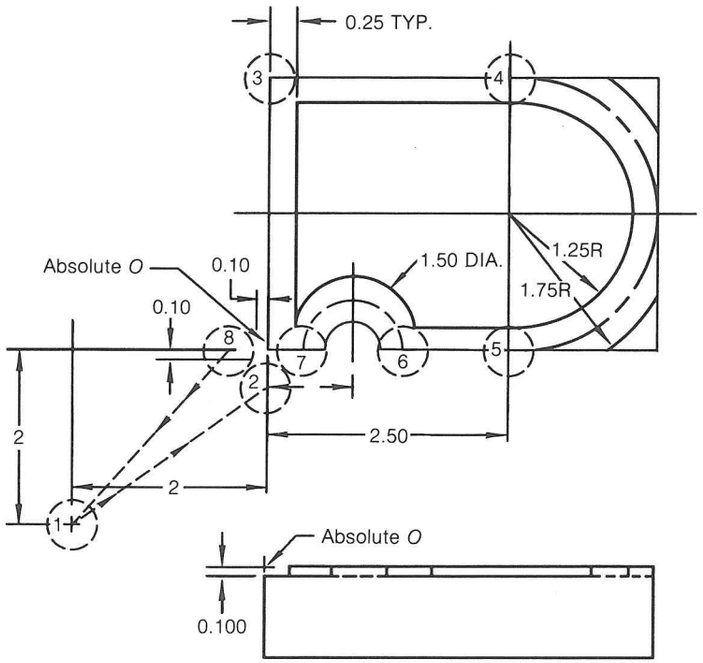


Figure 8.84 *Wells cut exercise.*

Wells Example of Program Inputs

Operation to be performed	Message on CRT	Operator keystrokes
With machine powered up, define tool to be used.	N1	Tool Definition Button
	TOOL NUMBER?	1 Enter
	TOOL LENGTH?	0 Enter
	TOOL RADIUS?	.25 Enter
	BLOCK COMPLETE?	Enter
Call tool to be active for machining.	N2	Press Tool Call
	TOOL NUMBER?	1 Enter
	WORKING SPINDLE AXIS X/Y/Z?	Z Enter
	SPINDLE SPEED S RPM?	1600 Enter
	BLOCK COMPLETE?	Enter
Move from tool change to position two on part.	N3	Press Linear Motion
	FIRST COORDINATE?	X0 Enter
	SECOND COORDINATE?	Y-.350 Enter

Operation to be performed	Message on CRT	Operator keystrokes
	TOOL RADIUS COMP RL/RR/NO COMP?	Enter
	FEED RATE?	220 Enter
	AUXILIARY FUNCTION M?	03 Enter
	BLOCK COMPLETE?	Enter

The question answer process is continued until the program is completed.

Wells Example Program Read-out

N1 TOOL DEF 1	L + 0.0000 R + 0.2500		Define Tool #1
N2 TOOL CALL 1	Z S1600.000		Active Tool #1
N3 L X + 0.0000	Y-0.3500		Move to Pos. 2
		R0 F2200 M039	Spindle on.
N4 Z+0.1000			Z clearance
		R0 F1500 M08	Coolant On
N5 Z0			Z to depth
		R0 F250 M	
N6 Y+3.0000			
		R0 F250 M	Feed to Pos. 3
N7 X + 2.5000			Feed to Pos. 4
		R0 F250 M	
N8 CC X+2.5000	Y+1.5000		Define circle center
N9 C X+2.5000	Y0		Feed in circle to
		DR- F250 M	Pos. 5
N10 X+ 1.5000			Feed to Pos. 6
		R0 F250 M	
N11 CC X+1.0000	Y0		Define circle center
N12 C X+.5000	Y0		Feed in circle
		DR+ F250 M	to Pos. 7
N13 X-.350			Feed to Pos. 8
		R0 F250 M09	coolant off
N14 Z+2.0000			Retract Z
		R0 F1500 M05	Spindle off
N15 L X-2.0000	Y-2.0000		Move to tool
		R0 F2200 M02	change
			End of Program

Example 2 The second conversational control program example is for a Hurco CNC mill with the Ultimax control. This machine has dual CRT screens so that positional and program data can be reviewed on one and tool path graphics on the other.

The part example to be used for the Hurco program is the base component of a drill vise (Figure 8.85). The operations to be performed are the milling of a 1.375 × 3.837 pocket with a .500 endmill, the milling of a .500 slot with a .432 endmill, and the milling of eleven .125 radius adjusting notches with a .250 ball nose end mill. See tool graphics for tool change position and program zero location.

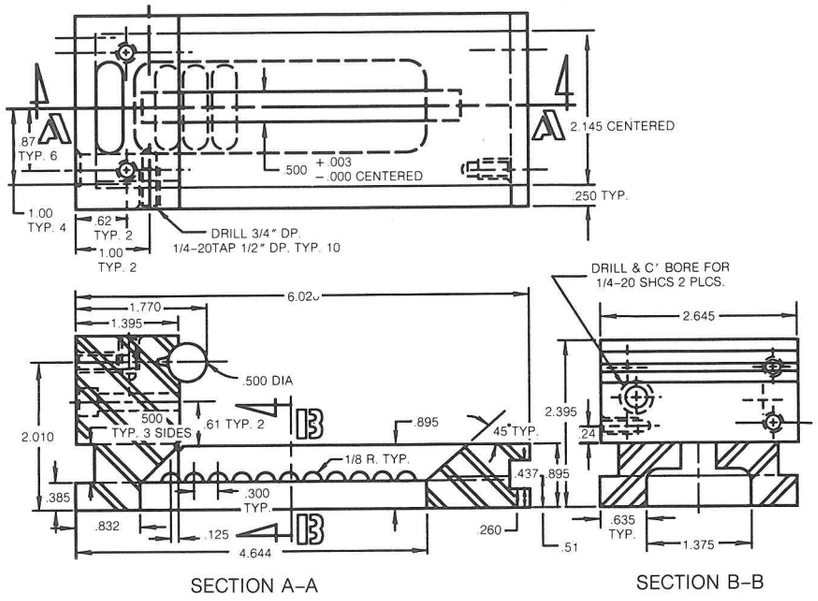


Figure 8.85 *Base component drill vise.*

Hurco Example of Program Inputs

Operation to be Performed	Message on CRT	Operator Key Strokes
With machine powered up, define tool to be used (Figure 8.86).		Push tool setup soft key
	Upper screen displays machine status Lower screen display reads tool setup display figure	
	Cursor at tool	Key in 1 and enter.
	Cursor at Type	Push End Mill key
	Cursor at Diameter	Key in 0.500
	Cursor at zero calibration	Jog tool down against reference point and push SET TOOL ZERO
		Push TOOL UP key to retract spindle
	Cursor at speed	Push CLOCKWISE key and then enter 0 speed to get control speed calculation
	Cursor at coolant	Push MIST key
	Cursor at material	Push HSS key
	Cursor at flutes	Enter number of flutes in cutter

TOOL SETUP

	MACHINE	PART		
X	0.0000	0.0000	TOOL IN SPINDLE	0
Y	0.0000	0.0000	SPINDLE	0
Z	0.0000	0.0000	FEED	0.0
			MAGAZINE	0

TOOL		>1
TYPE		
PITCH		0.000
DIAMETER		0.0000
ZERO CALIBRATION		0.0000
SPEED (RPM)	CW	0
COOLANT		OFF
MATERIAL		
FLUTES		2

DELETE TOOL
PART SETUP
PART PROGRAMMING
TOOL UP
SET TOOL ZERO

Figure 8.86 *Tool setup screen.*

Operation to be Performed	Message on CRT	Operator Key Strokes
Repeat process for tools 2 and 3		
Position tool away from part (Figure 8.87).	Block 1	Push POSITION soft key
	Cursor at TOOL	Enter the tool number followed by the *X* and *Y* coordinates for the position
	Cursor at STOP	Enter NO by pushing button so machine does not stop at this position
	Cursor at INDEX PULSES	Enter 0 no index table is involved.
Create MILL FRAME (Figures 8.88 and 8.89)	Block 2	Press MILLING soft key to get main milling menu
	Choose type of block	Press FRAME soft key
	Cursor at TOOL	Enter 1—using key pad
	Cursor at TYPE	Press POCKET
	Cursor at FINISH PASS	Press YES soft key
	Cursor at X CORNER	Enter −.563—to establish lower left hand corner of pocket in *X* axis

BLOCK 1 POSITION

TOOL >0

X 0.0000

Y 0.0000

STOP NO

INDEX PULSES 0

Figure 8.87 *Positioning.*

BLOCK 2 MILL FRAME

TOOL		1	Z START	0.0000
TYPE	>	ON	Z BOTTOM	0.0000
FINISH PASS		NO	PLUNGE FEED	0.0
X CORNER		0.0000	MILL FEED	0.0
Y CORNER		0.0000	SPEED (RPM)	200
X LENGTH		0.0000	PECK DEPTH	0.0000
Y LENGTH		0.0000	(TOOL DIAMETER	0.2500)
CORNER RADIUS		0.0000		

1) ON
2) INSIDE
3) OUTSIDE
4) INSIDE TANGENT
5) OUTSIDE TANGENT
6) POCKET

Enter cutter placement

Figure 8.88 *Mill frame.*

BLOCK 2 MILL FRAME

TOOL		1	Z START	0.0500
TYPE	>	ON	Z BOTTOM	−0.9500
FINISH PASS		YES	PLUNGE FEED	1.5
X CORNER		−0.5630	MILL FEED	8.0
Y CORNER		−0.6870	SPEED (RPM)	500
X LENGTH		3.8370	PECK DEPTH	0.1000
Y LENGTH		1.3750	(TOOL DIAMETER)	0.2500
CORNER RADIUS		0.2500		

1) ON
2) INSIDE
3) OUTSIDE
4) INSIDE TANGENT
5) OUTSIDE TANGENT
6) POCKET

Enter cutter placement

Figure 8.89 *Mill frame.*

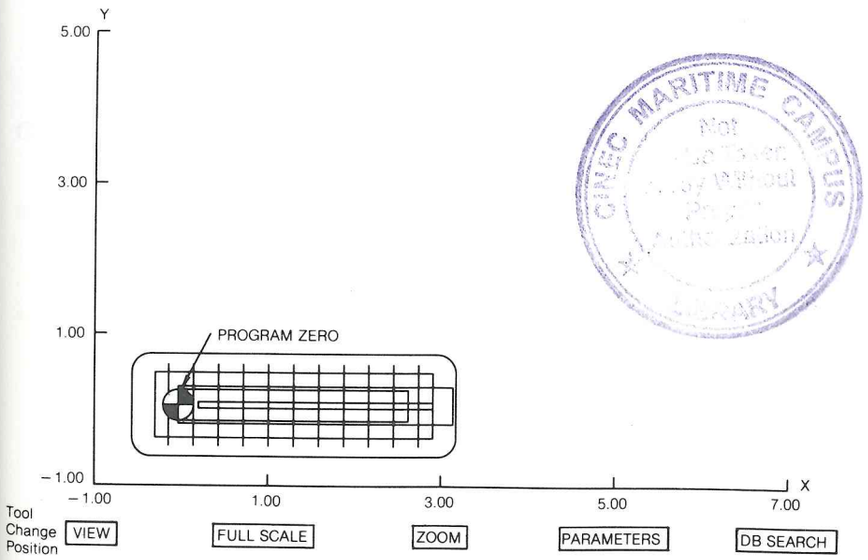


Figure 8.90 *Mill pocket tool graphics.*

Operation to be Performed	Message on CRT	Operator Key Strokes
	Cursor at Y CORNER	Enter −.687—to establish lower left hand corner of pocket in *Y* axis
	Cursor at X LENGTH	Enter 3.837—the "*X*" length of the pocket to be cut
	Cursor at Y LENGTH	Enter 1.375—the "*Y*" length of the pocket to be cut
	Cursor at CORNER RADIUS	Enter .250—the cutter radius
	Cursor at Z START	Enter .05—the starting depth of cut
	Cursor at Z BOTTOM	Enter −.950—the final depth of pocket
	Cursor at PLUNGE FEED	Enter 1.5—feed in IPM for plunging to depth
	Cursor at MILL FEED	Enter 8.0—feed in IPM for *X* and *Y* cutting moves
	Cursor at SPEED (RPM)	Enter 500—cutter rpm
	Cursor at PECK DEPTH	Enter .100—*Z* axis in feed increment for each pass.

This type of program entry process is continued until the entire program is complete.

Hurco Example Program Readout

HURCO ULTIMAX PART PROGRAM **INCH** **STANDARD**

PART SETUP

PART ZERO X	7.0481	SAFETY WORK REGION	
Y	5.9507	Z TOP (+)	999.0000
		Z BOTTOM (−)	−999.0000
		X LEFT (−)	−999.0000
		X RIGHT (+)	999.0000
		Y FRONT (−)	−999.0000
		Y BACK (+)	999.0000
		MATERIAL	CARBON STEEL

PROGRAM PARAMETERS

GENERAL	OVERRIDE LOCKOUT	OFF
	PROGRAM PROTECT	PARTIAL
	RETRACT CLEARANCE	99.0000 IN
	RAPID TRAVERSE	250.0 IPM
HOLES	BORE ORIENT RETRACT	0.0200 IN
	DRILL DWELL	0.5 SEC
	BORE DWELL	1.0 SEC

MILLING	BLEND OFFSET	0.1250 IN
	BLEND OVERLAP	0.1250 IN
	FINISH FEED	100%
	FINISH SPEED	100%
	FINISH XY	0.0050 IN
	FINISH Z	0.0050 IN
	MILLING DIRECTION	CONV
	POCKET OVERLAP	10%

TOOL SETUP

				(Tool Definitions)
1	TYPE	END MILL	COOLANT	MIST (Tool # 1
	DIAMETER	0.5000	MATERIAL	HSS .500 End
	ZERO CALIBRATION	1.2032	FLUTES	2 Mill)
	SPEED (RPM) CW	0		
2	TYPE	END MILL	COOLANT	MIST (Tool #2 .432
	DIAMETER	0.4320	MATERIAL	HSS end mill)
	ZERO CALIBRATION	1.4859	FLUTES	2
	SPEED (RPM) CW	0		
3	TYPE	END MILL	COOLANT	MIST (Tool #3
	DIAMETER	0.2500	MATERIAL	HSS .250 ball
	ZERO CALIBRATION	1.6097	FLUTES	2 nose end
	SPEED (RPM) CW	0		mill)

DATA BLOCKS

1	POSITION			
	TOOL	1	STOP	NO (Tool change
	X	−2.0000	INDEX PULSES	0 .5 end
	Y	−2.0000		mill)
2	MILL FRAME			
	TOOL	1	Z START	0.0500 (Milling of
	TYPE	POCKET	Z BOTTOM	−0.3850 1.375 ×
	FINISH PASS	YES	PLUNGE FEED	1.5 3.837
	X CORNER	−0.5630	MILL FEED	8.0 pocket)
	Y CORNER	−0.6870	SPEED (RPM)	500 (See
	X LENGTH	3.8370	PECK DEPTH	0.1000 graphics
	Y LENGTH	1.3750		of tool
	CORNER RADIUS	0.2500		path,
				Figure
				8.90)
3	POSITION			
	TOOL	1	STOP	NO (Tool #2
	X	−2.0000	INDEX PULSES	0 change
	Y	−2.0000		position)
4	MILL CONTOUR			(Mill .500 slot)
	SEGMENT 0	START		
	TOOL	2	Z START	−0.3500
	CUTTER COMP.	LEFT	Z BOTTOM	−0.9500
	FINISH PASS	YES	PLUNGE FEED	1.5
	X START	0.0000	SPEED (RPM)	500
	Y START	0.0000	PECK DEPTH	0.0750
	SEGMENT 1	LINE		
	X END	0.0000	XY LENGTH CAL	0.2510

	Y END		−0.2510	XY ANGLE CAL	−90.000	
	Z END		−0.9500	FEED	8.0	
	SEGMENT	2	LINE			
	X END		3.2490	XY LENGTH CAL	3.2490	
	Y END		−0.2510	XY ANGLE CAL	0.000	
	Z END		−0.9500	FEED	8.0	
	SEGMENT	3	LINE			
	X END		3.2490	XY LENGTH CAL	0.5020	
	Y END		0.2510	XY ANGLE CAL	90.000	
	Z END		−0.9500	FEED	8.0	
	SEGMENT	4	LINE			
	X END		0.0000	XY LENGTH CAL	0.3.2490	
	Y END		0.2510	XY ANGLE CAL	180.000	
	Z END		−0.9500	FEED	8.0	
	SEGMENT	5	LINE			
	X END		0.0000	XY LENGTH CAL	0.2510	
	Y END		0.0000	XY ANGLE CAL	−90.000	
	Z END		−0.9500	FEED	8.0	
5	POSITION					
	TOOL		2	STOP	YES	(Tool #3
	X		−2.0000	INDEX PULSES	0	change,
	Y		−2.0000			.250 End Mill)
6	PATTERN LOOP LINEAR					
	NUMBER		11	ANGLE	0.000	(Pattern loop
	X DISTANCE		0.3000	DISTANCE	0.3000	definition
	Y DISTANCE		0.0000			for .250 notches)
7	MILL CONTOUR					
	SEGMENT	0	START		−0.370	
	TOOL		3	Z START	−0.5100	(Machining
	CUTTER COMP.		NO	Z BOTTOM	1.5	moves for
	FINISH PASS		YES	PLUNGE FEED		top notches)
	X START		−0.1250	SPEED (RPM)	1200	
	Y START		0.5370	PECK DEPTH	0.0300	
	SEGMENT	1	LINE			
	X END		−0.1250	XY LENGTH CAL	1.0740	(Machining
	Y END		−0.5370	XY ANGLE CAL	−90.000	moves for
	Z END		−0.5100	FEED	4.0	bottom notches)
8	PATTERN END					(End of pattern and program)

QUESTIONS*

1 Devise a simple diagram to illustrate the axes of movement of a vertical machining center and explain why it is necessary to redefine some of these movements as an aid to part programming.

2 Make a simple sketch to show the difference between absolute and incremental dimensional data.

3 How is the difference between an incremental and absolute value indicated on word address control systems that permit the use of either in the same part program?

4 What value should be programmed when a drawing states an upper and lower limit to a toleranced dimension?

5 Briefly explain the difference in data required by the three methods of circular interpolation which use I, J and K values.

6 Explain when the use of a programmer-devised sub-routine would be justified.

7 How does a 'macro' differ from other types of programmer-devised routines?

8 Describe the concept of 'parametric' programming and suggest when its use would be advantageous.

9 Describe the programming technique of 'point definition' and describe the type of situation where it could be used to advantage.

10 What is the advantage of using a datum shift within a program?

11 What is the purpose of the: Operation Schedule? Tool Sheet? Coordinate Sheet?

12 What are the five responsibilities of a part programmer related to tooling?

13 What is meant by the term word address programming?

14 What is meant by the term conversational programming?

15 Write a definition for the following terms: Point to Point; Linear Interpolation; Circular Interpolation.

16 What is the difference between datum shift and the program datum point.

17 Explain what a canned cycle is.

18 What is the major purpose of tool offsets?

19 Describe the difference between translation and rotation of program segments.

20 Describe two basic planning considerations which facilitate producing a part in the shortest possible time.

21 Explain what is meant by a 'floating zero' and state the advantages of such a facility.

22 With the aid of a simple sketch, or sketches, describe a situation where two programmed zero shifts would be required during the production of a turned part.

23 Assuming a situation where the part programmer is not in close contact with the machine shop, what documentation should be prepared to facilitate the transfer of essential information between the two activities of programming and production?

24 Explain how the tool offset facility can be used to program a series of cuts along a turned profile using the same programmed slide movements for each pass.

25 Suggest one method each for keeping (i) tool indexing time, and (ii) slide movement to a minimum and state why this should be a programming objective.

26 With the aid of a simple sketch describe what is meant by 'tool nose radius compensation' and describe a practical approach that can be employed to determine whether the required compensation is to the right or left of a profile.

27 Describe in detail the methods that can be used to prove a part program.

---

*For additional programming exercises that can be completed by students refer to Appendix C.

9

PART PROGRAMMING CALCULATIONS

Having read the previous chapters the reader should now be aware that the application of CNC technology to the production of machined parts requires considerable knowledge and ability.

In the first instance there is the practical expertise associated with process planning, work-holding, tooling, and so on. In this respect there is continuity between what was expected of the traditional machine shop craftsman and what is required of the machine shop technician involved with new technology. The advent of CNC has had a considerable effect on the way machining tasks are tackled, but the need for practical expertise remains much the same. Furthermore, this expertise can only be obtained by shop floor experience. It is assumed that students involved with CNC part programming will have acquired, or will be in the process of acquiring, this essential knowledge.

Second, the machine shop technician using modern technology is required to become familiar with the programming languages employed by the control units fitted to the machines he or she will be using. In addition, if computer-aided part programming facilities are to be used, then the technician must also become proficient in the application of these techniques. In the last few years those who have been involved in the education and training of others in the use of CNC technology have found that students, in general, rapidly master the use of programming systems. The most advanced aspect of the technology seems to be the one which is most quickly mastered and applied, but unfortunately its application is often marred by a lack of practical expertise and mathematical ability.

It is the mathematical ability of workshop technicians that is the third area of expertise that has to be considered. Mention was made earlier of the need for the part programmer to be able to carry out calculations associated with speeds and feeds, profile intersection points, arc centers, and so on. Competent manual part programming is not possible without a fairly well-developed mathematical ability and a sound understanding of geometric construction.

The mathematics involved in part programming are essentially practical in nature. It is assumed that readers of this text will have already developed these particular skills, and that they will be capable of carrying out the necessary calculations.

Thus it is not the purpose here to provide a text for the student who wishes to learn mathematical concepts, but rather to provide a means of revising certain areas that are of particular interest to the CNC part programmer, and to

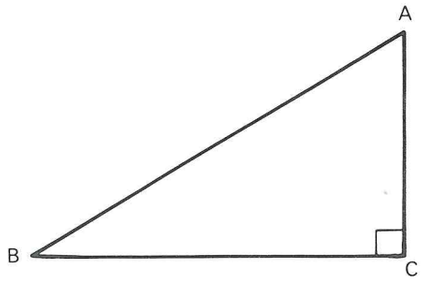
supply the student with examples that will develop the ability to deal with the mathematical elements of programming tasks. Reference to the geometry data contained below may be helpful in some cases.

While the following exercises are intended primarily for manual solution, a number of them will serve to introduce the facilities afforded by computer-aided part programming before the student attempts more comprehensive problems.

GEOMETRY DATA

The following information is provided for reference purposes.

Pythagoras' Theorem



$$AB^2 = AC^2 + BC^2$$

$$AB = \sqrt{AC^2 + BC^2}$$

Similarly,

$$AC = \sqrt{AB^2 - BC^2} \text{ and } BC = \sqrt{AB^2 - AC^2}$$

Example Figure 9.1 shows the detail of a milled component that has been dimensioned without regard to part programming needs. Dimension X is required. Using Pythagoras' theorem, calculate its value.

$$D^2 = 50.84^2 + 63.58^2$$

$$D = \sqrt{50.84^2 + 63.58^2}$$

$$= \sqrt{6627.2}$$

$$= 81.41$$

$$X = 81.41 + 10. = 91.41 \text{ mm}$$

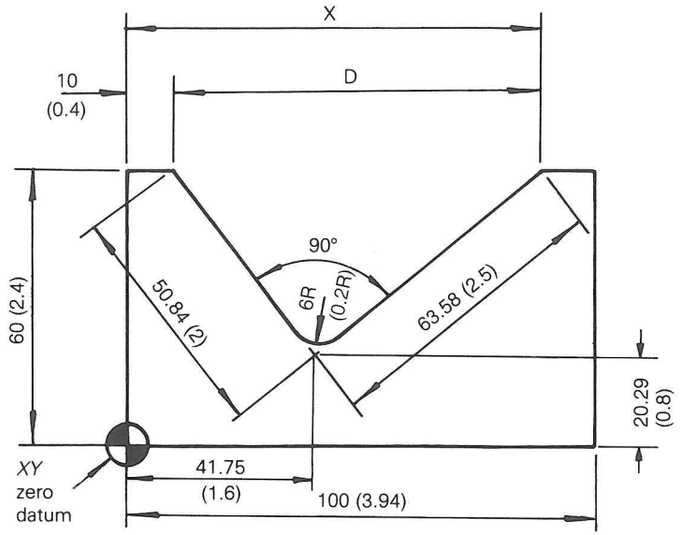
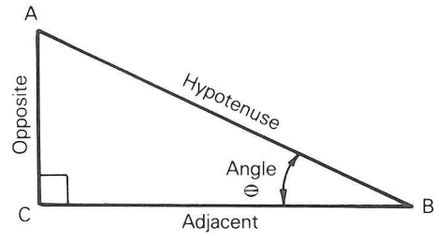


Figure 9.1 *Using Pythagoras' theorem. (Inch units are given in parentheses.)*

Trigonometrical Ratios



$$\text{Sine of the angle } \theta = \frac{\text{Opposite}}{\text{Hypotenuse}} = \frac{AC}{AB}$$

$$\text{Cosine of the angle } \theta = \frac{\text{Adjacent}}{\text{Hypotenuse}} = \frac{BC}{AB}$$

$$\text{Tangent of the angle } \theta = \frac{\text{Opposite}}{\text{Adjacent}} = \frac{AC}{BC}$$

Example Figure 9.2 shows the part detail of a component that is to be programmed with dimensional values being expressed in incremental mode. Thus the target position B has to be defined in relation to the end of the previous move A. Calculate the dimensions X and Y.

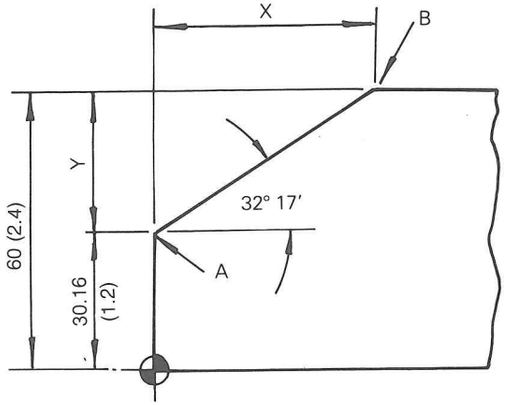


Figure 9.2 *Using trigonometrical ratios. (Inch units are given in parentheses.)*

To calculate Y

$$Y = 60. - 30.16 = 29.84 \text{ mm}$$

To calculate X:

$$\tan 32°17' = \frac{OPP}{ADJ} = \frac{Y}{X}$$

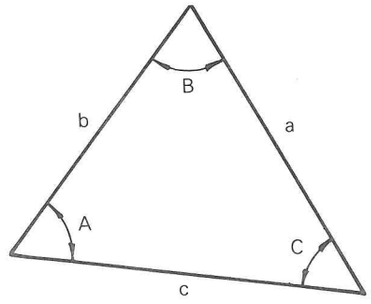
$$X = \frac{Y}{\tan 32°17'}$$

$$= \frac{29.84}{0.6317}$$

$$= 47.23 \text{ mm}$$

The Sine Rule

For use with triangles that are not right-angled.



$$\frac{a}{\sin A} = \frac{b}{\sin B} = \frac{c}{\sin C}$$

Example Figure 9.3 shows a drawing detail of a machined feature. To facilitate part programming, calculate the dimensions X and Y.

To calculate the dimensions X and Y it is first necessary to determine $A\hat{B}D$ and the length AB from the information given. Note that AB = c.

$$A\hat{B}D = B\hat{A}C = 35°$$

$$A\hat{C}B = 180 - (115 + 35) = 30°$$

Using the sine rule to calculate AB:

$$\frac{b}{\sin B} = \frac{c}{\sin C}$$

$$\frac{b \times \sin C}{\sin B} = c$$

$$c = \frac{60. \times \sin 115°}{\sin 30°}$$

$$= \frac{60. \times 0.906}{0.5}$$

$$= 108.72$$

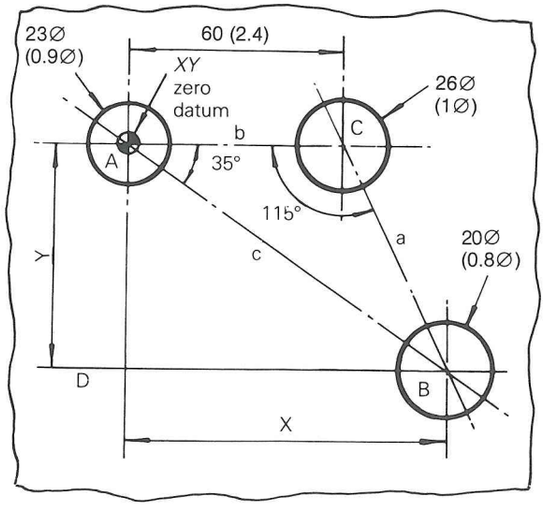


Figure 9.3 *Using the sine rule. (Inch units are given in parentheses.)*

To calculate dimensions X and Y using trigonometry:

$$\cos 35° = \frac{\text{ADJ}}{\text{HYP}} = \frac{\text{X}}{\text{AB}} \qquad \sin 35° = \frac{\text{OPP}}{\text{ADJ}} = \frac{\text{Y}}{\text{AB}}$$

$$\cos 35° \times \text{AB} = \text{X} \qquad \sin 35° \times \text{AB} = \text{Y}$$

$$\text{X} = 0.819 \times 108.72 \qquad \text{Y} = 0.574 \times 108.72$$

$$= 89.042 \text{ mm} \qquad \text{Y} = 62.405 \text{ mm}$$

EXERCISES

1. Figure 9.4 shows details of a turned component dimensioned in a manner that does not accord with the part programmer's wish to locate the prepared billet against the backface of the chuck, this being the *Z* axis zero for the machine to be used, and to use absolute dimensional definition.

   Make a half profile sketch of the component, and dimension it so that the process of preparing the part program will be simplified.

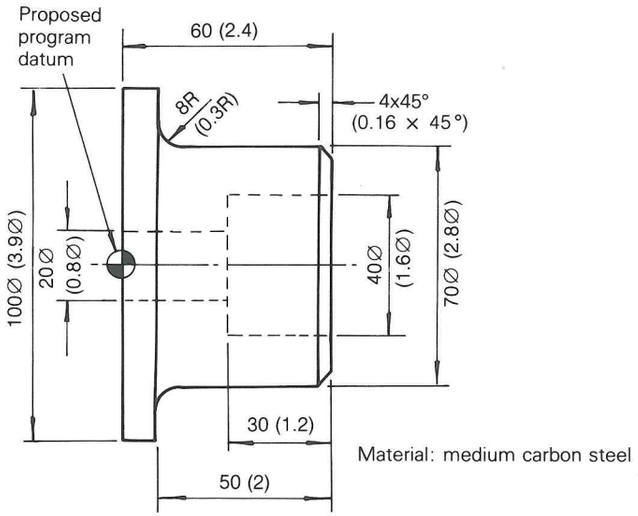


Figure 9.4 *Exercise 1. (Inch units are given in parentheses.)*

2. Figure 9.5 shows details of a component that is to be milled and drilled on a vertical machining center. The programmer has decided to use the corner of the component as the program zero in both the *X* and *Y* axes. Make a sketch of the component and dimension it in a manner

that will be more convenient from a programming point of view, assuming that the programmer intends to use incremental positioning data.

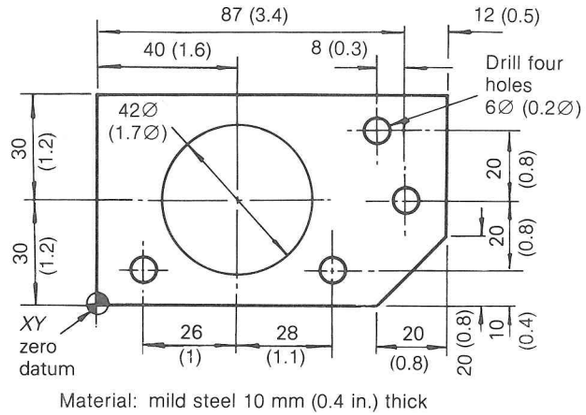


Figure 9.5 *Exercise 2. (Inch units are given in parentheses.)*

3. There is no obvious sequence in which to program the drilling of the six holes in the component shown in Figure 9.6. The programmer has initially chosen to adopt the sequence ABCDEF as indicated on the drawing. Since the time taken to actually drill the holes will be the same whatever sequence is used, the only time saving that can be made is by using the shortest possible positioning route.

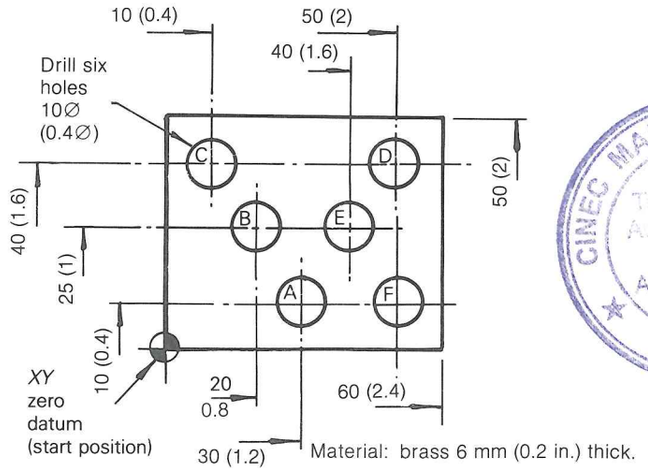


Figure 9.6 *Exercise 3. (Inch units are given in parentheses.)*

Using graph paper, draw the component accurately to scale, and check by measuring whether the proposed route is in fact the most efficient. If it proves not to be the shortest possible, state an alternative.

Given that the rapid feedrate for the machine is 2500 mm/min (100 in./min), estimate the total time saving that could be made by using this different positioning route, during a production run of 5000 components.

4. Figure 9.7 shows a component having a number of drilled holes. Assume that the starting point for the drilling operation is from a clearance plane 4 mm (0.2 in.) above the work surface and immediately above the *XY* zero datum indicated.

   Accurately redraw the component on graph paper, and by scaling the drawing determine the most economical drilling sequence, taking account of the time taken in positioning.

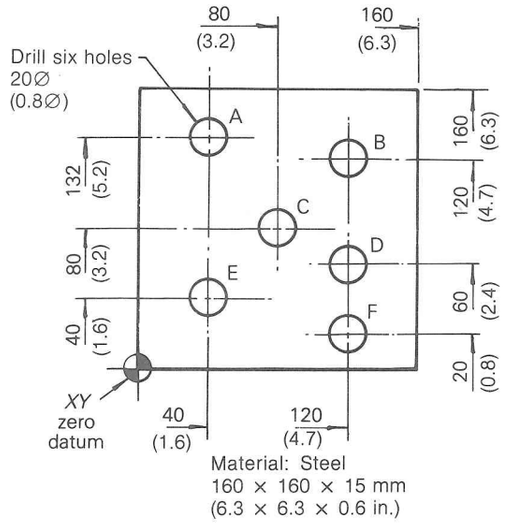


Figure 9.7 *Exercise 4. (Inch units are given in parentheses.)*

5. The milled impression shown in Figure 9.8 is to be machined on a vertical machining center.

   Determine the shortest continuous tool path possible to machine the impression, assuming a start and finish position at zero in the *X* and *Y* axes. (Ignore movement in the *Z* axis as this is likely to be identical regardless of the route chosen in the *X* and *Y* axes.)

   Calculate the time taken to complete the operation given that the rapid traverse rate for the machine is 4000 mm/min (158 in./min),

the feed rate for the metal cutting operation is 0.3 mm/rev (0.01 in./rev), the spindle speed is 3000 rev/min and the *Z* up position is 200 mm (8 in.) from the *Z* axis zero which is set at the top face of the work. Assume the tool change position is immediately above the *XY* axes zero. Normal *Z* axis clearance is 0.1 inch or 2 mm above the part face.

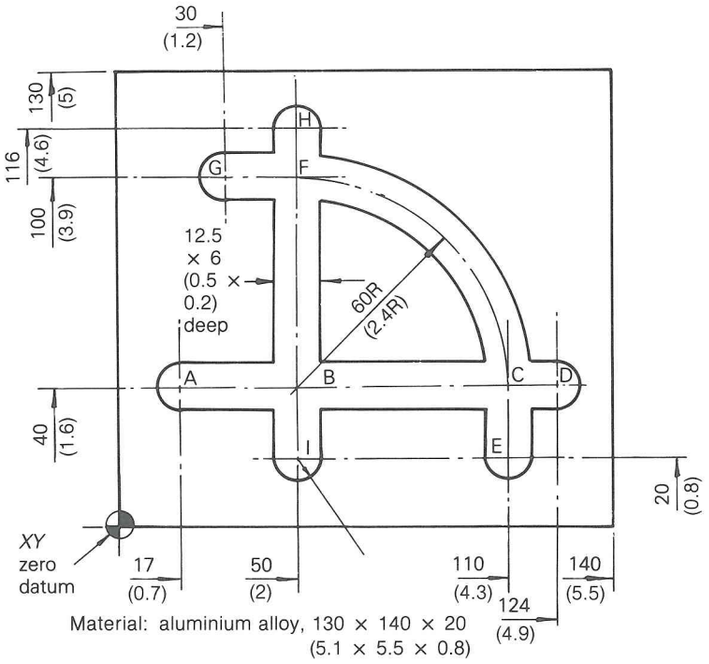


Figure 9.8 *Exercise 5. (Inch units are given in parentheses.)*

6. Assume that the feature shown in Figure 9.9, the machining of which involves a certain amount of stock removal prior to a finishing pass along the profile, is to be produced on a machine with a control system which does not possess a suitable canned cycle. Thus the programmer has no alternative but to determine, manually, the most efficient way of clearing the step. He or she may choose to program a series of cuts, with a small amount of cutter overlap to ensure a clean face, with the bulk of the stock being removed as the cutter travels along the *X* axis or, alternatively, along the *Y* axis.

   Reproduce the pocket accurately to scale on graph paper and determine which, if any, of the two cutting directions would remove the

stock in the shortest time. Assume a cutter diameter of 12 mm (0.5 in.).

Express the movements required to remove the stock in terms of X and Y values, which could be incorporated as data in a part program.

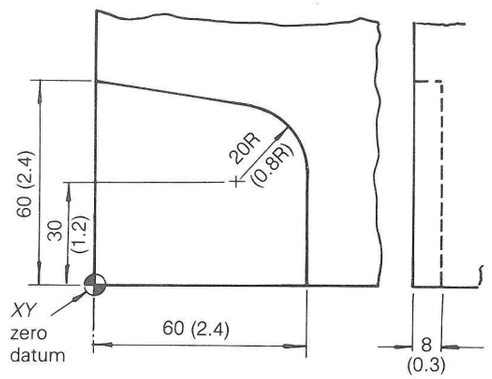


Figure 9.9 *Exercise 6. (Inch units are given in parentheses.)*

7. On a piece of graph paper, accurately reproduce the component shown in Figure 9.10 to a scale of twice full size and indicate on the drawing a series of roughing cuts, each having a depth of 4 mm (0.2 in.), that would reduce the bar sufficiently to leave approximately 1 mm (0.03 in.) along the profile for the final finishing cut.

   Express the roughing cuts in terms of X and Z, that would enable them to be incorporated in the part program.

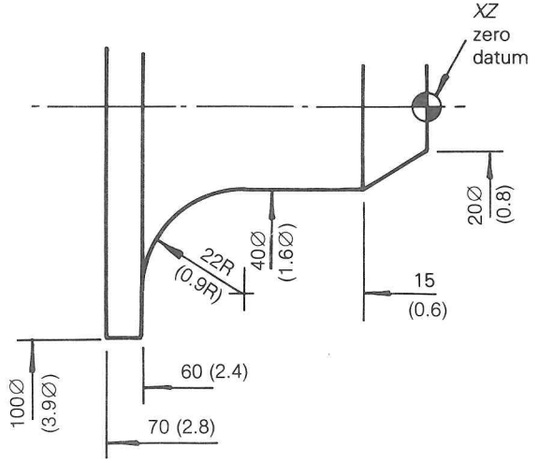


Figure 9.10 *Exercise 7. (Inch units are given in parentheses.)*

8. The angular feature of the component shown in Figure 9.11 is to be machined on a CNC-controlled machine, the control system of which does not include the cutter radius compensation facility. Calculate the incremental linear moves that will need to be included in the part program, assuming movement starts from and returns to the indicated datum. The diameter of the cutter to be used is 50 mm (2 in.), and there is to be approach and overrun distances of 4 mm (0.2 in.) as indicated.

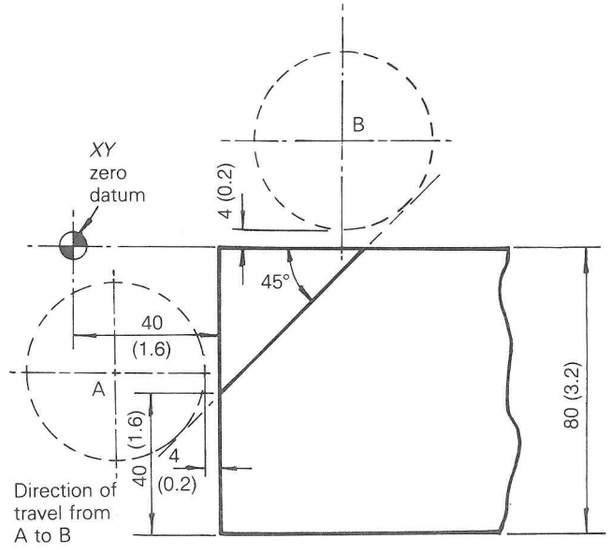


Figure 9.11 *Exercise 8. (Inch units are given in parentheses.)*

9. Calculate the incremental linear movements in the X and Z axes that will be required to finish machine the taper shown in Figure 9.12.

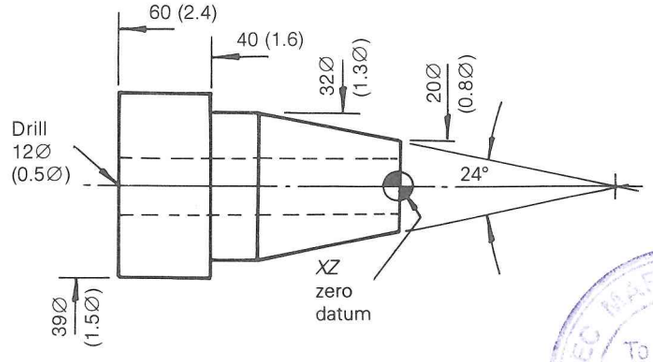


Figure 9.12 *Exercise 9. (Inch units are given in parentheses.)*

10. Figure 9.13 shows a feature of a turned component. Two of the dimensions necessary to complete a part program are not given.

    Determine which dimensions are missing and calculate their values.

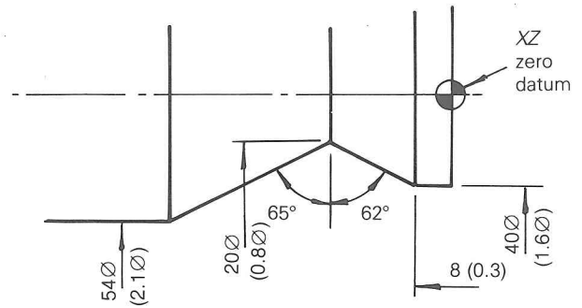


Figure 9.13 *Exercise 10. (Inch units are given in parentheses.)*

11. Figure 9.14 shows a feature of a component, namely, two parallel slots positioned at an angle of 52° to the *X* axis. The programmer intends to specify the numerical data for the required slide movements in absolute terms from the *XY* zero datum indicated on the drawing. To do this a number of calculations will have to be made and the drawing must be redimensioned.

    Carry out the necessary calculations and produce a second drawing of the component dimensioned in a manner that will be more appropriate.

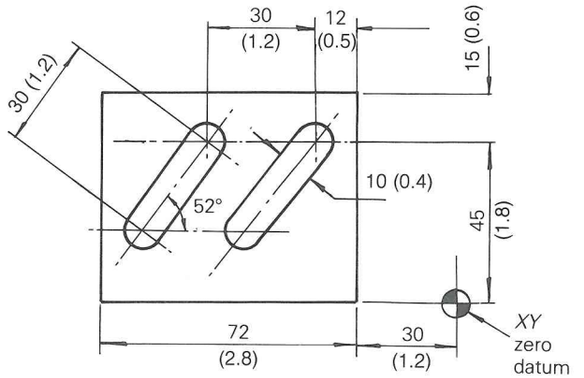


Figure 9.14 *Exercise 11. (Inch units are given in parentheses.)*

12. The component shown in Figure 9.15 is to have two holes drilled in the positions indicated. The programmer has opted to program slide

movements in absolute mode from the *XY* zero datum. In order to program in this way it will be necessary to calculate the linear coordinates for one of the holes.

Sketch the component, carry out the required calculations, and dimension your drawing accordingly.

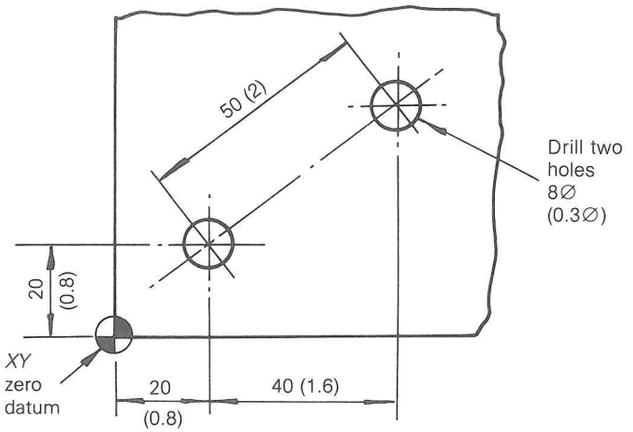


Figure 9.15 *Exercise 12. (Inch units are given in parentheses.)*

13. Calculate and list in programmable form the absolute coordinate dimensions in the *XY* axes necessary to drill the four holes shown in Figure 9.16 in the sequence ABCD, starting and returning to the program datum. Assume that a drill cycle controlling the depth to be drilled is already operative, and may be cancelled by programming G80.

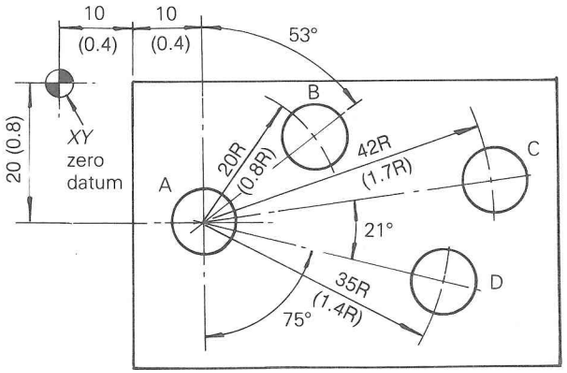


Figure 9.16 *Exercise 13. (Inch units are given in parentheses.)*

14. Figure 9.17(a) shows the position of three holes dimensioned in such a way that is not particularly helpful to the person preparing a part program, since the facility to program polar coordinates is not available. The preferred method of dimensioning is indicated in Figure 9.17(b). From the information provided, calculate the missing dimensions.

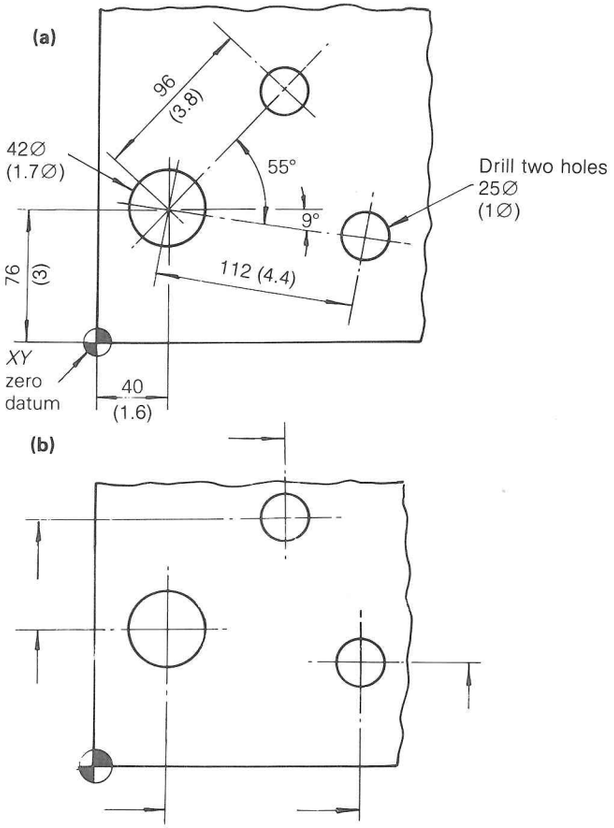


Figure 9.17 *Exercise 14. (Inch units are given in parentheses.)*

15. The four holes shown in Figure 9.18 are to be drilled in the sequence ABCD with the necessary slide movements expressed in absolute terms. Assuming a *Z* up travel of 200 mm (8 in.) immediately above the program datum to accommodate a manual tool change, and that a G81 drill cycle (cancelled by a programmed G80) is available, carry out the following:

    From the data stated on the drawing determine the dimensional value of the required slide movements.

List the data, in word address format, that would be required to complete the machining, assuming that all speeds, feeds etc. are already operative.

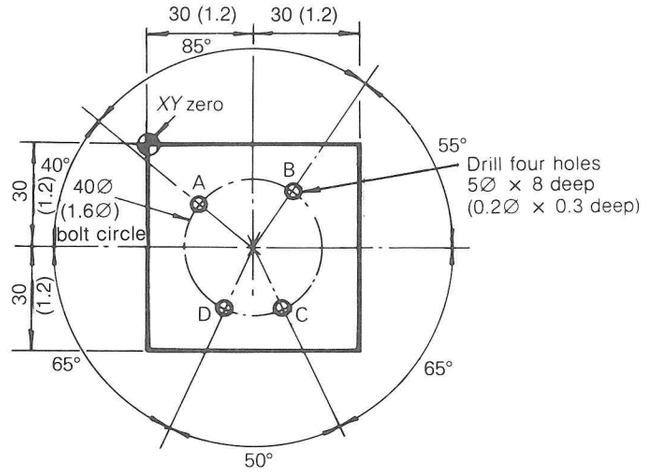


Figure 9.18 *Exercise 15. (Inch units are given in parentheses.)*

16. Eight holes are to be drilled on a bolt circle diameter as shown in Figure 9.19. The holes are to be drilled in the sequence A to H immediately after the central hole has been drilled. Taking the position of the central hole as the zero datum in both the *X* and *Y* axes carry out the following:

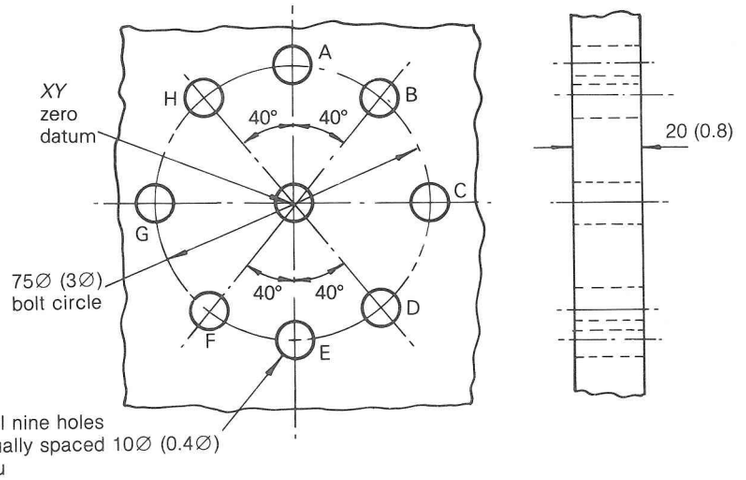


Figure 9.19 *Exercise 16. (Inch units are given in parentheses.)*

Calculate the dimensional value of the required slide movements, to facilitate programming in incremental mode.

List the data as they would be presented in a word address program, assuming a *Z* datum clearance of 2 mm (0.1 in.) and an excess travel of 5 mm (0.2 in.) on breakthrough. Assume that a spindle speed and feed rate have already been programmed and that a G81 drilling cycle is to be used.

17. The positions of three holes are given on a drawing as shown in Figure 9.20. The holes are to be drilled in the sequence ABC and the previous program data have brought the machine spindle into vertical alignment with hole A. Calculate the linear values of the incremental moves to be included in the part program to control slide movement in the *X* and *Y* axes.

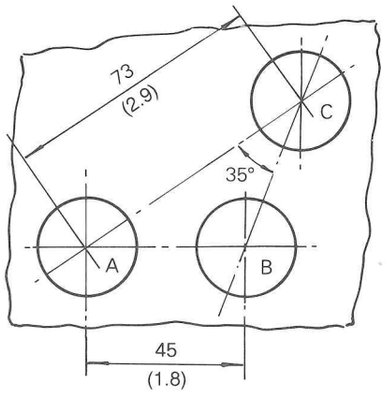


Figure 9.20 *Exercise 17. (Inch units are given in parentheses.)*

18. Complete the program data given below (for metric values only) for achieving relative cutter movement from P1 to P2 on the profile illustrated in Figure 9.21. The circular arcs are to be programmed by defining the target positions using *X* and *Y* values, and the arc centers are defined in relation to the starting point of the arc using *I* and *J* values.

N60 G01 *Y*-25
N70 G03
N80 G01 *X*20
N90
N100 G02
N110 G01 *X*10
N120
N130 G03
N140 G01

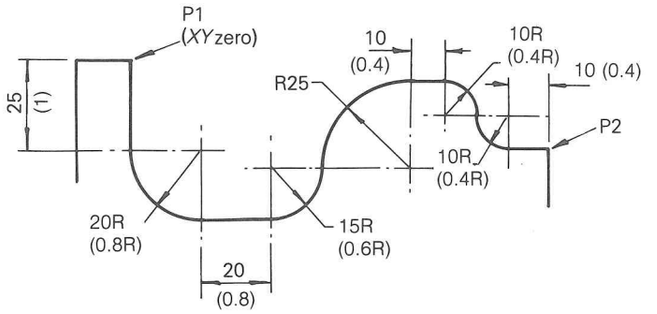


Figure 9.21 *Exercise 18. (Inch units are given in parentheses.)*

19. The turned component shown in Figure 9.22 is to be produced on a turning center where arc centers are defined in relation to the program datum using the address letters I and K. Complete the program block below in incremental terms (in metric units only):

N 025 G02 X-- Z-- I-- K--

Repeat the above exercise with the data expressed in absolute terms.

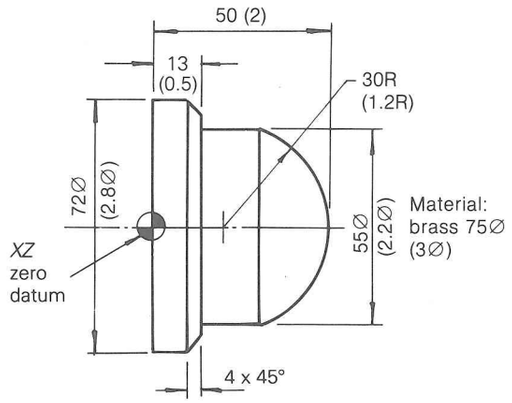


Figure 9.22 *Exercise 19. (Inch units are given in parentheses.)*

20. The concave arc of the turned component shown in Figure 9.23 is to be produced on a lathe where the control system will simply require a data input stating the radius of the arc, the direction of rotation, and the target position defined by *X* and *Z* values. Assume that the 8 mm (0.3 in.) diameter hole has already been drilled, and that the curve is to be produced by making a cutting pass starting from the center of the component and working outward. Complete the data necessary to

machine the feature, with the tool starting and finishing in the positions indicated. Assume that a suitable cutting speed and feed rate, and also cutter radius compensation, are already active.

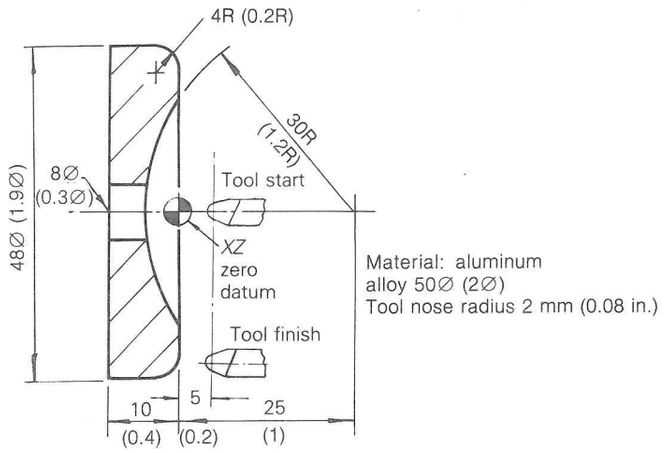


Figure 9.23 *Exercise 20. (Inch units are given in parentheses.)*

21. A cutout of 30 mm (1.2 in.) radius is to be machined in a 40 mm (1.6 in.) diameter disk as shown in Figure 9.24. The cutter start and finish positions giving a suitable approach and runout are indicated. Assume that cutter radius compensation mode will be operative and that the circular interpolation will involve two separate data blocks, one for each quadrant. Determine the two data blocks that will be required, defining the arc start point in relation to the center using I and J values. Express the positional data in absolute terms.

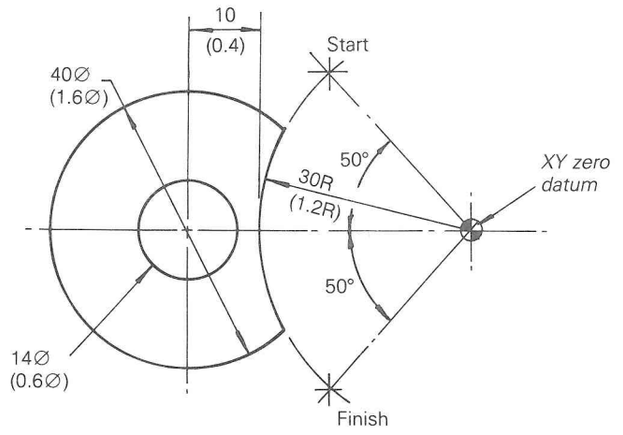


Figure 9.24 *Exercise 21. (Inch units are given in parentheses.)*

22. Figure 9.25 shows a radial slot that is to be machined using a relative cutter movement from the start point to the finish point as indicated.

    Calculate the target position in relation to the indicated program datum.

    If the start position is to be defined in relation to the arc center using I and J definition, state their numerical values.

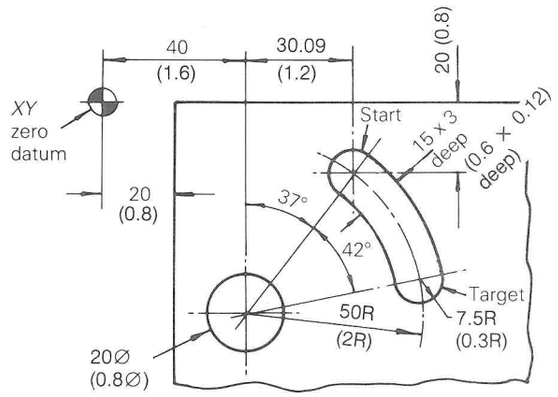


Figure 9.25 *Exercise 22. (Inch units are given in parentheses.)*

23. The two radial slots shown in Figure 9.26 are to be produced on a vertical machining center, with the starting points as indicated and the relative cutter movement being in a clockwise direction. The control system requires arc centers to be defined in relation to the arc starting point.

    Determine the additional dimensional data, in absolute terms, that will be necessary for programming purposes in order to machine the slots.

24. The turned component in Figure 9.27 is to be produced on a machine fitted with a control system that requires the circular interpolation data entries to be programmed as follows:

    (a) The target position using *X* and *Z* numerical values.

    (b) The arc center in relation to the program datum using *I* and *K* numerical values.

    Assuming that the program is to be compiled using absolute dimensions, complete the following N125 G02 X-- Z-- I-- K-- program entry for machining the arc.

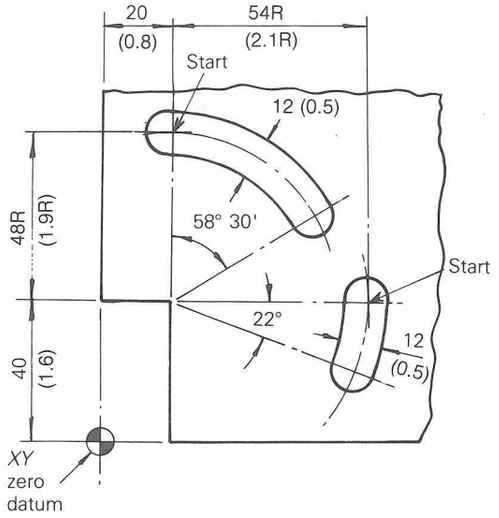


Figure 9.26 *Exercise 23. (Inch units are given in parentheses.)*

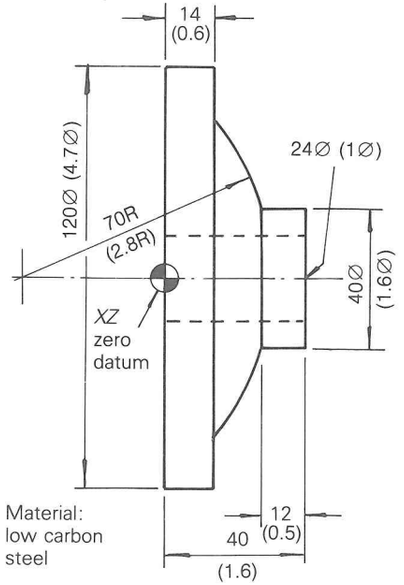


Figure 9.27 *Exercise 24. (Inch units are given in parentheses.)*

25. Figure 9.28(a) gives the details of part of a turned component. Figure 9.28(b) shows the same feature but with an indication of the dimensioning method that would be more appropriate since the machine to be used does not cater to multiquadrant programming.

    Calculate the alternative dimensions required and make a redimensioned sketch of the component feature.

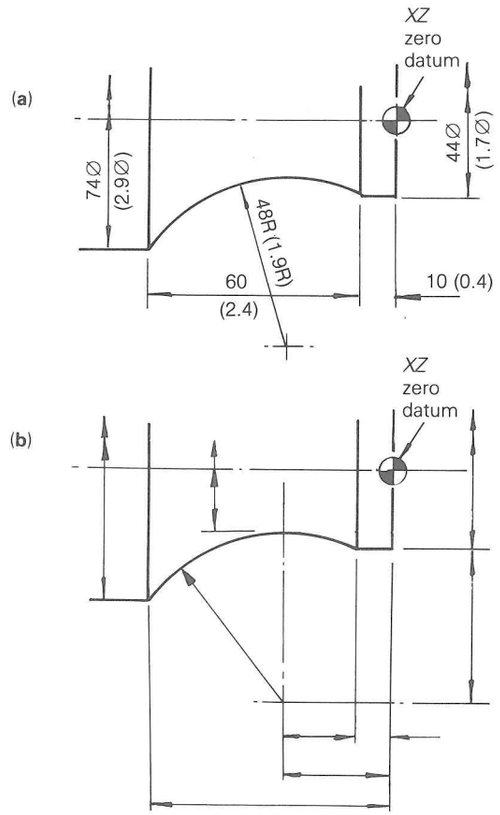


Figure 9.28 *Exercise 25. (Inch units are given in parentheses.)*

26. Figure 9.29 shows part of the profile of a milled component. The profile consists of four blending arcs of varying dimensions. In order to program the necessary circular moves the programmer will need to know the exact points at which the curves intersect with each other.

    Construct accurately on graph paper the given profile and locate on your drawing the points of intersection.

    Calculate and define numerically each intersection point.

Given the following programming information, list the program data necessary to achieve a cutter path that would machine the profile, commencing with linear movement from the *XY* zero datum and ending at point B.
Programming information:

G01 Linear interpolation, programmed feed rate.
G02 Circular interpolation, clockwise.
G03 Circular interpolation, counterclockwise.

Define all target positions incrementally and the arc centers in relation to the arc starting points using I and J values. Assume spindle control data, feed rates, etc., have already been programmed and that cutter radius compensation is active.

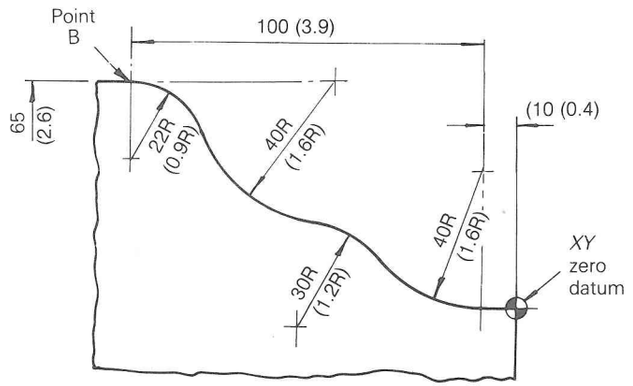


Figure 9.29 *Exercise 26. (Inch units are given in parentheses.)*

27. The milled profile shown in Figure 9.30 is to be produced on a machine with a control system that requires numerical definition of the target positions of all slide movements using the address letters X and Y, and arc centers to be defined in relation to the start of the arc using address letters I and J. Also, the system is not capable of multiquadrant programming, so movement in each quadrant must be programmed separately even when the same radius passes from one quadrant into a second.

    Accurately construct the profile and indicate in absolute terms in relation to the program zero all intersection points and also arc start positions in relation to the arc centers.

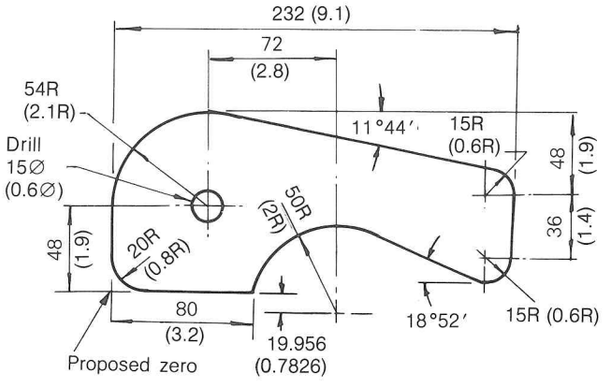


Figure 9.30 *Exercise 27. (Inch units are given in parentheses.)*

28. The internal profile of the component shown in Figure 9.31 is to be produced on a vertical machining center. Calculate and indicate on an appropriate sketch the additional dimensions that will be required to define the profile intersection points.

    State which common programming facility would be suited to facilitate machining the profile.

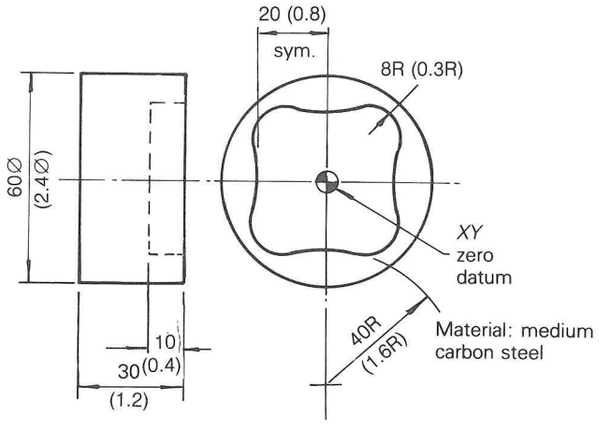


Figure 9.31 *Exercise 28. (Inch units are given in parentheses.)*

29. Figure 9.32 shows details of a milled component.

    Draw the raised profile accurately to size and indicate on your drawing the arc centers and the intersecting points of the profile.

Calculate all the profile intersecting points required to complete a part program and dimension your drawing accordingly. Assume that arcs can only be programmed for one quadrant in any one block, with the arc centers being specified in relation to the program zero using I and J values.

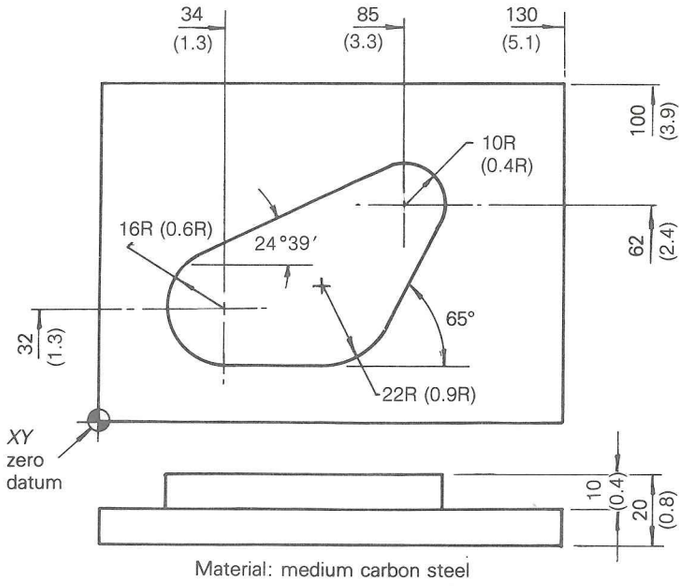


Figure 9.32 *Exercise 29. (Inch units are given in parentheses.)*

Refer to the cutting speeds and feeds given in the Appendices for appropriate data to complete the following questions.

30. Calculate a suitable spindle speed for drilling 6 mm (0.2 in.) diameter holes in a medium carbon steel using a high speed steel drill.
31. Determine a suitable spindle speed for face cutting T6 aluminum using a shell end mill of 40 mm (1.6 in.) diameter with cemented carbide insert teeth.
32. What increase in spindle speed would be appropriate if, when drilling a low carbon steel, a 20 mm (0.8 in.) diameter high speed steel drill is replaced with another having cemented carbide brazed tips?
33. Select a suitable cutting speed for turning a complex profile from brass using a cemented carbide insert turning tool. Given that the component is of variable diameter along its length, why would it be preferable to program a constant surface cutting speed rather than a set spindle speed?

34. Calculate the spindle speed to be used when milling grey cast iron using a 20 mm (0.8 in.) diameter cemented carbide insert end mill. If the speed calculated eventually proved to be too high, what action should be taken by the machine operator to rectify the situation?
35. Calculate suitable spindle speeds for roughing and finishing cuts when turning T6 aluminum, using cemented carbide tooling on a component having a nominal diameter of 38 mm (1.5 in.).
36. Select a suitable feedrate in mm/rev for turning medium carbon steel, using cemented carbide insert tooling.
37. Calculate a suitable feedrate in mm/min or in./min for a light turning operation on a stainless steel component having a diameter of 75 mm (3 in.), and when using cemented carbide tooling.
38. Given that an appropriate feedrate per tooth for face milling a low carbon steel is 0.3 mm (0.01 in.) per tooth when using cemented carbide tooling, what would be a suitable program entry in mm/min or in./min when using a 75 mm (3 in.) diameter cutter having six teeth.
39. The profile shown in Figure 9.33 is of part of a component that is to be machined from brass. The feed rate is to be programmed in mm/rev (in./rev) and the spindle speed controlled by programming a constant surface cutting speed in m/min (ft/min).

    Select appropriate values for cutting speed and feed to be included in the part program, when using cemented carbide insert tooling.

    Using the above combination of data, would the resulting surface finish be identical for both the parallel surfaces and the taper? If not, and assuming that a variation is not acceptable, how could the program be modified?

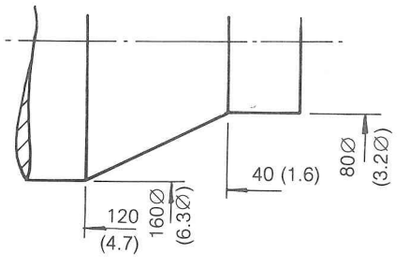


Figure 9.33 *Exercise 39. (Inch units are given in parentheses.)*

40. A face milling operation on aluminum alloy is to be carried out using a nine-cartridge cutter of 100 mm (4 in.) diameter and having cemented carbide inserts. Determine a suitable spindle speed and feed rate in mm/rev (in./rev) for inclusion in the part program.

10

COMPUTER-AIDED PART PROGRAMMING

THE APPLICATION AND ADVANTAGES OF COMPUTER-AIDED PART PROGRAMMING

When NC, as opposed to CNC, was first introduced, the only way a curve could be produced was by approximation, making a series of angular moves using slide movements in two axes that, when blended together, approximated to the curve required. The larger the number of small angular movements made, the more precise the final curve.

On the face of it this would appear to be a very reasonable solution to the problem of producing a curve until the work involved in making the necessary calculations is considered, not to mention the mathematical ability required. It was the sort of situation where a little computer help with the calculations was much appreciated. The more complex the profile—imagine an elliptical path, for instance—the more essential computer help became.

Today, thanks to the inclusion of a microcomputer as an integral part of even the most basic CNC control systems, the programming of a constant radius curve is a very simple matter indeed, often requiring nothing more than dimensional definition of the target position and the value of the radius. Even the more complex elliptical profiles can be programmed on some control systems simply by defining the major and minor axes.

The reader will recall that the programming of radial cutter paths is referred to as "circular interpolation," while the facility used to program an elliptical profile is known as a special canned or program cycle.

There is a wide variety of canned cycles currently available with modern machine controllers. A number of these were described in Chapter 8. All of these canned cycles were designed and included in the control system with one objective in mind, that is, to simplify programming.

Canned cycles cater to sequences that are likely to recur regularly. But not all complex profiles regularly recur and when such a profile does occur, it can present problems as difficult, and possibly more so, as the problems associated with curves in the early days of NC.

In particular, the machining of complex profiles, or rather the preparation of the part program to achieve the machining, means that fairly complex calculations have to be performed to determine the geometry intersection points.

There is also the problem of determining efficient cutter paths to remove stock, and that can also be a laborious business.

It is to meet these requirements that special computer-based programming systems have been developed. The process of using these systems is referred to as Computer-Aided Part Programming, generally referred to as CAPP. (Note: the initials CAPP are also used in production engineering to denote Computer-Aided Process Planning, which is concerned with the total organization of a production operation of which Computer-Aided Part Programming may be an included activity.)

CAPP provides for a simpler, quicker, and more accurate approach to preparing a CNC part program. But at the same time it represents a considerable capital outlay that has to be justified by an eventual increase in efficiency and productivity.

The CAPP process involves preparing a program using a specially developed language, entering the resulting data into a computer and receiving back from the computer a program presented in a format acceptable to the machine controller. During the process the computer will have processed the data to verify their validity and, where necessary, performed computing tasks (many that would have been mathematically difficult and/or time-consuming if attempted manually), and then processed the results into machine language.

CAPP can, in the hands of experienced users, provide rapid programming solutions for the most difficult of work. Even for very simple work, where manually prepared programs could be produced fairly quickly, the use of CAPP is still a viable proposition.

A trend among engineering contractors is to ask the potential customer for a drawing of the component that is to be produced and then to program, using CAPP, and then manufacture a component. The result is then returned to the customer with the quotation, indicating very effectively the speed and quality of the service they offer. At the same time the contractor is able to cost the contract precisely, since it will be known exactly how long it took to machine the sample. This process is normally referred to as a customer bench mark.

A further advantage of CAPP is that the program is prepared and initially proven "off-line," that is, away from the machine. Data transfer electronically to the machine is available and rapid so there is no delay in getting a machine back into production following job changes.

COMPUTER INSTALLATIONS

CAPP systems are available for use on all types of computer installations from mainframe to micro.

The large mainframe computers possess the greatest computing power and their use is indispensable for very complex programming requirements. Unfortunately they are extremely costly and their installation is only economically

viable in large organizations where the computing needs—not only for CAPP but for a whole range of industrial and commercial activities—are considerable.

A mainframe computer will cater to a large number of work-stations or terminals that, provided the distance between the two is not excessive, can be permanently cable-linked. Where the distance is considerable, they can be linked via telephone modem devices. This also makes it possible to cross international frontiers and even link continents.

Small industrial and commercial organizations can gain access to a mainframe computer, and the required software, on a "time-share" basis. The computer, which may be many miles distant, is accessed via the public telephone network. The facility is available to numerous subscribers and each pays for the actual computing time used; but to this cost it is necessary to add the normal telephone charges which can be considerable.

Subscription to such a system also provides access to a range of back-up services such as the use of the latest software and professional advice regarding its application. Help is also available to solve problems encountered when using the system generally.

In addition, time-share subscribers do not have to concern themselves with maintenance or servicing of the system, as is the case with an in-house installation when a service contract with the manufacturer or supplier would be another costly but essential requirement.

One drawback to time-sharing, assuming that the financial considerations are acceptable, is that access to the computer may not always be conveniently available because too many subscribers are trying to "log on," or connect into the system, at any one time. Another is that the data security may be inadequate if the work concerned is of a sensitive nature.

Many organizations have requirements that do not justify the installation of a mainframe, but at the same time could not be satisfactorily allowed for by time-sharing. They often install their own large to medium-sized systems. The available capacity and computing power will be less extensive than that available from a mainframe computer, but still capable of servicing a very large organization.

A feature of these installations is the permanent linking of work stations, or terminals, to the host computer. When terminals are linked in this way, they are said to be "networked." By this means the complete system may become fully interactive, making it possible for data to be originated and accessed by a number of users. It may be possible for a programmer to utilize data originated by a designer, thus providing a link between design and manufacture, referred to as CAD/CAM (Computer-Aided Design/Computer-Aided Manufacturing), which will be presented subsequently. There may also be direct connections to the machine tools on the shop floor, a facility referred to as Direct Numerical Control (DNC).

A variation of this approach is to transfer the completed part program to a temporary data storage facility on the shop floor, which is situated alongside and connected to the machine that is to manufacture the part. The production controller can download the data into the machine control unit as and when required. Programs can also be downloaded from the machine into the storage unit. The computer systems can also be configured to deliver program data to tape punch, cassette tape, or floppy disk units.

The networking of user terminals described above is also available on even smaller computer installations, but the number of terminals and, of course, the computing power available is proportionally reduced. However, a "mini" or "supermini" computer, small enough to fit under a desk, is still capable of handling a comprehensive 3D CAD/CAM system, providing control of both the design and manufacturing elements of engineering and often including other functions such as costing, invoicing, etc.

Finally, there are the installations involving "micro" or "supermicro" computers. Stand-alone CAPP systems that operate on microcomputers are now capable of handling very complex programming requirements and are widely used. This type of installation is relatively cheap, which makes their installation by smaller companies a feasible proposition.

Micro-based systems can be networked and linked to peripheral equipment such as plotters, printers, and tape punches. Provided the distance is not too great, they can also be cable-linked to machine tools.

Because of restrictions on expenditure, most educational establishments have found it necessary to install microcomputer-based CAPP systems, and it is this type of installation which students are most likely to use when first being introduced to computer-aided part programming.

Figure 10.1 shows a general arrangement of a CAPP work station utilizing a microcomputer.

It is possible that a company may not become involved in any of the arrangements outlined above, since there are many consulting offices that offer a program preparation and proving service. The use of a consultant can sometimes be attractive to companies that do not have sufficient programming work to sustain a part programmer working full time, and which may prefer to retain a lower, and therefore cheaper, level of skill on the shop floor. Even when the skill levels on the shop floor are such that part programming could satisfactorily be undertaken, the use of a consultant, providing an already proved program, means that no time is lost between ending one production run and beginning the next.

The staff of programming consultant offices will work in close cooperation with the client company, so that the machining is processed in a way acceptable to all concerned. In some cases they also offer supporting services associated with the selection of tooling and the design of special work-holding arrangements if required.

Figure 10.1 *General arrangement of a CAPP work station.*

HARDWARE CONFIGURATIONS

Whatever the computer installation used, it is possible to establish, in a general way, the hardware configurations associated with CAPP.

The diagram 10.2 shows a very basic computer-aided part programming system: a computer, a data recording facility in the form of a tape punch, a data transfer facility in the form of a tape reader attached to the machine tool, and the machine tool itself.

In Figure 10.3 this basic arrangement is extended and includes, instead of the tape punch and tape reader, a direct cable link to the machine tool, which

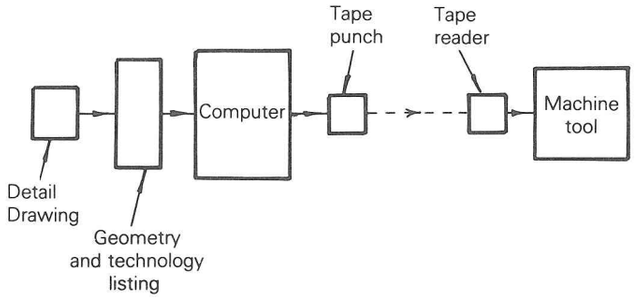


Figure 10.2 *Basic CAPP system.*

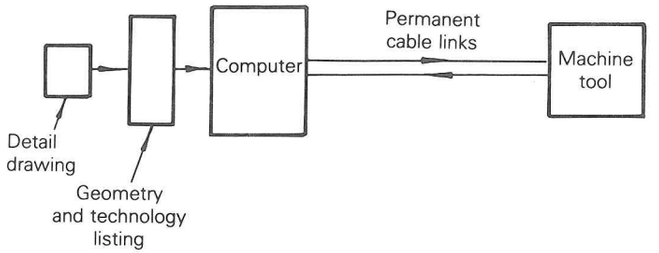


Figure 10.3 *Simple direct numerical control.*

is a much quicker and more convenient method of transferring data. When the computer and machine are linked by cable, the arrangement is referred to as Direct Numerical Control (DNC).

Figure 10.4 shows the concept extended still further, this time to include cable links to a number of machine tools which may be of different types. These machines may be arranged in a particular way, possibly associated with robot work-handling devices, to form a machining cell.

Taking an even broader view, there may even be a number of programming stations and a number of machines, or machining cells, all linked to the same computer. In this situation the part programming function could be but a small element of a totally integrated computer-controlled manufacturing environment including design, marketing, accounting, materials handling, personnel control, and so on. Complex systems such as this are referred to as Computer Integrated Manufacturing, CIM. Of course, the greater the demands on the system the more extensive will be the required computing and data storage facilities.

From the foregoing the reader will appreciate that it is possible for the CAPP system with which he or she is concerned to be a relatively simple stand-alone

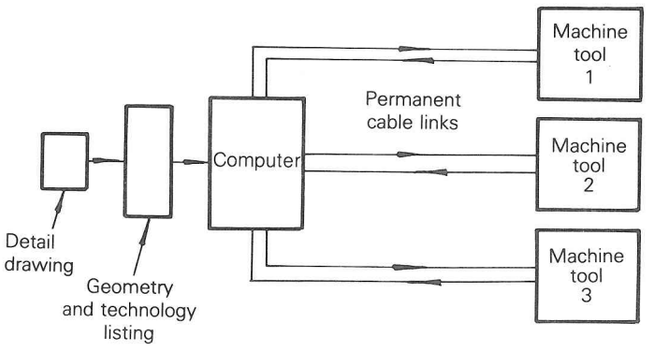


Figure 10.4 *Direct numerical control of a machine group.*

system, the purpose of which is solely to produce CNC part-programs for a particular machine or a limited number of machines, or it may be an integral part of a much more complex installation.

Even if the overall computing arrangement is complex, it is still possible to consider the CAPP element in isolation and return to the basic objective: preparing a part program with computer assistance and then transferring the resulting data to the machine tool. The main elements in this process are shown in Figure 10.5. Also indicated is the range of peripheral equipment that can be used to support the activity.

The CAPP system available for use by the reader may include all or only some of the items indicated. Systems are structured according to the funds available at the time of purchase, and limited finance can have a restricting effect.

However, it is essential that a part programmer is fully conversant with any system he or she is going to use. Time spent in getting to understand the system

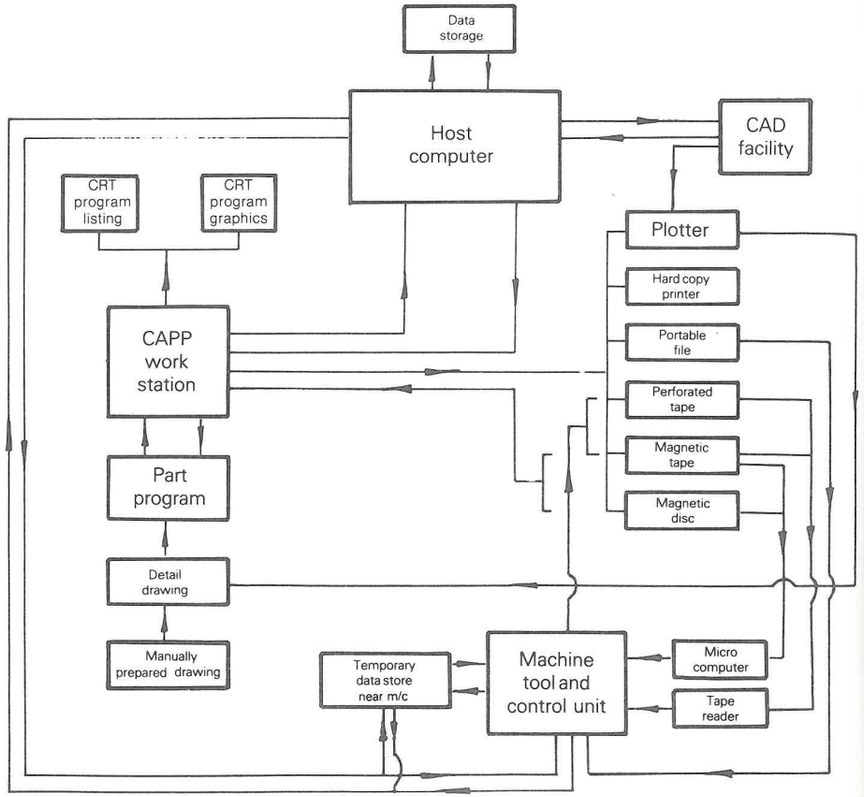


Figure 10.5 *CAPP station incorporated in a system.*

before making any attempt to prepare a part program is time well spent, since it can do much to eliminate the time-consuming and often frustrating need to ask for help every time an unfamiliar feature of the system is encountered.

INPUT AND CONTROL DEVICES

There are a number of ways by which data can be entered into a computer during the CAPP process.

Input via the familiar alphanumeric keyboard is laborious and rather slow, bearing in mind that the average programmer is likely to be a little lacking in keyboard skills. But since many of the data entries associated with CAPP are repetitive, it is possible to speed up the process by using various supplementary devices and techniques.

Selection from a menu, that is, a list of options, is one such facility. The menu may be included as an overlay on a digitizing tablet or be displayed on the CRT screen.

A digitizing tablet (Figure 10.6a) is a device like a small rectangular board

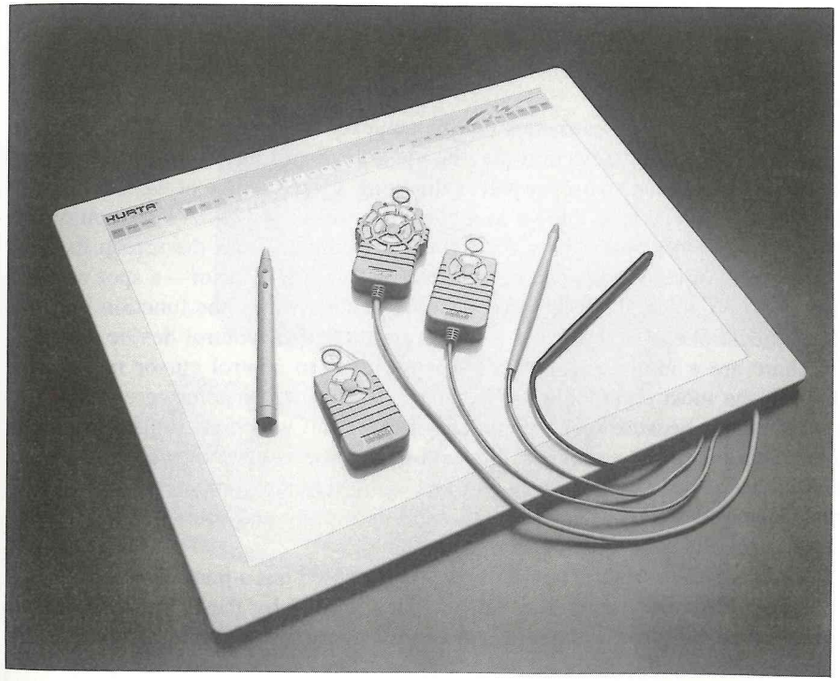


Figure 10.6a *Digitizing tablet. (Photograph of IS/ONE courtesy of Kurta Corp., 3007 E. Chambers, Phoenix, AZ 85040.).*

that is positioned alongside the computer keyboard and is cable-linked to the computer. The tablet is capable of detecting the position of a puck, light pen, or stylus when any of these devices is placed on it.

An overlay is like a plan that divides the surface of the tablet into a number of small areas. Each area is allocated to a specific function such as representing an item from the menu.

A puck is a device with cross-hairs mounted in a small block. It is traversed by hand over the tablet until the cross-hairs are located over the function to be selected. The selection is then confirmed either by an appropriate keystroke or by pressing a button on the puck. A light pen or stylus is rather like a pen. It is used to identify the menu item required and the choice is then activated by pushing a button or applying slight pressure to the tablet or CRT screen position.

A digitizing tablet is a feature of the programming station illustrated in Figure 10.1, which also shows the puck used for menu selection.

Screen menus can either occupy a complete CRT screen, in which case the graphic image which is an important feature of the CAPP process is temporarily lost, or it can occupy part of the screen so that the graphic image is retained. On small CRTs, the second arrangement can mean that the graphic display is rather cramped. A hardware configuration that will eliminate the disadvantages inherent in both arrangements is to have a two-screen display, one for the menu and program listing and one for the graphic display, but this does add to the cost of the installation.

Selection from screen menus can be achieved in several ways. If the menu items are numbered, selection may be via a keyboard input. A second method is to use a light pen, which involves directing a light source at the CRT screen that is of a special type known as a "vector refresh" screen. A variation of this approach is a pen that senses the light being emitted from the screen itself.

Third, the menu items can be selected by moving a cursor—a spot or cross that can be moved about the screen—and then activating the function by making a keystroke or by pressing a button on the cursor control device.

There are a number of devices that are used to control cursor movement. One of the most commonly used is the "mouse." It has some resemblance to a real mouse because of its shape and its long tail which is, in fact, the cable connecting it to the computer. Beneath the mouse is a set of wheels/balls; as the mouse is moved about a flat surface alongside the computer—a table top, for instance—the wheels/balls detect the movement and cause the screen cursor to make a corresponding movement.

Another device is the "tracker ball." This device has a partially exposed ball mounted in a small box. The ball is rolled around by the palm of the hand. The movement of the ball is detected and, as with the mouse, a corresponding cursor movement appears on the screen.

A third method of cursor control is by the use of a "joystick." As this is moved around in all directions, the screen cursor moves in unison. A joystick is shown in the illustration of the CAPP work station in Figure 10.6b.

It is possible that an area of a digitizing tablet can be allocated to represent the CRT screen. The stylus or the puck referred to earlier can then be used to pass over the screen area and effect a corresponding movement on the screen.

Apart from menu selection, a cursor is also used within the CAPP process to identify geometric elements such as points, lines and circles that have been constructed on the screen. For example, to construct a line at 90° to a base line will first need a menu selection to identify the type of construction required, followed by identification of the point on the base line at which the second line is to be constructed. The cursor will be moved to identify the point, and the function activated by making a keystroke or pressing a button on the cursor control device. Similarly, a line may need to be deleted, in which case it is first identifed by positioning the cursor and then removed by making a keystroke or by pressing a delete button on the cursor control.

The foregoing description has, of necessity, been general in nature. Cursor

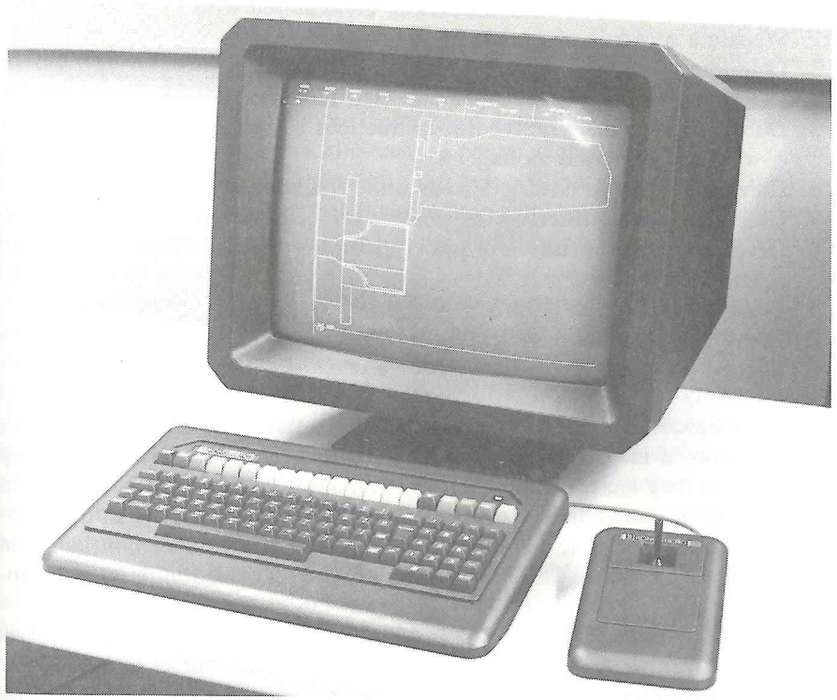


Figure 10.6b *Part programming station showing use of computer graphics for program proving.*

controllers, even of one particular type, are very variable in design. The way in which cursors are used varies from one CAPP system to another. The essential thing to appreciate is that a part programmer is not required to be familiar with all the possibilities; nevertheless, it will be necessary to become competent in the application of the particular devices associated with the program system he or she will be using, and this can only be achieved by using the equipment.

COMPUTER-ASSISTED PROGRAMMING ACTIVITIES

In Chapter 8 the procedure for manual part programming, taking the detail drawing as a starting point, was listed. That list is reproduced below, but this time the programming activities that will be assisted by the use of CAPP are shown in bold type.

1. Select a machine capable of handling the required work.
2. Prepare a schedule of machining operations.
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 feedrates.**
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.**

Before proceeding further it is necessary to make the point that the more practical elements of the list, that is, those numbered from 1 to 5, are as necessary for CAPP as they are for manual part programming. CAPP does not eliminate the need for the programmer to have a good working knowledge in the practicalities of metal cutting. The reader is referred to the fuller consideration given to these aspects of CNC machining which is included in previous chapters.

To return to the CAPP process. It will be assumed that consideration has been duly given to the practicalities of a particular machining task, and the computer-aided element of preparing the part program can begin.

It is not possible to list precisely the stages in the CAPP process since there is some variation in approach according to the type of system being used. But in general terms the stages may be itemized as follows:

1. Define the geometric detail of the component. This will involve a series of individually constructed elements that embraces the final component detail.
2. Use the geometric detail to define appropriate machining sequences.
3. Supplement the proposed machining sequences with technology data relating to tooling, feedrates, spindle speeds, etc.
4. Process these data to determine tool paths and to produce a cutter location data file.
5. Produce a CRT screen or paper tool path plot for initial program prove out.
6. Postprocess these data into a form or language that is acceptable to the machine to be used.
7. Transmit these data direct to the machine tool. Alternatively, a punched tape may be produced, or the program otherwise recorded for future use.

CAPP SYSTEMS

Before a CAPP system can be used to prepare a program, time will have to be spent in becoming familiar with the techniques or language to be used—just as it is necessary to study the language of a machine control system before programs for a particular machine can be prepared manually. But the use of CAPP does have a major advantage in this respect: it is probable that the programmer will be required to become familiar with only one technique, since it is possible to postprocess or translate the data into whatever machine control language is to be used.

It is not possible in a text of this nature to give a comprehensive review of every CAPP system, since there are far too many currently in use. Neither would it be of value to consider a particular system in detail. Later in the text, however, programming examples are included in an attempt to give at least a general impression of some of the variations that exist. These examples will be given in the Compact II language by Manufacturing Data Systems Incorporated and APT (Automatically Programmed Tools). Compact II and APT were selected as examples for their leading roles in the development of CAPP and their major influences in CAD/CAM evolution.

In reality, a part programmer will find it necessary to devote as much time as possible to becoming proficient in the application of the particular system he or she will be required to use. Some of the techniques and skills developed in the use of one system are likely to be transferable to another if the need arises.

Although the number of CAPP systems available are many and varied they may be generally defined as being either language or graphics based.

The basic difference between the two concepts is the way in which the ap-

propriate tool paths for the machining sequences are ascertained. The following text deals with language-based programming. Graphics-based programming will be discussed further.

LANGUAGE-BASED SYSTEMS

Early CAPP systems were entirely language-based, the geometry of the part being described by a series of statements constructed from letters of the alphabet, numbers, and a few other symbols. The systems were not interactive, there was no indication if errors had been made in the data, and therefore no correction was possible during the input process. Confirmation of the validity of data could only be ascertained by processing it. If necessary, the program could then be edited. The only visual confirmation of the program data was via a diagram produced on an interfaced plotter after the data entry was completed.

Modern language-based CAPP systems also use alphanumeric input supplemented by certain symbols, but have been considerably improved by the incorporation of computer graphics. As data are entered, there is an instant corresponding graphic display giving an indication of the validity of the input. The systems are fully interactive: if data are not acceptable, the fact will be indicated, often with messages to indicate why this is so, and the programmer can then act on this information and modify the input as programming proceeds.

Having defined the component by the use of a series of geometric statements, the programmer then selects elements from the overall construction and includes these in what is, in effect, a composite statement that will form the basis of a particular machining operation. The composite statement, which may represent, for example, a profile or a series of holes, will be expressed in language form.

The composite statements are now supplemented by data relating to speeds and feeds, tooling, etc. These are referred to as tool change data statements, and will be discussed further.

At this stage these data are processed to determine tool paths and to generate a cutter location data file, referred to as CL Data. Finally, the data are post-processed to generate a program in machine tool code for the particular machine to be used. Each of these stages is explained further in the following text.

GEOMETRIC DEFINITION

It is assumed that a person making a study of CAPP will already be familiar with manual part programming techniques, and will therefore appreciate that

any machined feature or profile can be geometrically defined. He or she will already be familiar with defining tool movements in relation to the workpiece as being linear, circular, and point-to-point. An appreciation of how one geometric feature can intersect with another and the need to define dimensionally such intersection points should also be well understood.

In manual part programming the profile is, in effect, split up into its geometric elements. This is also the case with CAPP, in which the shape or feature to be machined is expressed in terms of directions, distances, lines, points, and circles.

The way in which these geometric elements are generally defined when using CAPP systems is listed below. The lists should not be considered to be definitive since the approach to geometric construction differs between programming systems, and there are also variations in the words used to describe what is essentially the same feature. A further complication is that some of the definitions used, while perfectly logical and therefore acceptable when applied to geometric construction involving computer graphics, do not conform to true mathematical expression. However, the reader is assured that the descriptions that follow are typical of those likely to be encountered.

Directions are defined in the usual way, that is by the use of the letters X, Y and Z which relate to the axes of movement of the machine tools. Distances are given a dimensional value in millimeters or inches and angles are stated in degrees.

A point may be defined in a number of ways as follows:

(a) As a zero;
(b) as a point with known Cartesian coordinates;
(c) as a point with known polar coordinates;
(d) as an intersection of two straight lines;
(e) as an intersection of a straight line and a curve;
(f) as an intersection of two curves.

Examples of the above definitions are illustrated in Figure 10.7 (a) to (f), respectively.

Straight lines may also be defined in a number of ways and the following descriptions correspond to the illustrations in Figures 10.8 (a) to (h).

(a) As being parallel to a stated axis;
(b) as being at a known distance from a previously defined point and at a known angle to a previously defined straight line;
(c) as being between two known points;
(d) as being tangential to two known circles;
(e) as being parallel to a defined straight line;
(f) as being perpendicular to a defined straight line;
(g) as being perpendicular to a defined point;
(h) as passing through a defined point, at an angle.

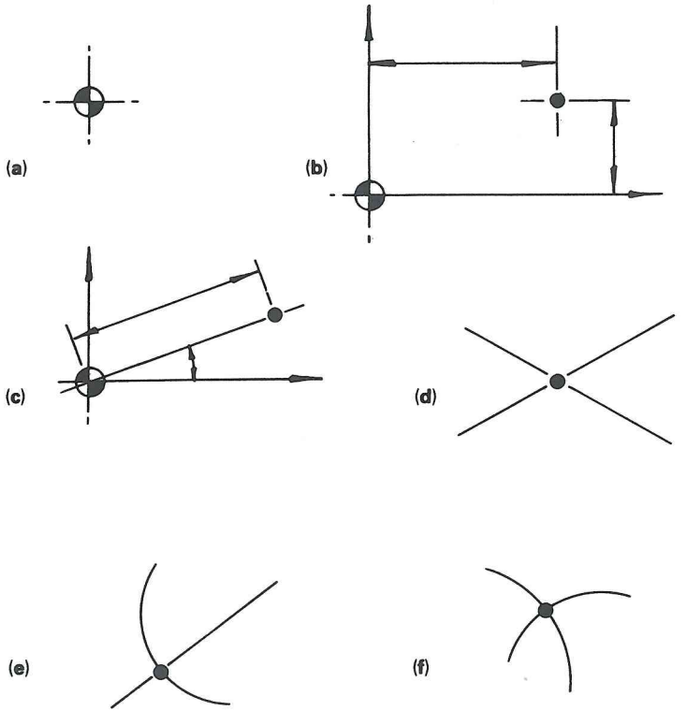


Figure 10.7 *Point definition.*

Circles may be defined as follows, and as illustrated in Figures 10.9 (a) to (g).

(a) As a known radius passing through two defined points;
(b) as an unknown radius passing through three defined points;
(c) as a center point and passing through a defined point;
(d) as a center point and tangential to a defined straight line;
(e) as being tangential to a defined line, passing through a defined point and with a known radius;
(f) as being tangential to two defined lines and with a known radius;
(g) as being tangential to three lines and with a known radius.

A further complication with some constructions is that two versions are sometimes possible. Consider Figure 10.10(a), a radius passing through two defined points. One construction is shown in full line and an alternative construction is shown in broken line. Similarly, the construction shown in Figure 10.10(b), of a circle of given radius tangential to two defined lines, has four possible versions as indicated.

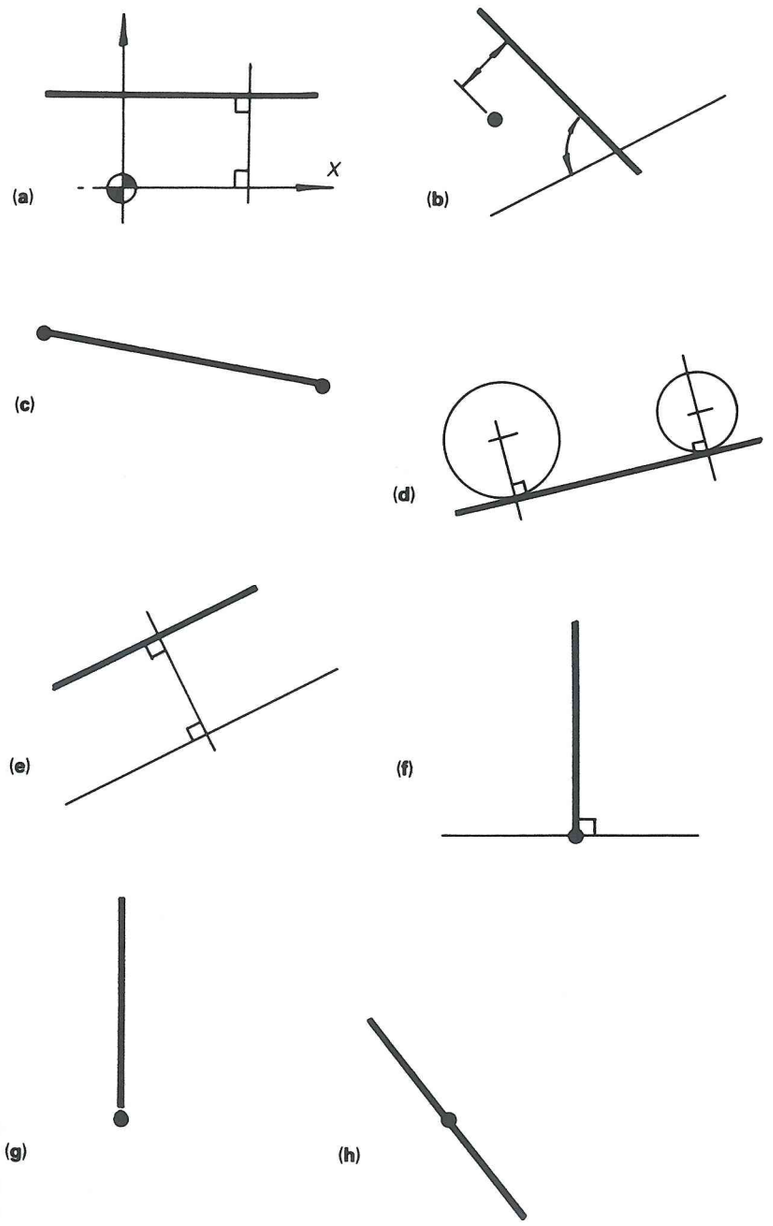


Figure 10.8 *Line definition.*

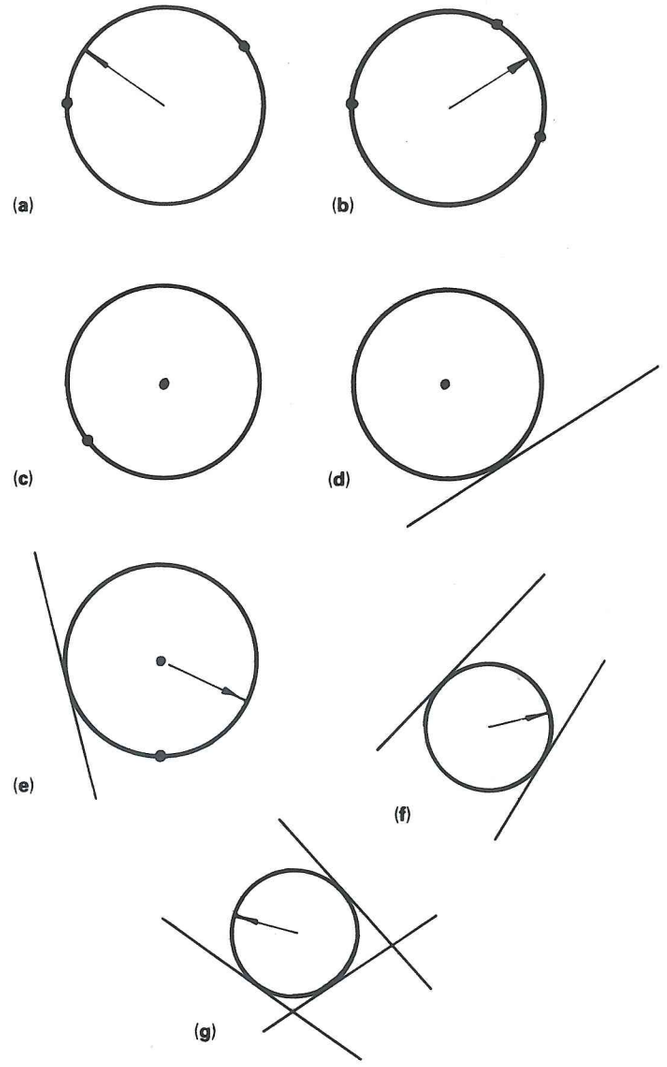


Figure 10.9 *Circle definition.*

Clearly there is a need to clarify the situation by providing the new element with a sense of orientation or direction in relation to the existing geometry. The way this is achieved differs between one system and another and the student will require specific instructions relating to the system he or she will be using.

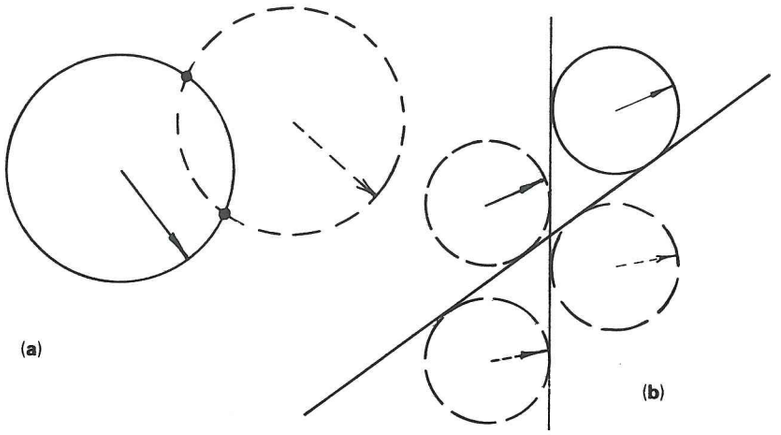


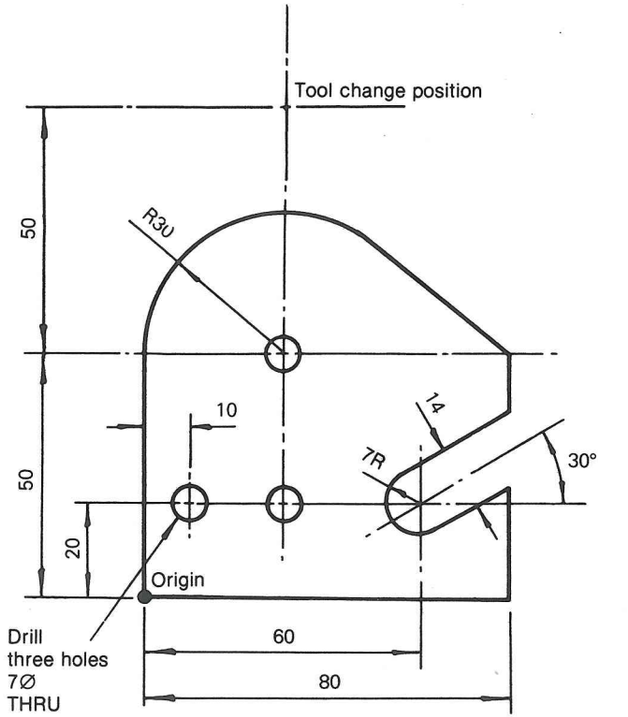
Figure 10.10 *Alternative constructions.*

GEOMETRY FILE CONSTRUCTION

The programmer begins the process of geometric definition by first studying the profile or feature to be machined and then giving each element an identity. He or she may do this by marking a drawing prior to entering data into the computer or, if sufficiently competent in the use of the system, the programmer may allocate identities as the entries are made. The student is likely to benefit, at least in the early stages of using a particular CAPP system, by adopting the first approach. As with all methods of programming a logical approach is essential to avoid making frustrating mistakes.

The precise method used to identify elements varies from one system to another, but it is common practice to give each element a numerical identity, which is then followed by the appropriate definition. Thus a line identified as line number 7 that is to be constructed from point number 1 at an angle of 90° may be programmed simply as L7, P1, A90. A complete profile consisting of a series of lines, circles, and points previously defined may be listed as follows: PF, P1, L1, L2, L3, L4, C1, L5, L6, P2. The initials PF identify the statement as being a profile.

It should be possible to gain a general appreciation of the techniques used by studying the two examples of geometry statements listed below. Both lists relate to the milled component illustrated in Figure 10.11. For the sake of simplicity the problems associated with holding such a component while the profile is machined have been ignored. Normally, clamping arrangements and work-holding devices have to be accommodated within the part program if collisions are to be avoided. Areas they will occupy, which in effect become "no go" areas for the tool, have to be identified dimensionally and may be displayed graphically as part of the general geometry.



Material: mild steel 10 mm thick.

Figure 10.11 *Component detail.*

The program definition lists which follow were prepared by two different CAPP systems for comparison. The result is two different approaches to the geometry definition. Reference to Figures 10.12 and 10.13 will indicate how each definition refers to the part geometry.

The reader will note the common use of P or PT to indicate a point, L or LN to indicate a line, and C or CIR to indicate a circle. But there, apart from the numerical identity referred to above, the similarities end. Further study of the lists will show that the variations become even more pronounced when the individual elements are compared.

Example No. 1 Software: Compact II (Figure 10.12)

```
MACHIN, MILL                          Statement identifies machine used
IDENT                                 Program identifier
SETUP, 30LX, 100LY, 100LZ             Machine setup information
BASE, 0XB, 0YB, 20ZB                  Absolute zero location
```



```
DPT1, 0XB, 0YB, 0ZB
DPT2, PT1, 10X, 20Y                   Definition of point locations
DPT3, PT2, 30XB
DPT4, PT3, 50YB
DCIR1, PT4, 30R                       Definition of circles
DCIR2, 60XB, 20YB, 0ZB, 7R
DLN1, XB
DLN2, 80XB, 50YB, 0ZB, CIR1, YL
DLN3, 80XB                            Definition of lines
DLN4, YB
DLN5, 30CCW, CIR2, YL
DLN6, PARLN5, CIR2, YS
```

Example No. 2 Software: APT

```
PARTNO PROFILE DEFINITION EXAMPLE     Program identification
       CUTTER/0                       Cutter call up
       MACHIN/MILL                    Statement identifies machine used

P1  = POINT/0,0,20
P2  = POINT/10,20,20                  Definition of point locations
P3  = POINT/30,20,20
P4  = POINT/30,50,20
C1  = CIRCLE/CENTER,P4,RADIUS,30      Definition of circles
C2  = CIRCLE/60,20,20,7
L1  = LINE/P1,PARLEL,LY
L2  = LINE/80,50,20,RIGHT,TANTO,C1
L3  = LINE/PARLEL,L1,XLARGE,80        Definitions of lines
L4  = LINE/P1,PARLEL,LX
L5  = LINE/LEFT, TANTO,C2 ,ATANGL,30,YLARGE
L6  = LINE/PARLEL,L5,YSMALL,14

SETPT  = POINT/30,100,100             Machine setup point
```

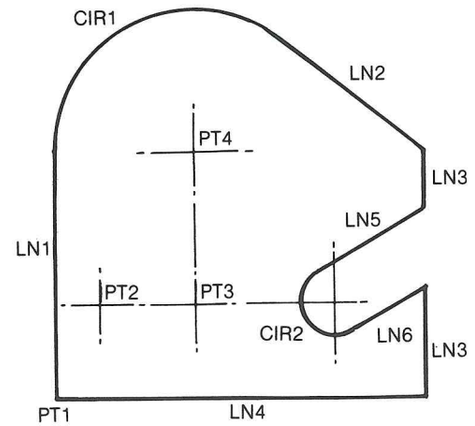


Figure 10.12 *Profile definition—MDSI Compact II (Manufacturing Data Systems Inc.).*

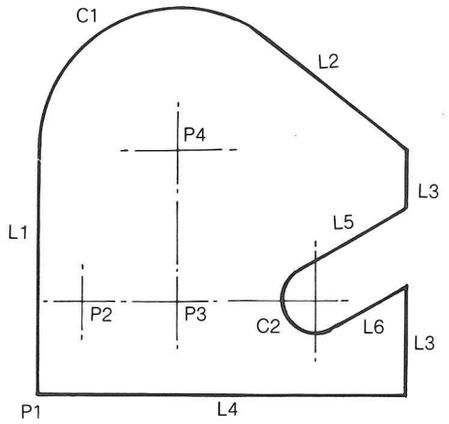


Figure 10.13 *Profile definition—APT (Automatic Programmed Toolpath).*

Now that we have looked at two examples of geometry statements from popular computer-assisted languages let us review an entire program in Compact II. It is noted that programs of this type can be broken down into five major areas of initialization, geometry, tool change, tool motion, and termination as can be seen in Figures 10.14 through 18. The programming process

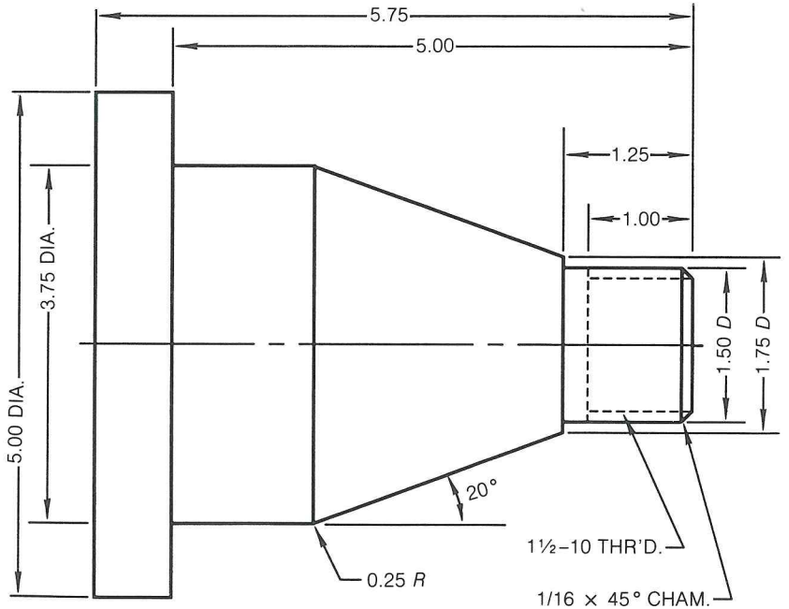


Figure 10.14 *Spindle bolt part print. (From Numerical Control Technology Workbook, courtesy of Schlumberger Technologies Inc., Ann Arbor, MI).*

starts from a part print such as Figure 10.14, from which the programmer decides on the machine, tools, and process to use. In this example the programmer has decided to use three tools, the first for rough turning leaving 0.020 in. finish stock on the diameter and 0.005 in. stock on the length for finish. The second tool selected is for finish turning, while the third and final tool is for a single-point threading operation.

The second step the programmer has taken is to determine what the machine setup should be for this program. Figure 10.15 shows the machine layout decided upon. The machine zero is located at the chuck end stop end of the part and the machine tool turret reference point will be established by the operator 10.5 in. from this point in the *Z* axis. The *X* axis turret reference point will be established 8.0 in. from the center of the workpiece.

Section I Initialization

Once the machine setup has been determined, the program initialization statements can be written for the computer program. Figure 10.16 shows the five

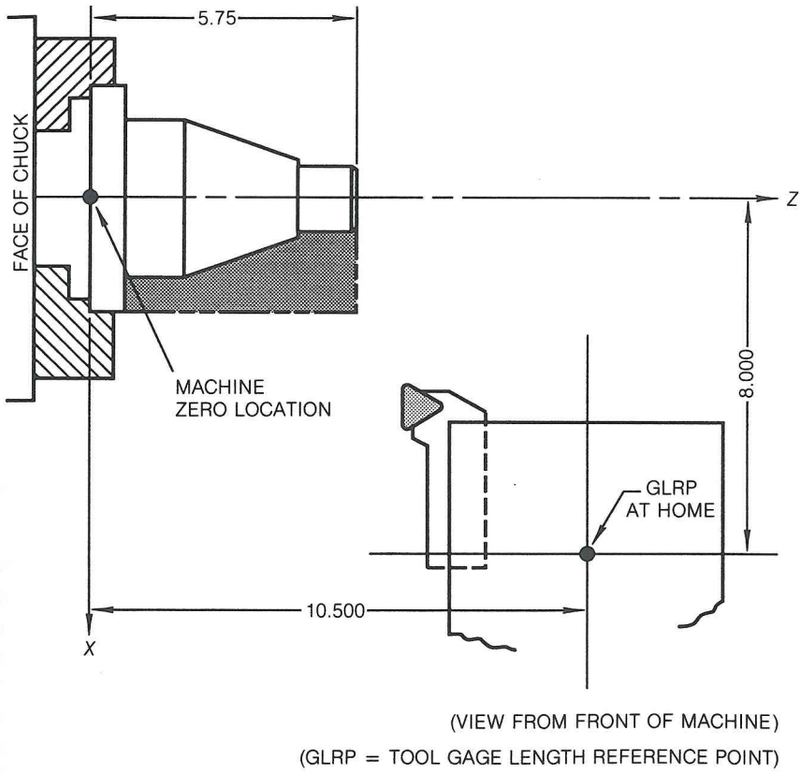


Figure 10.15 *Machine setup layout. (From Numerical Control Technology Workbook, courtesy of Schlumberger Technologies Inc., Ann Arbor, MI.).*

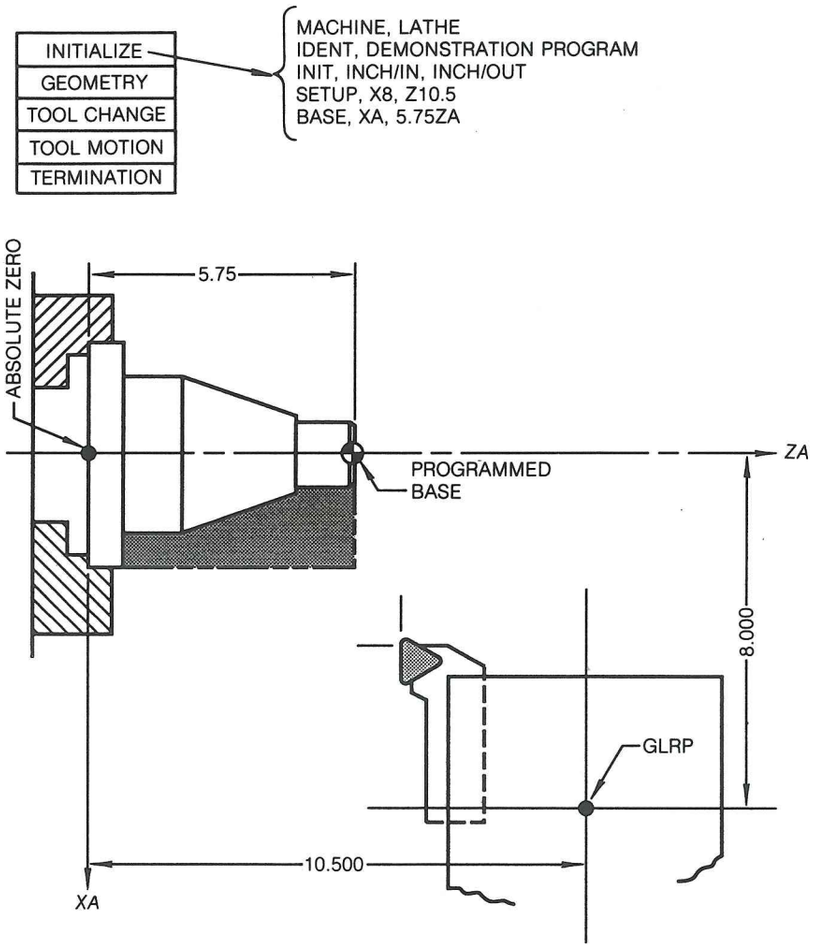


Figure 10.16 *Initialization statements. (From Numerical Control Technology Workbook, courtesy of Schlumberger Technologies Inc., Ann Arbor, MI.).*

initialization statements for a Compact II program. Starting with the MACHINe statement the computer is told what specific type of machine tool it is creating a program for. Since this is a demonstration program, just a generic post or machine file is used. For each specific type of machine and control there will be a special command. Next is the IDENTification statement, which assigns the title to be output on the various files generated by the computer. The third line is a INITialization statement to indicate whether input will be inch or metric and whether output is wanted in inch or metric. The fourth statement SETUP informs the computer as to where the machine tool turret reference point is in

relation to the machine absolute zero point location. The fifth and final initialization statement is BASE. The base statement informs the computer of the location of the program zero location, used for defining part geometry, in relation to the machine absolute zero.

Section II Geometry

The second major section of a computer program involves the description of the part's geometry to be machined. The programmer first marks up the part print as to the geometry required to machine the part. The geometry labels are normally placed on the part as in Figure 10.17 as a record for future reference. In the example in Figure 10.17 the programmer has defined two lines and two

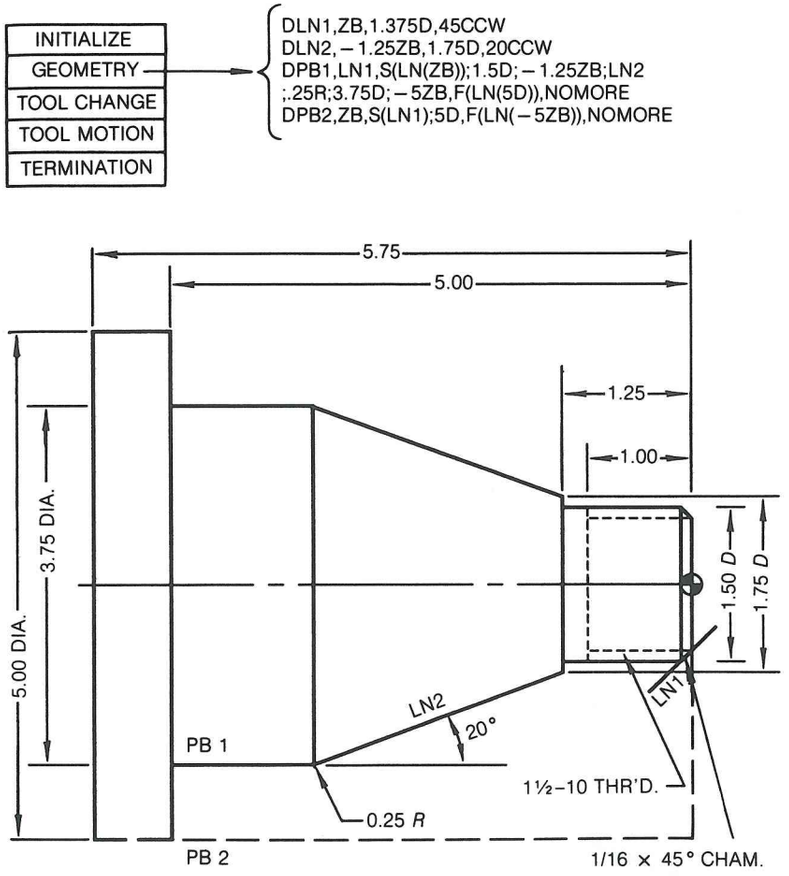


Figure 10.17 *Part geometry definition. (From Numerical Control Technology Workbook, courtesy of Schlumberger Technologies Inc., Ann Arbor, MI.).*

part boundaries. Line LN1 represents the 1/16 × 45° chamfer, while Line LN2 is the 20° part angle. Part boundary PB1 is defined as the finish part outline, while PB2 is the outline of the rough stock.

The amount of geometric definitions required and their complexity will vary with the skill of the programmer and the capability of the language to handle complex statements. Computer systems will handle geometry verifications in different ways. Some systems process all statements in a batch mode together for statement validity and then allow the system to plot the results on paper or a CRT screen. Other language-based systems are fully interactive: the correctness of data input is verified as programming proceeds and error messages are displayed if appropriate. This ensures that the data input is acceptable to the system, but it does not necessarily ensure that the programmer has not made other mistakes, so it makes sense to reprocess the input in its entirety as a final check.

When the geometry statements have been verified as being correct, it is possible to obtain a print from an interfaced printer of the data listing. This may be required for filing for future reference, forming part of the general documentation relating to that particular job.

It is also possible to obtain printed copies of the graphic construction developed during the programming process. Figure 10.18 illustrates the graphics that appeared on the plotter when, using the Compact II software, the geometry statements were entered for the lathe exercise in Figure 10.17.

From the plot the programmer can easily see the various pieces of geometry defined and can visualize the outline of the rough and finish piece part. Geometry relationships and gross errors can be determined in this type of plot.

TOOL CHANGE DATA

After the geometry of the component has been defined, the programmer has to consider the more practical aspects of producing a machined component,

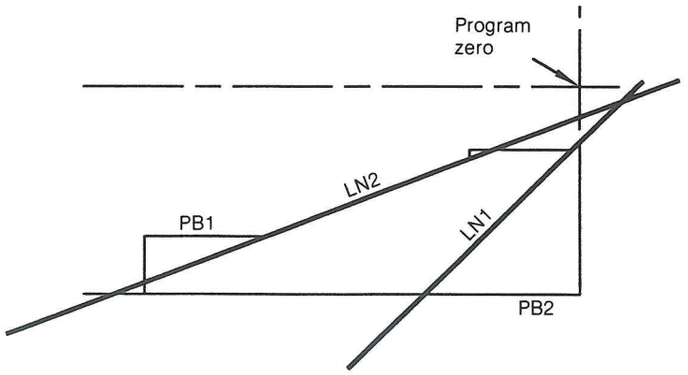


Figure 10.18 *Geometry plot.*

such as the sequence of operations, the cutting tools to be used, and the choice of appropriate cutting speeds and feeds. If the programming task has been approached in a logical manner, most of these aspects will have been considered before the CAPP process was started. Now data defining these factors have to be added to the part program to supplement the geometric data previously entered. It is possible that some computer assistance may be available.

Tool Change Data statements are included in the Compact II and APT program examples that follow. See Figure 10.19. The tool change statement will be found in various locations of the machining body of the program, wherever a tool selection is required, and will be followed by its tool motion statements.

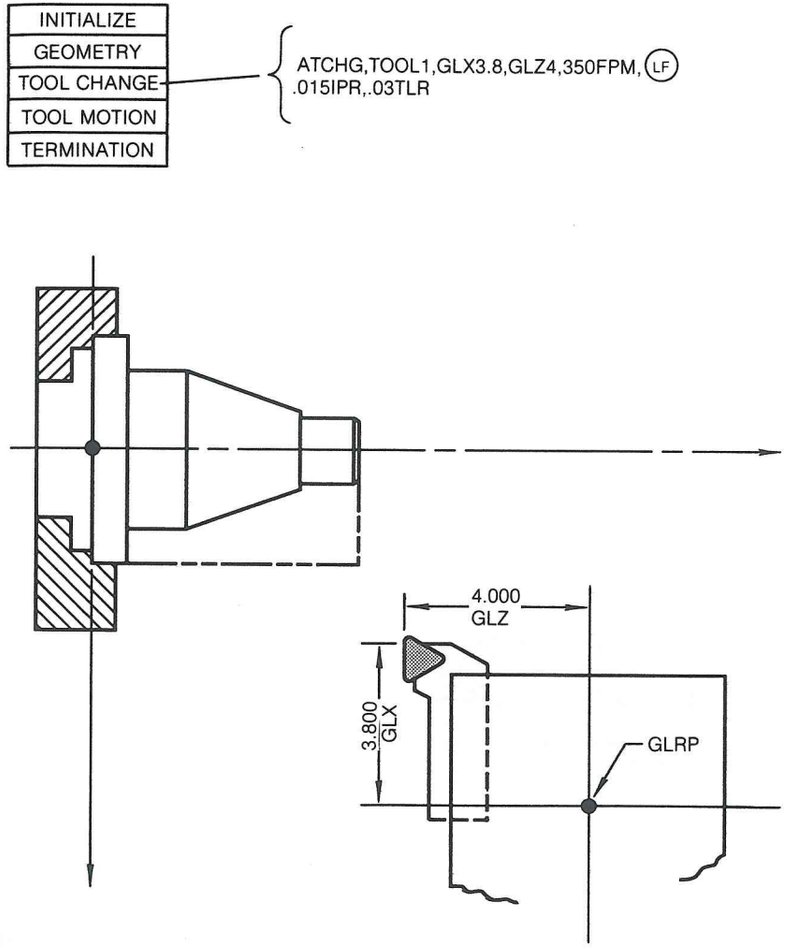


Figure 10.19 *Tool change data. (From Numerical Control Technology Handbook, courtesy of Schlumberger Technologies Inc., Ann Arbor, MI.).*

Tool statements normally will include information on tool number, gage lengths and diameters, cutting speeds, and feed rates.

On some systems the tooling available for a particular machine will be listed and contained on file within the programming system. Such a data file is referred to as a "tool library" and can be displayed on the screen. All the data relevant to a particular tool, such as the material from which it is made, its shape, and its dimensions, will be indicated together with an identity code for use within the part program. The dimensions of the tool, in particular its radius or diameter, are of particular importance since they will have a direct effect on the cutter paths automatically generated at the next stage of the CAPP process.

Cutting speeds and feeds can be determined without computer assistance, and entered into the program in much the same way as when preparing a manual CNC part program. On the other hand, there may be assistance via the system in response to data input identifying the cutting tool and the part material. The correct speeds and feeds will then be determined automatically and included in the program.

INITIALIZE	
GEOMETRY	
TOOL CHANGE	
TOOL MOTION	→ CUT,PB1,MB2,MAXDP.255,XSTK.02,ZSTK.005
TERMINATION	

(One statement will produce all the roughing passes indicated.)
This statement causes the rough cutting of material between part boundary 1 and material boundary 2. A finish stock of 0.02 in. will be left in the *X* axis, while 0.005 in. will be left in the *Z* axis.

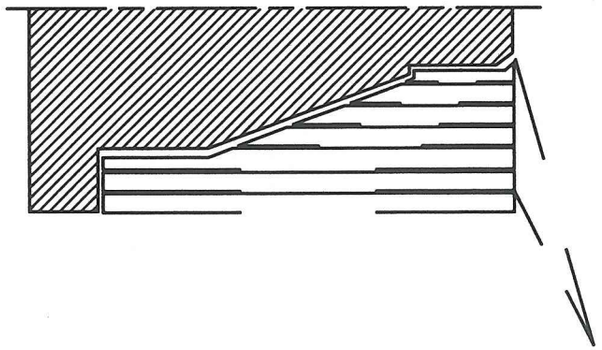


Figure 10.20 *Roughing cutting cycle in Compact II. (From Numerical Control Technology Workbook, courtesy of Schlumberger Technologies Inc., Ann Arbor, MI.).*

TOOL MOTION DATA

The fourth major section of a CAPP program is Tool Motion statements, which are grouped with the corresponding tool change statement. These statements will determine how the part will be cut. Machining cycles and feed/rapid traverse moves will be established. Knowledge of the particular language used is important here because there are many options from which to choose. Figures 10.20, 10.21, and 10.22 give only a few examples of tool motion statements for lathe work.

When defining the sequence and types of machining operations to be carried out, the programmer will be required to take into consideration the special cycles that are an inbuilt feature of the programming system. All the normal machining sequences—drilling, screw cutting, face milling, boring, etc.—are likely to be catered for. It will also be possible to generate subroutines. To use these facilities effectively the programmer will need to be fully conversant with the particular system being used, and this is only achieved by experience.

INITIALIZE	
GEOMETRY	
TOOL CHANGE	→ ATCHG,TOOL2,GLX3.8,GLZ4,450FPM,^{LF} .008IPR,.03TLR
TOOL MOTION	→ CUT,PB1,FINISH,OSTK
TERMINATION	

The first statement calls for a tool change to the finish turning tool. The second statement calls for the finish profile of the part to be cut.

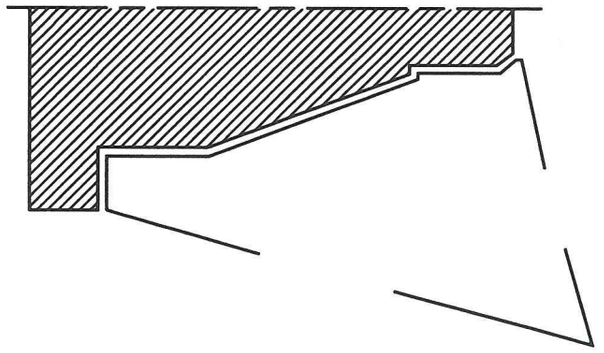
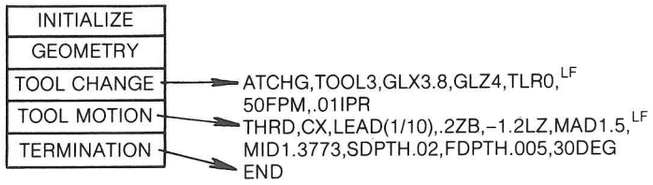


Figure 10.21 *Finishing turning—Compact II. (From Numerical Control Technology Workbook, courtesy of Schlumberger Technologies Inc., Ann Arbor, MI.).*



The first statement calls for a tool change to the threading tool. The second statement calls for a multiple threading cycle to be generated. The third statement calls for termination of the computer program and commands the tool back to its home position.

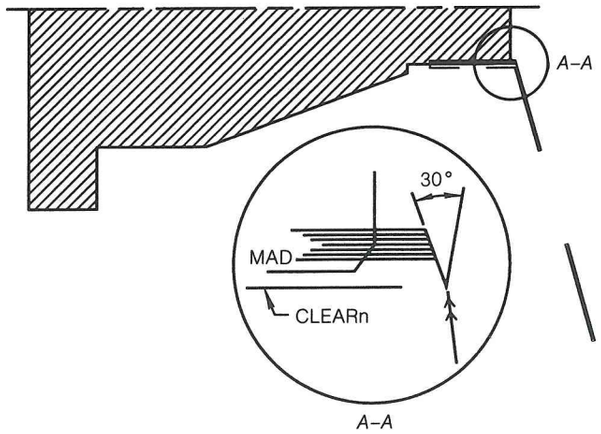


Figure 10.22 *Threading cycle—Compact II. (From Numerical Control Technology Workbook, courtesy of Schlumberger Technologies Inc., Ann Arbor, MI.).*

TERMINATION

The fifth major section of most CAPP programs is the termination of the program. Many systems use a single statement of END or FINISH to indicate that the program is complete. This single statement will normally cause the generation of machine "CNC" program steps to return the tool to its home position and complete an orderly machine shut down. The complete Compact II program can be reviewed in Figure 10.23.

CUTTER LOCATION DATA

When the computer is programmed with all the data defining the part geometry, the machining sequence, tooling, etc., the process of determining the cutter location data, referred to as the CL Data, can begin. In simple terms, the CL

Compact II Part Program

MACHIN,LATHE	Identifies machine used
IDENT,DEMONSTRATION PROGRAM	Program file identification
INIT,INCH/IN,INCH/OUT	Set up data format in and out
SETUP,X8,Z10.5	Machine setup
BASE,XA,5.75ZA	Program zero location
DLN1,ZB,1.375D,45CCW	Define Line 1
DLN2,−1.25ZB,1.75D,20CCW	Define Line 2
DPB1,LN1,S(LN(ZB));1.5D;−1.25ZB;LN2;.25R; 3.75D;−5ZB,F(LN(5D)),NOMORE	Define finish part boundary
DPB2,ZB,S(LN1);5D,F(LN(−5ZB)), NOMORE	Define material boundary
ATCHG,TOOL1,GLX3.8,GLZ4,350FPM, .015IPR,.03TLR	Define roughing tool
CUT,PB1,MB2,MAXDP.255,XSTK.02, ZSTK.005	Create roughing cuts
ATCHG,TOOL2,GLX3.8,GLZ4,450FPM, .008IPR,.03TLR	Define finish turning tool
CUT,PB1,FINISH,0STK	Create finish turning pass
ATCHG,TOOL3,GLX3.8,GLZ4,TLR0, 50FPM,.01IPR	Define threading tool
THRD,CX,LEAD(1/10),.2ZB,−1.2LZ,MAD1.5, MID1.3773,SDPTH.02,FDPTH.005,30DEG	Create threading pass
END	Terminate program

Figure 10.23 *Compact II program. (From Numerical Control Technology Workbook, courtesy of Schlumberger Technologies Inc., Ann Arbor, MI.).*

Data can be described as the dimensional definition of the cutter path from a defined datum point.

In determining the CL Data the computer automatically calculates the movements necessary to achieve the geometric features previously defined. In doing so, account will be taken of cutter sizes; where appropriate, compensation for the cutter radius will be made. Where area clearance is required, and excess stock material has to be removed, the computer will determine the appropriate tool paths.

CL Data can be viewed on some CRT screens and a printout can be obtained if required.

Tool paths can be displayed graphically, in some cases with a three-dimensional or pictorial effect. It is also possible to produce, via a plotter or a printer, a diagrammatic representation of the part geometry and the cutter paths in relation to that geometry. Both the graphical display on the CRT screen and the plotter output are usually enhanced by the use of different colors to indicate different features: one color for the geometric shape of the component and a different color indicating the paths of each tool to be used, for instance.

In Figures 10.20 through 10.22 the printouts of the cutter paths for programming the component shown in Figure 10.14 is illustrated.

When the CL Data are considered to be correct, the final stage in the CAPP process, that of postprocessing, can be undertaken.

With the CL Data file compiled, it is also possible, on some systems, to determine the time that will be taken to machine the part. The computer calculation is based on the cutting speeds and feeds entered as part of the technology data.

It should be noted that computer languages may be interactive or batch processed. If interactive, the system may check each individual statement for validity, then create its CL Data, postprocess it to machine language, and plot/display the results or error before proceeding to the next statement. Batch processing systems, on the other hand, takes the entire program through each step of processing (statement validity, CL Data, postprocessing, and display) before going on to the next step. In batch processing the computer processing is normally stopped at the end of the step reporting errors with an error statement print out. Interactive program systems will allow error correction "on the fly."

POSTPROCESSING

Postprocessing is the stage in the CAPP process where the CL Data and other information relevant to the machining of the component is assembled into a form that will be accepted and meaningful to the control system of the particular machine to be used. Features such as G and M codes, previously not part of the program data, are now automatically incorporated.

Because there are many variations in the control systems fitted to machine tools, it is necessary to have a postprocessor to suit each control system for which part programs are to be prepared. The manufacturers of CAPP systems will supply specific postprocessors to order, these being immediately available for the more widely used machine controls. It is also possible with some systems to purchase a "writing kit/generic post," which permits users to compile their own postprocessor, for it is actually a relatively simple computer program. In this way, if a new machine is acquired, it can be readily assimilated into the CAPP system.

The postprocessing of data into a machine control language is achieved very rapidly, being simply a case of making a few key strokes. The resulting program can now be recorded for future use in whatever form is deemed to be appropriate.

Examples of postprocessed programs follow in Figures 10.24 and 10.25. Figure 10.24 shows a combination program output that is good for programmer use during machine tryout. This output shows the computer statement followed

by the CL Data or the machine information it generates. It is good for tryout because when an error is found in the machine program you know right away what computer statement generated it. Figure 10.25, on the other hand, is a machine tool operator's program read out. On the machine tool program output no other information is included to cloud the issue.

```
>MACHIN,LATHE                                          }
MAIN 9.27 LINK 3.31 SYS 8.23 L# 136                    }
>IDENT, DEMONSTRATION PROGRAM                          }
12-22-90 10:55                                         }
>INIT,INCH/IN,INCH/OUT                                 }   INITIALIZATION
>SETUP,X8,Z10.5                                       }
>BASE,XA,5.75ZA                                        }
= X . Z 5.75                                           }
>DLN1,ZB,1.375D,45CCW                                  }
= X .6875 Z 5.75 A 45.                                 }
>DLN2,-1.25ZB,1.75D,20CCW                              }
= X .875 Z 4.5 A 20.                                   }
>DPB1,LN1,S(LN(ZB));                                   }
>;1.5D;                                                }
>;-1.25ZB;                                             }   GEOMETRY
>;LN2;                                                 }
>;.25R;                                                }
>;3.75D;                                               }
>;-5ZB,F(LN(5D)),NOMORE                                }
>DPB2,ZB,S(LN1);                                       }
>;5D,F(LN(-5ZB)),NOMORE                                }
>ATCHG,TOOL1,GLX3.8,GLZ4,350FPM,                       TOOL CHANGE
  .015IPR,.03TLR                                       Rough Turning Tool
```

Figure 10.24 *Combination file output of part in Figure 10.14. (*Note: *Programmer-generated statements are indicated by the ">" symbol. All other information is generated by the computer.).*

```
>CUT,PB1,MB2,MAXDP.255,XSTK.02,ZSTK.005
N001 X-01.825 Z-00.6352 F200.00 S0298 T0001 M03
N002 X-00.1 Z-00.0348 F010.00
N003 Z-05.045 F004.47
N004 X 00.255 F004.04 S0269
N005 Z 04.945 F200.00
N006 Z 00.1 F010.00
N007 X-00.41 F200.00 S0336
N008 X-00.1 F010.00
N009 Z-05.045 F005.04
N010 X 00.255 F004.49 S0299
N011 Z 04.945 F200.00
N012 Z 00.1 F010.00
N013 X-00.41 F200.00 S0386
N014 X-00.1 F010.00
N015 Z-03.6317 F005.79
N016 X 00.1428 Z-00.3924 F005.37 S0358
N017 G02 X 00.0172 Z-00.0974 I00.2678
  K00.0974 F005.33 S0355
N018 Z-00.9235 F005.30 S0353
N019 X 00.095 F005.07 S0338
N020 Z 04.945 F200.00
N021 Z 00.1 F010.00
N022 X-00.41 F200.00 S0453
N023 X-00.1 F010.00
N024 Z-02.9311 F006.80
N025 X 00.255 Z-00.7006 F005.82 S0388
N026 Z 03.5317 F200.00
N027 Z 00.1 F010.00
N028 X-00.41 F200.00 S0547
N029 X-00.1 F010.00
N030 Z-02.2305 F008.21
N031 X 00.255 Z-00.7006 F006.84 S0456
N032 Z 02.8311 F200.00
N033 Z 00.1 F010.00
N034 X-00.41 F200.00 S0692
N035 X-00.1 F010.00
N036 Z-01.5298 F010.38
N037 X 00.255 Z-00.7006 F008.28 S0552
N038 Z 02.1304 F200.00
N039 Z 00.1 F010.00
```

TOOL MOTION
Rough Turning Tool

Figure 10.24 *(Continued)*


```
N040 X-00.41 F200.00 S0941
N041 X-00.1 F010.00
N042 Z-00.073 F014.12
N043 X 00.055 Z-00.055 F013.28 S0885
N044 Z-01.167 F013.10 S0873
N045 X 00.1145 F011.52 S0768
N046 X 00.0855 Z-00.2348 F010.50 S0700
N047 Z 01.4298 F200.00
N048 Z 00.1 F010.00
N049 X-00.2125 F200.00 S1024
N050 X-00.1 F010.00
N051 Z-00.0155 F015.36
N052 X 00.0575 Z-00.0575 F014.33 S0955
N053 X 03.355 Z 00.7215 F200.00
N054 X 00.1 Z 00.0215 F010.00
ABS D 8.4 ZB .75
```

```
>ATCHG,TOOL2,GLX3.8,GLZ4,450FPM,
  .008IPR,.03TLR
```

TOOL CHANGE
Finish Turning Tool

```
>CUT,PB1,FINISH,0STK
N055 X-03.4125 Z-00.6509 F200.00 S1307 T0002
N056 X-00.1 Z-00.0191 F010.00
N057 Z-00.0375 F010.46
N058 X 00.0925 Z-00.0925 F009.29 S1161
N059 Z-01.17 F009.16 S1145
N060 X 00.116 F008.03 S1004
N061 X 00.2754 Z-00.7545 F006.02 S0753
N062 X 00.3798 Z-01.0447 F004.52 S0565
N063 X 00.3369 Z-00.9266 F003.72 S0465
N064 G02 X 00.0169 Z-00.0957 I00.2631
  K00.0957 F003.68 S0460
N065 Z-00.9285 F003.66 S0458
N066 X 00.625 F002.76 S0345
N067 X 01.6408 Z 05.62 F200.00
N068 X 00.0292 Z 00.1 F010.00
ABS D 8.4 ZB .75
```

TOOL MOTION
Finishing Turning Tool

```
>ATCHG,TOOL3,GLX3.8,GLZ4,TLR0, 50FPM,.01IPR
```

TOOL CHANGE
Threading Tool

Figure 10.24 *(Continued)*

```
>THRD,CX,LEAD(1/10),.2ZB,-1.2LZ,MAD1.5,
  MID1.3773,SDPTH.02, FDPTH.005, 30DEG
N069 X-03.4 Z-00.55 F200.00 S0250 T0003
N070 X-00.0695 Z-00.0401 F002.50
N071 G33 Z-01.1599 K0.1
N072 X00.0695 F200.00
N073 Z 01.2
N074 X-00.0854 Z-00.0493 F002.50
N075 G33 Z-01.1507 K0.1
N076 X 00.0854 F200.00
N077 Z 01.2
N078 X-00.0976 Z-00.0564 F002.50
N079 G33 Z-01.1436 K0.1
N080 X 00.0976 F200.00
N081 Z 01.2
N082 X-00.1063 Z-00.0614 F002.50
N083 G33 Z-01.1386 K0.1
N084 X 00.1063 F200.00
N085 Z 01.2
N086 X-00.1113 Z-00.0642 F002.50
N087 G33 Z-01.1358 K0.1
N088 X 00.1113 F200.00
N089 Z 01.2
ABS D 1.6 ZB .2
```

TOOL MOTION
Threading Operation

```
>END
```

PROGRAM TERMINATION

```
N090 X 03.4 Z 00.55
N091 M05
ABS D 8.4 ZB .75
N092 M30
END MIN: .8 FT: 20.9 MTR: 6.3
```

TOOL MOTION
Return to Home Position

Figure 10.24 *(Continued)*

```
N001 X-01.825 Z-00.6352 F200.00 S0298 T0001 M03
```

Rough Turning Tool

```
N002 X-00.1 Z-00.0348 F010.00
N003 Z-05.045 F004.47
N004 X 00.255 F004.04 S0269
N005 Z 04.945 F200.00
N006 Z 00.1 F010.00
N007 X-00.41 F200.00 S0336
N008 X-00.1 F010.00
N009 Z-05.045 F00
```

Figure 10.25 *Machine tool output from Compact II program.*


```
N010 X 00.255 F004.49 S0299
N011 Z 04.945 F200.00
N012 Z 00.1 F010.00
N013 X-00.41 F200.00 S0386
N014 X-00.1 F010.00
N015 Z-03.6317 F005.79
N016 X 00.1428 Z-00.3924 F005.37 S0358
N017 G02 X 00.0172 Z-00.0974 I00.2678 K00.0974 F005.33 S0355
N018 Z-00.9235 F005.30 S0353
N019 X 00.095 F005.07 S0338
N020 Z 04.945 F200.00
N021 Z 00.1 F010.00
N022 X-00.41 F200.00 S0453
N023 X-00.1 F010.00
N024 Z-02.9311 F006.80
N025 X 00.255 Z-00.7006 F005.82 S0388
N026 Z 03.5317 F200.00
N027 Z 00.1 F010.00
N028 X-00.41 F200.00 S0547
N029 X-00.1 F010.00
N030 Z-02.2305 F008.21
N031 X 00.255 Z-00.7006 F006.84 S0456
N032 Z 02.8311 F200.00
N033 Z 00.1 F010.00
N034 X-00.41 F200.00 S0692
N035 X-00.1 F010.00
N036 Z-01.5298 F010.38
N037 X 00.255 Z-00.7006 F008.28 S0552
N038 Z 02.1304 F200.00
N039 Z 00.1 F010.00
N040 X-00.41 F200.00 S0941
N041 X-00.1 F010.00
N042 Z-00.073 F014.12
N043 X 00.055 Z-00.055 F013.28 S0885
N044 Z-01.167 F013.10 S0873
N045 X 00.1145 F011.52 S0768
N046 X 00.0855 Z-00.2348 F010.50 S0700
N047 Z 01.4298 F200.00
N048 Z 00.1 F010.00
N049 X-00.2125 F200.00 S1024
N050 X-00.1 F010.00
N051 Z-00.0155 F015.36
N052 X 00.0575 Z-00.0575 F014.33 S0955
```

Figure 10.25 *(Continued)*

```
N053 X 03.355 Z 00.7215 F200.00
N054 X 00.1 Z 00.0215 F010.00
N055 X-03.4125 Z-00.6509 F200.00 S1307 T0002      Finish Turning Tool
N056 X-00.1 Z-00.0191 F010.00
N057 Z-00.0375 F010.46
N058 X 00.0925 Z-00.0925 F009.29 S1161
N059 Z-01.17 F009.16 S1145
N060 X 00.116 F008.03 S1004
N061 X 00.2754 Z-00.7545 F006.02 S0753
N062 X 00.3798 Z-01.0447 F004.52 S0565
N063 X 00.3369 Z-00.9266 F003.72 S0465
N064 G02 X 00.0169 Z-00.0957 I00.2631 K00.0957 F003.68 S0460
N065 Z-00.9285 F003.66 S0458
N066 X 00.625 F002.76 S0345
N067 X 01.6408 Z 05.62 F200.00
N068 X 00.0292 Z 00.1 F010.00
N069 X-03.4 Z-00.55 F200.00 S0250 T0003          Threading Tool
N070 X-00.0695 Z-00.0401 F002.50
N071 G33 Z-01.1599 K0.1
N072 X 00.0695 F200.00
N073 Z 01.2
N074 X-00.0854 Z-00.0493 F002.50
N075 G33 Z-01.1507 K0.1
N076 X 00.0854 F200.00
N077 Z 01.2
N078 X-00.0976 Z-00.0564 F002.58
N079 G33 Z-01.1436 K0.1
N080 X 00.0976 F200.00
N081 Z 01.2
N082 X-00.1063 Z-00.0614 F002.50
N083 G33 Z-01.1386 K0.1
N084 X 00.1063 F200.00
N085 Z 01.2
N086 X-00.1113 Z-00.0642 F002.50
N087 G33 Z-01.1358 K0.1
N088 X 00.1113 F200.00
N089 Z 01.2
N090 X 03.4 Z 00.55
N091 M05
N092 M30                                          End of Program Rewind
```

Figure 10.25 *(Continued)*

APT PROGRAM

As a comparison of the many computer languages besides Compact II, APT (Automatically Programmed Tools) was selected. APT was selected because it is found in many similar forms such as ADAPT, UNIAPT, and AUTOSPOT. APT has played a large role in the computer-assisted programming language market. Many other languages and CAD/CAM systems grew out of this language. This language was developed and is still heavily used in the aerospace industry due to its ability to handle multiaxis contour programming to the fifth and seventh axis level. Other languages like Compact II were only developed to do contouring moves to three axes deep, after which additional axes became positional only. It is because of this that the author feels many CAD/CAM systems have patterned themselves after the APT language. A comparison of actual computer statements generated on a CAD/CAM system will easily show this. As the APT program is reviewed, refer to the Compact II program to see the minor differences in program statements and structures.

The part to be reviewed is a simple milled part in which the periphery and a pass around the top of the part will be machined (see Figure 10.26). As in the Compact II program the part print must be reviewed, a machine selected, and setup determined before programming can start.

The machine zero is indicated on Figure 10.26 to be 2.0 in. to the left of the finish part profile on the *X* axis. The *Y* axis zero is 2.0 in. below the profile and *Z* zero is 1.0 in. beneath the part. To keep the process simple, it has been decided to machine one path around the periphery at full depth, and then to

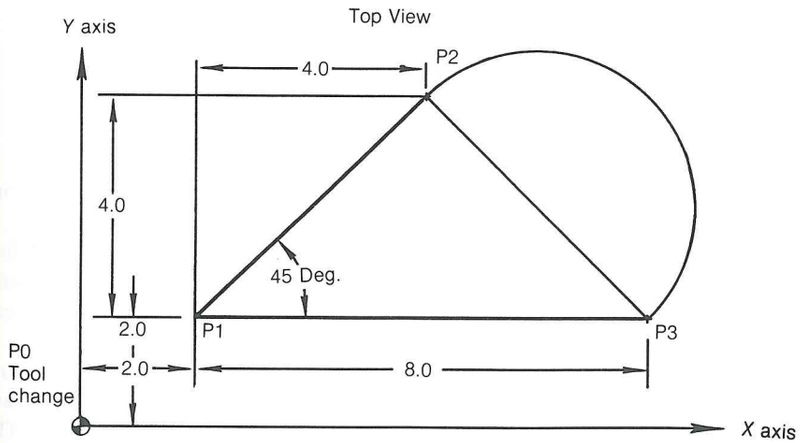
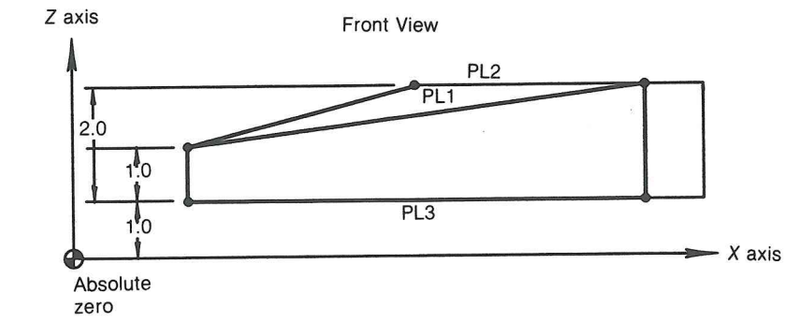


Figure 10.26 *APT mill program example.*



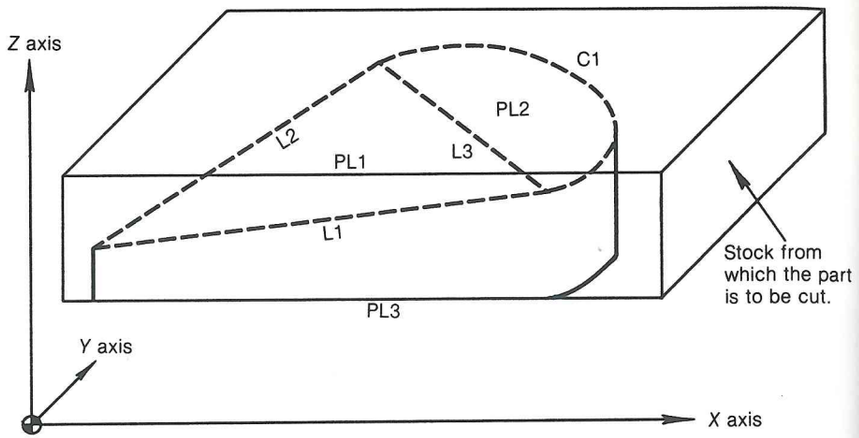


Figure 10.26 *(Continued)*

take one pass around the top surface. The programmer has selected a 1.0 in. diameter cutter with a 0.250 in. corner radius.

The next step in the process is to mark the geometry on the part print that needs to be defined by the programmer. Figure 10.26 shows the indicated points "P," lines "L," circles "C," and planes "PL" necessary to describe the piece part.

Once the needed geometry is determined, programming can begin. Figure 10.27 shows one programmer's solution to this particular problem. For easy reference, the program has been marked up to show its major sections. The following pages give a line by line description of the various commands in the UNIAPT program.


```
                                                    Identification statement*
                                                    Program statement
                                                    Printer statements
                                                    Point definition
                                                    Line definition
                                                    Plane definitions
                                                    Circle definition
                                                    Program statement
                                                    Cutter definition
2466 ↑LU,
OPT=4
INITIALIZATION  001 PARTNO  UNIAPT SAMPLE PART PROGRAM
Statements      002 $$  GEOMETRY
                003     CLPRNT
                004     PRINT/ON
GEOMETRY        005 P0=POINT/0,0
Statements      006 P1=POINT/2,2,2
                007 P3=POINT/10,2,3
                010 L1=LINE/P1,P3
                011 L2=LINE/P1,ATANGL,45
                012 L3=LINE/(P2=POINT/6,6,3),P3
                013 PL1=PLANE/P1,P2,P3  $$ CANTED PLANE ON TOP
                014 PL2=PLANE/0,0,1,3  $$ PLANE ,Z=3
                015 PL3=PLANE/PARLEL,PL2,ZSMALL,2  $$ Z=1
                016 R =(LNTHF(VECTOR/P2,P3))/2
                017 C1 =CIRCLE/TANTO,L2,XSMALL, P3,RADIUS,R
                020 $$
                021 $$      MAKE PASS AROUND PERIPHERY
                022 $$
TOOL            023     CUTTER      /1,.25 $$DIA.=1,CORNER RADIUS=.25
CHANGE          024     FROM        /P0
Statements      025     DNTCUT
```

Figure 10.27 *Source program listing—UNIAPT Milling Example. (*Note: 001 becomes a MACHIN/statement when programming for a specific machine tool.)*

```
            026      GOTO        /-.75,.75,1 $$ MOVE CLEAR OF PART
            027      GO          /L2,PL3,ON,L1,10 $$10 IPM FEEDRATE
            030      CUT                                                Tool motion for
            031  TLLFT,GOLFT     /L2                                    periphery cut
            032      GOFWD       /C1,PAST,L1
            033      GORGT       /L1,PAST,L2
TOOL        034      GOTO        /P0
MOTION      035 $$
Statements  036 $$   MAKE A PASS AROUND TOP OF PART                     Program statement
            037 $$
            040      GODLTA      /0,0,2
            041      GO          /ON,L2,PL1,L1,20 $$ FEEDRATE=20 IPM
            042  TLON,GOFWD      /L2,PAST,PL2
            043      PSIS        /PL2 $$ AUTOPS WOULD BE THE SAME       Tool motion for
            044      GOFWD       /L2,TANTO,C1                           top edge
            045      GOFWD       /C1,ON,L1
            046      GO          /ON,L1,PL1,ON,L3 $$GET BACK ON PL1
            047      INDIRV      /-1,0,0 $$ESTABLISH DIRECTION
            050      GOFWD       /L1,PAST,L2
            051      GOTO        /P0
TERMINATION 052      STOP
Statements  053      END
            054      FINI
```

Figure 10.27 *(Continued)*

001 PARTNO UNIAPT SAMPLE PROGRAM PART

Identifies the part and serves as a title for the program listing as well as identification for the CNC control tape.

002 $$ GEOMETRY

The double dollar sign indicates a comment line.

003 CLPRNT

Specifies that the cutter location input data to the postprocessor is to be printed. This statement caused the X, Y, Z outputs in the following cutter location data file "CL Data" from program line numbers 24 through 51.

004 PRINT/ON

Causes surface data to be printed in its canonical form. This statement caused the output from program line numbers 005 through 17 in the CL Data file.

005 P0 = POINT/0,0

Assigns the name P0 to the point whose coordinates are $x = 0$, $y = 0$. The processor assumes $z = 0$ if only two points are given and no ZSURF statement has been given.

006 P1 = POINT/2,2,2

Assigns the name P1 to the point whose coordinates are $x = 2$, $y = 2$, $z = 2$.

007 P3 = POINT/10,2,2

Assigns the name P3 to the point whose coordinates are $x = 10$, $y = 2$, $z = 2$.

***010 L1 = LINE/P1,P3**

Defines a line through the two points P1 and P3.

011 L2 = LINE/P1, ATANGL, 45

Defines a line through the point P1 and at an angle of 45° to the X axis.

012 L3 = LINE/(P2 = POINT/6,6,3), P3

Define a line through two points P2 and P3. P2 is an example of a nested definition, i.e., it defines P2 in terms of its coordinates ($x = 6$, $y = 6$,

---

*Although called a line these are actually surfaces with an infinite plane perpendicular to the X–Y plane of the paper. They are called lines for convenience. It is always understood that a line in UNIAPT is really a plane. The same is true for a circle. It is actually an infinite cylinder whose sides are perpendicular to the XY, YZ, or XZ plane.

$z = 3$) within the line statement. P2 could have been written as a separate statement in which case the preceding statement would have appeared simply as L3 = LINE/P2, P3.

013 PL1 = PLANE/P1,P2,P3 $$ CANTED PLANE ON TOP

Defines a canted plane (which is the top of the part) in terms of three points: P1,P2,P3. The words following the double dollar sign are not processed by the program. They are for programmer documentation only.

014 PL2 = PLANE/0,0,1,3 $$ PLANE, Z = 3

Defines a plane in terms of the coefficients of the plane equation $ax + by + cz - d = 0$, where $a = 0$, $b = 0$, $c = 1$, $d = 3$.

015 PL3 = PLANE/PARLEL,PL2,ZSMALL,2 $$ Z = 1

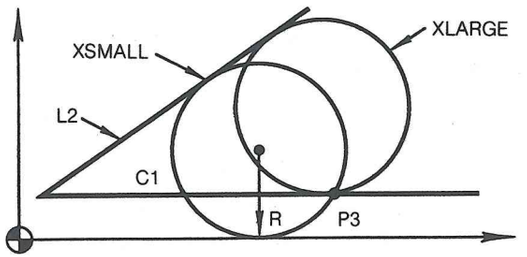
Defines a plane parallel (PARLEL) to plane (PL2) 2 units away from PL2. Two planes can meet this definition—one above and one below PL2. The modifier ZSMALL removes the ambiguity by specifying the plane with the smaller Z value: below PL2.

016 R = (LNTHF(VECTOR/P2,P3))/2

Defines the radius (R) by computing the length (LNTHF) of the vector (VECTOR) which connects points P2 and P3 and taking one-half of the resulting value.

017 C1 = CIRCLE/TANTO,L2,XSMALL,P3,RADIUS,R

Defines the circle (C1) in terms of line (L2) that it (C1) is tangent to and the point P3 that it passes through with a radius R. As shown in the following schematic, two circles can meet these requirements. The modifier XSMALL indicates to the system to use the one with the smallest X value.



020 $$

Causes a line to be skipped.

021 $$ MAKE A PASS AROUND PERIPHERY

A comment.

022 $$

Causes a line to be skipped.

023 CUTTER/1,.25 $$ DIA. = 1, CORNER RADIUS = .25

Defines cutter diameter (1 in.) and corner radius (0.25 in.).

024 FROM/P0

Defines cutting tool beginning position as being at point (P0). To see tool path generated by computer statement refer to Figure 10.28.

025 DNTCUT

Indicates that tool center information for motion statements is not to be passed on to the postprocessor.

026 GOTO/−.75,.75,1 $$ MOVE CLEAR OF PART

Move cutter (clear of part) tool to $x = -0.75$, $y = 0.75$, and $z = 1$.

027 GO/L2,PL3,ON,L1,10 $$ 10 IPM FEEDRATE

Move cutting tool to the position where L2, PL3, and L1 intersect. PL3 is established as the part surface (the second surface in the command) for all subsequent motion commands. This command also sets the feedrate to 10 ipm.

030 CUT

Indicates that tool center information for motion statements is to be passed on to the postprocessor. It causes cutter location data output to begin with preceding statement.

031 TLLFT, GOLFT/L2

Position the tool to left (TLLFT) of L2 and move it along the intersection of surfaces L2 and PL3 to C1. A check surface is implied, which means to use the drive surface from the next command for the check surface for this command. Also a TANTO condition is indirectly specified since the next command is a GOFWD (*drive surface:* surface which tool is moved along; *check surface:* surface which tool will stop at.)

032 GOFWD/C1,PAST,L1

Move tool along intersection of surfaces C1 and PL3 until past the plane L1.

033 GORGT/L1,PAST,L2

Turn right and move tool along intersection of L1 and PL3 past the plane L2.

034 GOTO/P0

Move tool from its present position to P0 ($x = 0$, $y = 0$, $z = 0$).

035 $$

A remark—causes a line space.

036 $$ MAKE A PASS AROUND TOP OF PART

A remark. Notes that the next group of statements will begin machining the part surface top.

037 $$

A line space.

040 GODLTA/0,0,2

Adds the indicated increments to the tool position. Since x and y are zero, tool is moved straight up 2 in. To see tool path generated by computer statement, refer to Figure 10.29.

041 GO/ON,L2,PL1,L1,20 $$ FEEDRATE = 20 IPM

Position the tool at the intersection of drive surface L2, part surface PL1, and check surface L1. Tool end is centered on the L2 surface and to the PL1 and L1 surfaces. Feedrate is set at 20 ipm. PL1, since it is the second surface in the GO/ command, is established as the modal part surface.

042 TLON, GOFWD/L2,PL2

Tool is centered on (TLON) the drive surface which is L2 in this statement example. Go forward (GOFWD) along the intersection of L2 and PL1, which is the part surface. Stop at PL2, which is the check surface.

043 PSIS/PL2 $$ AUTOPS WOULD BE THE SAME

Causes part surface to be changed to PL2. PSIS means "part surface is." The comment refers to the command AUTOPS, which can be used to establish a plane at the cutters current z height.

044 GOFWD/L2,TANTO,C1

Move tool along the intersection of L2 and PL2 until tangent to circle C1.

045 GOFWD/C1,ON,L1

Move tool along the intersection of C1 and PL2. Stop on surface L1.

046 GO/ON,L1,PL1,ON,L3 $$ GET BACK ON PL1

Move tool to the position where L1, PL1, and L3 intersect. The "ON" modifier means the tool center is to be positioned "on" L1 and "on" L3.

047 INDIRV/−1,0,0 $$ ESTABLISH DIRECTION

This command, in the direction of vector, establishes a tool direction as being in the minus x direction ($x = -1$, $y = 0$, $z = 0$).

050 GOFWD/L1,PAST,L2

Go forward along the intersection of L1 and PL1 and stop just past the surface L2.

051 GOTO/P0

A point to point command. Position tool from where it is to point P0.

052 STOP

A postprocessor command that stops the machine tool and machine tool controller input reader. Feedrate is reduced to zero and spindle and coolant are turned off.

053 END

A postprocessor command that signals the end of a logical section of a part program.

054 FINI

The end of the complete part program.

In an APT program's processing, after the programmer creates the computer source program, it is submitted to the computer. The computer upon receipt of source information checks to see if it is a valid statement. Once validity is established, the computer creates the cutter location file data either by interaction or through batch processing. The CL Data file consists of all the geometry and cutter location information required to machine the part in coordinate numerical form. This information can then be transformed into the correct numerical control format for the particular machine tool and CNC control unit. The information in Figure 10.30 is a printout of the CL Data from the UNIAPT sample program. Note: For the readers benefit, the axis designators are shown in parentheses.

After computer processing has occurred to the point of the CL Data file just reviewed, the programmer can then request tool path graphics from the system.

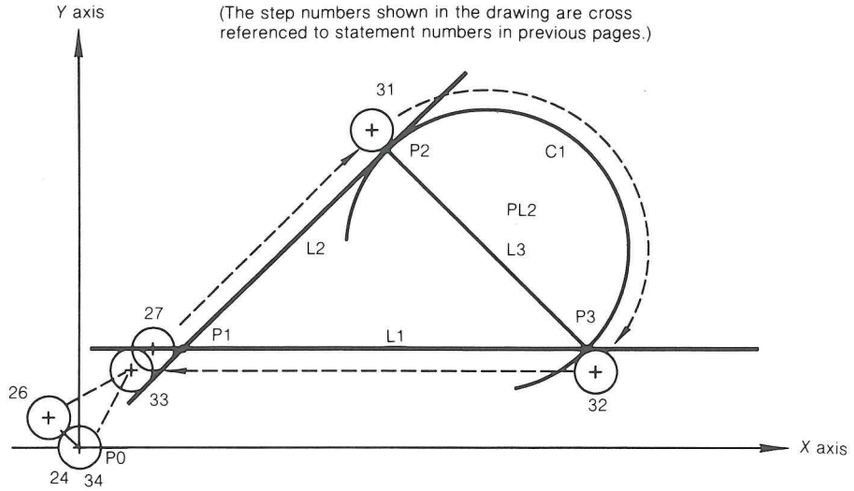


Figure 10.28 *Top view showing path of cutter, which makes a pass around periphery. (The step numbers shown in the drawing are cross referenced to statement numbers in previous pages.).*

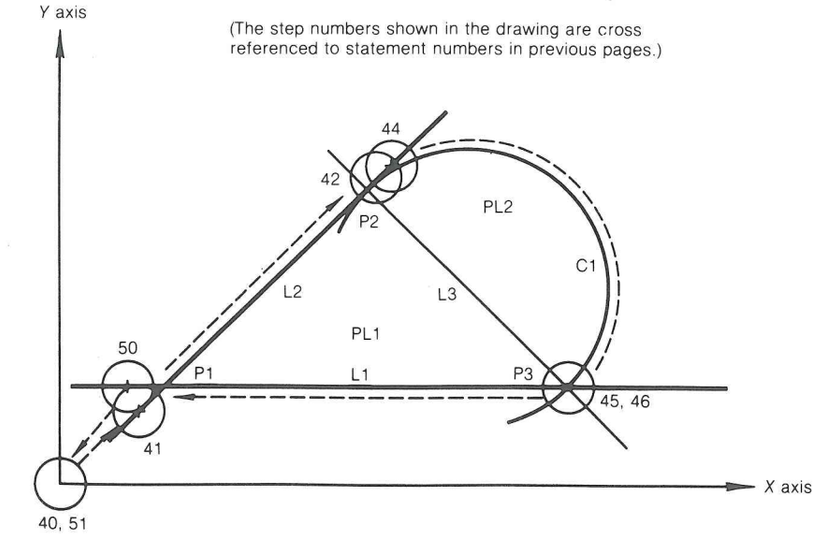


Figure 10.29 *Top view showing path of cutter, which makes a pass around top of part. (The step numbers shown in the drawing are cross referenced to statement numbers in previous pages.).*


```
Source Program Line #
REC0005 SURFACE=POINT   P0
  (X)  .000000     (Y)  .000000     (Z) .000000
REC0006 SURFACE=POINT   P1
  (X) 2.000000     (Y) 2.000000     (Z)2.000000
REC0007 SURFACE=POINT   P3
  (X)10.000000     (Y) 2.000000     (Z)3.000000
REC0010 SURFACE=LINE   L1
  (X)  .000000     (Y) 1.000000     (X) .000000      (Y)2.000000
REC0011 SURFACE=LINE   L2
  (X)  .707106     (Y)-.707106      (X) .000000      (Y) .000000
REC0012 SURFACE=POINT   P2
  (X) 6.000000     (Y) 6.000000     (Z)3.000000
REC0012 SURFACE=LINE   L3
  (X)  .707106     (Y)  .707106     (X) .000000      (Y)8.485282
REC0013 SURFACE=PLANE   PL1
  (X) -.123091     (Y)-.123091      (Z) .984731          1.477097
(X, Y, Z values of unit vector perpendicular to plane, fourth value is
minimum distance of plane from origin.)
REC0014 SURFACE=PLANE   PL2
  (X)  .000000     (Y)  .000000     (Z)1.000000      (Z)3.000000
(First three values establish plane vector.)
REC0015 SURFACE=PLANE   PL3
  (X)  .000000     (Y)  .000000     (Z)1.000000      (Z)1.000000
REC0016 SURFACE=VECTOR
      4.000000      -4.000000          .000000
REC0016 SURFACE=SCALAR   R
      2.828427
REC0017 SURFACE=CIRCLE   C1
  (X) 8.000000     (Y) 3.999999     (Z) .000000     (X1) .000000
  (Y1) .000000   (Z1) 1.000000   (Radius) 2.828427
(X1, Y1, Z1 form unit vector of cylinder passing through circle.)
REC0024
  (X)  .000000     (Y)  .000000     (Z) .000000
REC0030
  (X) 1.292893     (Y) 2.000000     (Z)1.000000
REC0031
  (X) 5.646446     (Y) 6.353553     (Z)1.000000
REC0032
  (X) 5.687013     (Y) 6.394128     (Z) 1.000000
      5.770902         6.472423         1.000000
      5.857440         6.547782         1.000000
```

Figure 10.30 *Program output; surface data in canonical form and cutter location data. (*Note: *Axis call labels will not be printed on computer generated readouts.).*

5.946523	6.620113	1.000000
6.038047	6.689330	1.000000
6.131902	6.755352	1.000000
6.227977	6.818100	1.000000
6.326156	6.877500	1.000000
6.426326	6.933480	1.000000
6.528365	6.985975	1.000000
6.632152	7.034921	1.000000
6.737565	7.080262	1.000000
6.844478	7.121942	1.000000
6.952764	7.159913	1.000000
7.062294	7.194129	1.000000
7.172939	7.224550	1.000000
7.284566	7.251140	1.000000
7.397043	7.273866	1.000000
7.510238	7.292701	1.000000
7.624013	7.307626	1.000000
7.738235	7.318619	1.000000
7.852769	7.325669	1.000000
7.967477	7.328768	1.000000
8.082225	7.327911	1.000000
8.196873	7.323100	1.000000
8.311290	7.314340	1.000000
8.425335	7.301643	1.000000
8.538875	7.285022	1.000000
8.651775	7.264497	1.000000
8.763900	7.240094	1.000000
8.875118	7.211841	1.000000
8.985297	7.179771	1.000000
9.094303	7.143923	1.000000
9.202010	7.104339	1.000000
9.308289	7.061067	1.000000
9.413013	7.014158	1.000000
9.516059	6.963666	1.000000
9.617302	6.909654	1.000000
9.716624	6.852184	1.000000
9.813907	6.791326	1.000000
9.909034	6.727149	1.000000
10.001891	6.659734	1.000000
10.092372	6.589157	1.000000
10.180365	6.515504	1.000000
10.265769	6.438862	1.000000
10.348479	6.359323	1.000000

Note: Amount of output will depend on path tolerance setup in computer.

Figure 10.30 *(Continued)*

10.428398	6.276979	1.000000
10.505432	6.191930	1.000000
10.579490	6.104276	1.000000
10.650482	6.014123	1.000000
10.718326	5.921575	1.000000
10.782939	5.826745	1.000000
10.844246	5.729745	1.000000
10.902172	5.630689	1.000000
10.956650	5.529695	1.000000
11.007617	5.426883	1.000000
11.055007	5.322376	1.000000
11.098769	5.216298	1.000000
11.138848	5.108775	1.000000
11.175198	4.999934	1.000000
11.207775	4.889905	1.000000
11.236541	4.778819	1.000000
11.261459	4.666807	1.000000
11.282504	4.554002	1.000000
11.299648	4.440540	1.000000
11.312870	4.326555	1.000000
11.322158	4.212180	1.000000
11.327496	4.097554	1.000000
11.328883	3.982812	1.000000
11.326312	3.868091	1.000000
11.319790	3.753526	1.000000
11.309322	3.639254	1.000000
11.294922	3.525411	1.000000
11.276608	3.412132	1.000000
11.254401	3.299551	1.000000
11.228325	3.187802	1.000000
11.198414	3.077018	1.000000
11.164703	2.967332	1.000000
11.127232	2.858872	1.000000
11.086044	2.751768	1.000000
11.041190	2.646148	1.000000
10.992722	2.542136	1.000000
10.940697	2.439856	1.000000
10.885180	2.339430	1.000000
10.826233	2.240978	1.000000
10.763928	2.144615	1.000000
10.698340	2.050457	1.000000
10.629544	1.958615	1.000000
10.557624	1.869199	1.000000

Figure 10.30 *(Continued)*

```
      10.482666      1.782316      1.000000
      10.404758      1.698067      1.000000
      10.323992      1.616553      1.000000
      10.240463      1.537871      1.000000
      10.197368      1.500000      1.000000
REC0033
   (X) .792893   (Y) 1.500000   (Z) 1.000000
REC0034
   (X) .000000   (Y) .000000    (Z) .000000
REC0040
   (X) .000000   (Y) .000000    (Z) 2.000000
REC0041
  (X) 1.500000   (Y) 1.500000   (Z) 1.923070
REC0042
   (X)5.807718   (Y)5.807718    (Z)2.999999
REC0044
   (X)6.000000   (Y)5.999999    (Z)2.999999
REC0045
   (X)6.037401   (Y)6.037408    (Z)2.999999
      6.114954      6.109367      2.999999
      6.195144      6.178376      2.999999
      6.277858      6.244339      2.999999
      6.362981      6.307162      2.999999
      6.450393      6.366759      2.999999
      6.539973      6.423045      2.999999
      6.631595      6.475944      2.999999
      6.725130      6.525378      2.999999
      6.820449      6.571281      2.999999
      6.917417      6.613588      2.999999
      7.015899      6.652239      2.999999
      7.115758      6.687181      2.999999
      7.216853      6.718365      2.999999
      7.319044      6.745746      2.999999
      7.422187      6.769288      2.999999
      7.526138      6.788957      2.999999
      7.630752      6.804725      2.999999
      7.735883      6.816570      2.999999
      7.841382      6.824476      2.999999
      7.947103      6.828432      2.999999
      8.052899      6.828432      2.999999
      8.158621      6.824476      2.999999
      8.264121      6.816570      2.999999
      8.369251      6.804725      2.999999
```

Figure 10.30 *(Continued)*


```
      8.473865      6.788957      2.999999
      8.577816      6.769288      2.999999
      8.680958      6.745746      2.999999
      8.783149      6.718364      2.999999
      8.884245      6.687180      2.999999
      8.984103      6.652238      2.999999
      9.082585      6.613586      2.999999
      9.179553      6.571279      2.999999
      9.274872      6.525377      2.999999
      9.368407      6.475942      2.999999
      9.460029      6.423044      2.999999
      9.549609      6.366757      2.999999
      9.637020      6.307160      2.999999
      9.722144      6.244336      2.999999
      9.804858      6.178374      2.999999
      9.885048      6.109365      2.999999
      9.962601      6.037406      2.999999
     10.037410      5.962597      2.999999
     10.109370      5.885043      2.999999
     10.178378      5.804853      2.999999
     10.244339      5.722139      2.999999
     10.307163      5.637016      2.999999
     10.366760      5.549604      2.999999
     10.423046      5.460023      2.999999
     10.475944      5.368402      2.999999
     10.525379      5.274866      2.999999
     10.571282      5.179548      2.999999
     10.613589      5.082580      2.999999
     10.652240      4.984098      2.999999
     10.687182      4.884239      2.999999
     10.718365      4.783143      2.999999
     10.745747      4.680953      2.999999
     10.769290      4.577810      2.999999
     10.788958      4.473859      2.999999
     10.804725      4.369245      2.999999
     10.816570      4.264114      2.999999
     10.824477      4.158614      2.999999
     10.828433      4.052892      2.999999
     10.828433      3.947097      2.999999
     10.824477      3.841376      2.999999
     10.816570      3.735876      2.999999
     10.804725      3.630746      2.999999
     10.788956      3.526132      2.999999
```

Figure 10.30 *(Continued)*

```
      10.769287        3.422181        2.999999
      10.745746        3.319038        2.999999
      10.718364        3.216847        2.999999
      10.687179        3.115752        2.999999
      10.652237        3.015893        2.999999
      10.613586        2.917411        2.999999
      10.571278        2.820443        2.999999
      10.525375        2.725125        2.999999
      10.475940        2.631589        2.999999
      10.423042        2.539968        2.999999
      10.366756        2.450388        2.999999
      10.307159        2.362976        2.999999
      10.244335        2.277853        2.999999
      10.178372        2.195139        2.999999
      10.109363        2.114950        2.999999
      10.037403        2.037396        2.999999
      10.000000        2.000000        2.999999
REC0046
  (X)10.000000     (Y)2.000000     (Z)3.048070
REC0050
    (X)1.292893     (Y)2.000000     (Z)1.959681
REC0051
     (X).000000      (Y).000000      (Z).000000
OPT=
```

Figure 10.30 *(Continued)*

Graphics such as those in Figures 10.28 and 10.29 will either be drawn on a plotter or displayed on a CRT screen depending on the system. Note though that the computer system does not normally indicate cross reference numbers of the program statements.

The next computer processing step will be for the CL Data to be postprocessed into a machine control program file. During postprocessing the computer converts the CL Data into the proper format to be understood by the machine tool control unit. At this time the postprocessor also adds additional function codes to specify types of motion, speeds, and feeds and auxiliary codes for turning program options on and off. Figure 10.31 shows a typical machine tool output or tape file generated by the postprocessor for our UNIAPT sample program.

GRAPHICS-BASED SYSTEMS

Graphics-based systems are referred to as Graphical Numerical Control (GNC). They differ from language-based systems in the following manner.


```
N010 G00 X0 Y0 Z10 S400 M03          Check home position P0
                                       start spindle
N020 X-.75 Y.75 Z1 M08          Move clear of part, set Z depth
N030 G01 X1.2929 Y2.000 F10.0   Feed to Ref. 27, Figure 10.10
N040 X5.6464 Y6.3536                    Feed to Ref. 31
N050 G02 X8.0 Y7.3284 I2.3536 J.9748    Cut circle 1
N060 X11.3284 Y4.00 I0 J3.3284
N070 X10.1974 Y1.5 I3.3284 J0           Cut circle to Ref. 32
N080 G01 X.7929                         Feed to Ref. 33
N090 G00 Z3.1 M09               Retract Z, turn coolant off
N100 X0 Y0 Z10                  Rapid to P0 Ref. 34
N110 Z2.0                       Rapid position Z Ref. 40
N120 G01 X1.5 Y1.5 Z1.9231 F20 M08      Feed to Ref. 41
                                       turn coolant on
N130 X5.8077 Y5.8077 Z3.0               Feed to Ref. 42
N140 X6.0 Y6.0                          Feed to Ref. 44
N150 G02 X8.00 Y6.8284 I2.0 J.8284      Cut circle 1 top
N160 X10.8284 Y4.0 I0 J2.8284           Cut circle 1
N170 X10.0 Y2.0 I2.8284 J0              Cut circle 1 Ref. 45
N180 G01 Z3.0481                Position Z axis Ref. 46
N190 X1.2929 Z1.9597            Feed to position Ref. 50
N200 G00 Z3.1 M09     Retract Z position Ref. 50, turn coolant off
N210 X0 Y0 Z10 M05    Rapid to home position P0, turn spindle off
N220 M30                                End of program
```

Figure 10.31 *Postprocessor output UNIAPT sample program.*

When the geometric detail of the component has been constructed on the CRT screen, the outline shape of the cutting tool, or tools, to be used are superimposed on the component image and can be freely moved around using the cursor control device. Thus the programmer can, in effect, select appropriate tool paths to facilitate machining of the component or component detail. A better impression of the process may be obtained by considering the following example.

Figure 10.32 illustrates a component detail as it might be "drawn" on the computer screen. Also shown are two different-sized circles representing the superimposed cutters required to machine the outside profile and clear the inner shape, which represents a pocket.

Consider the large-diameter cutter first. This cutter is to be used to machine the outside profile. The size and type of cutter will have been established already and entered into the data file, possibly via a keyboard entry. The programmer now has free control to move the cutter to any position he or she chooses.

The cutter is first positioned in a suitable starting position. If cutter radius

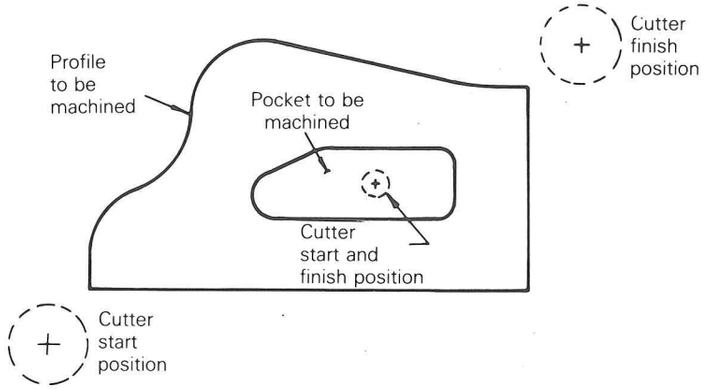


Figure 10.32 *A graphical numerical control display.*

compensation is required, then the starting position will be a suitable distance away from the profile so that the cutter can attain the correct position before contacting the work surface. This start position is then entered into the data file either via the cursor control or the keyboard.

Having established the start position the programmer moves the cutter image to the finish position, and this too is entered into the file.

Using this very simple data entry the computer is able to compute the complete cutter route, indicating it on the screen alongside the component. It will also list, in numerical form, the cutter location in relation to a previously established datum at each stage in the machining process; the list appears on a second CRT screen or an interfaced printer.

Now consider the machining of the pocket. To achieve this the programmer simply has to move the cutter to the start position at the center of the pocket as indicated. All that is now needed are keyboard data entries to establish that a pocket clearance routine is required. From this data entry the computer will establish all the moves necessary to clear the pocket and machine the final profile. Again, the cutter paths will be shown graphically and also presented in numerical form, the dimensional values being in relation to a predetermined datum.

Note that in this simple description no mention has been made of movement in the third, that is, Z axis. Clearance in the Z axis, and the depth to which the cutter is required to go, will be entered via the keyboard as the program is constructed. Data relating to speeds and feeds will be entered in the same way.

In addition to the component image displayed on the screen, it is possible to include any workholding or tool-holding arrangements that may influence the choice of tool paths. An example of such a display for a turning operation is shown in Figure 10.33. By including such detail it is possible for the programmer to be reasonably confident that all programmed moves will be collision-free.

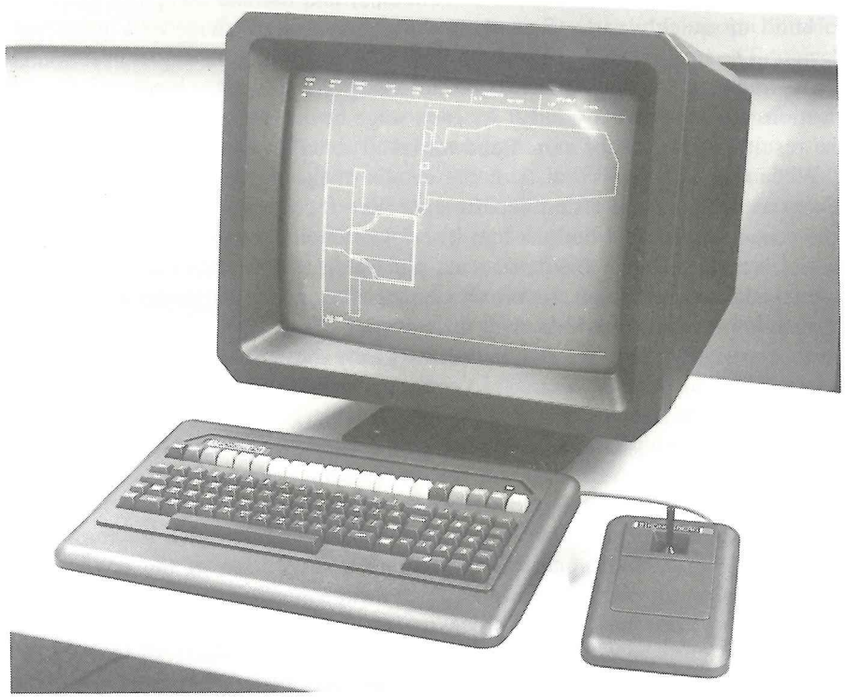


Figure 10.33 *Part programming station showing use of computer graphics for program proving.*

As with language-based systems, it will be possible to obtain copies of the data entered via an interfaced printer.

Data can also be reprocessed to verify its accuracy prior to postprocessing into machine control language. Graphics-based systems have also given way to the more powerful CAD/CAM systems that can generate commands through graphics for all axes.

CAD/CAM: COMPUTER-AIDED DESIGN AND MANUFACTURE

The expression "computer-aided design," or more commonly the abbreviation CAD, is the term used to describe the process by which engineering designers use the computer as a creative tool allowing them to produce, evaluate, modify, and finalize their designs. The computer becomes a terminal at which the de-

signer sits to analyze data, make calculations, and use the computer graphics to build up quickly and efficiently a three-dimensional image of a projected design. The image can be rotated and viewed from different angles, sectioned through various planes, stretched, condensed, and generally assessed. Modifications can be made instantly. As each stage of the design process proceeds, the resulting data can be stored and retrieved at will.

When the designer is satisfied, the details of the design can be transferred electronically to the drafting department. Here the draftsman or draftswoman transforms the original designs into a series of engineering drawings which he or she creates on the computer screen, and again at this stage each individual component can be rotated, sectioned, scaled up or down and so on in a further process of evaluation which may or may not result in modifications to the original design. When this task is finished, fully dimensioned drawings can be printed from an interfaced printer or plotter, or alternatively the information can be stored as numerical data for later retrieval. The use of computers as an aid to manufacturing processes is referred to as Computer-Aided Manufacturing (CAM). The CAPP process is part of that general definition. Together CAD and CAM form the basis of Computer-Aided Engineering (CAE).

From the foregoing text the reader will now appreciate that an important element of the CAPP process is the geometric definition of the component detail. It is, of course, also central to any computer-aided drafting process. It is logical therefore that the two processes should be linked. Most CAD/CAM software currently in use provides this facility.

The transfer involves reducing the detail drawing (Figure 10.39) to its basic geometry by removing all dimensions, notes, leader lines etc. This is easily and instantly achieved, since the drawing will have been compiled by the use of overlays or layers, with one layer containing the basic drawing and subsequent layers containing dimensions and other data. Each layer is capable of being displayed independently of the others.

With the drawing reduced to its basic form, the geometry created as part of the drafting process can be used for part programming purposes, thus eliminating the geometry construction stage of the CAPP process, and speeding part programming activity considerably.

Computer-aided manufacturing (CAM) is the term generally used to describe manufacturing processes that are computer controlled. One very important manufacturing process is metal cutting, and the computer involvement in this area of activity has been the subject of this text. Metal cutting is, however, just one type of manufacturing process that is computer controlled. There are many others: welding, flame cutting, presswork, electrodischarge machining, parts assembly, and so on.

All the processes listed previously are truly manufacturing processes, that is, the end result is a component or an assembly of components. But there are a host of other essential functions that play a part in the overall setup. The supply of materials and tooling, part programming, process control, and inspection/quality control, are some functions that are workshop-related. Spreading the net further, there are the financial aspects—marketing, stock control, and distribution, for example—and of course there is the design and drafting process discussed earlier. It is possible for all these functions to be interrelated via computer control into a total computerized manufacturing system referred to as CIM (Computer-Integrated Manufacturing).

Since it is the practical aspects of the system that are most likely to be of interest to the reader, that is, the design and making of the product, the relationships between these two areas are worthy of further comment and are perhaps the key elements in the system. In the past, CAD and CAM, that is the production element of CAM, have developed as two separate activities, with the application of computers to the production process being somewhat ahead of CAD. Increasingly, even in small companies, they are now being seen not as two related functions but as one integrated function. Already the more sophisticated design/drafting systems are linked to the manufacturing process via part-programming facilities. Manufacturing aspects are fully considered at the design stage, and machine-control programs are produced direct from design data rather than from a separate, and therefore error prone, analysis of an already finalized engineering drawing. The process may also, ultimately, eliminate the need for conventional drawings. Totally computerized engineering has arrived, and the rate at which it is implemented, especially in large companies, is likely to be rapid.

CAD/CAM systems like computer-assisted programming systems can be found with many different capabilities running on a multitude of platforms. Capabilities run from low-level two-dimensional to high-level three-dimensional systems. The hardware used ranges from small personal computers through minis and mainframes. All systems, though, will have a keyboard for operator commands, a user interface device (mouse, joystick, puck and tablet, or light pen), graphics screen, and central processing unit (CPU) with an information storage media device like a floppy disk or magnetic tape (Figure 10.34). For discussion purposes in this book we will be looking at a mini-based system that uses a mouse-actived command system displayed on a CRT screen. Figure 10.35 shows the area of the screen where various information will be displayed.

User Interface

CAD/CAM systems normally use some form of standard user interface, which essentially replaces direct keyboard input to an applications program with six types of interactive input:

- interactive headline
- graphic icons
- pop-up menus
- single keyboard input
- pointer strokes
- text macro processor

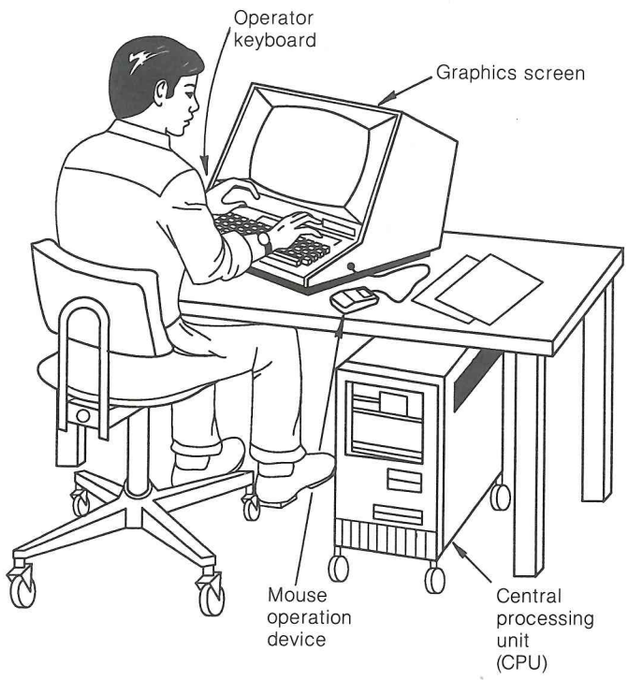


Figure 10.34 *CAD/CAM (Computer-Aided Design/Computer-Aided Manufacturing) workstation.*

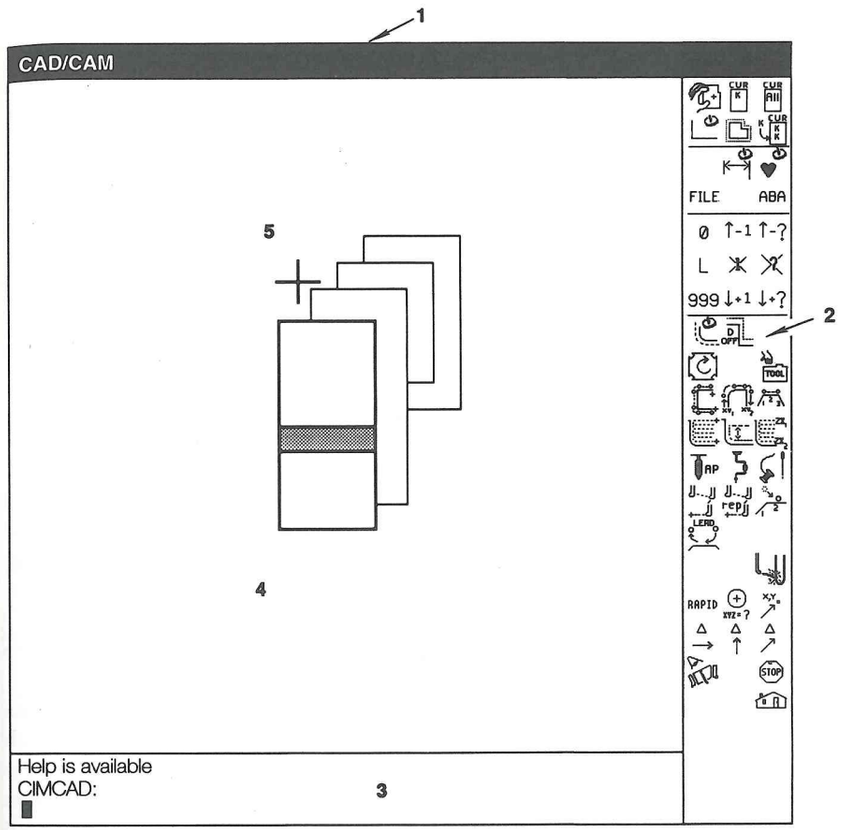
Graphic icons are always visible and reside in a fixed location at the right side of the user CRT screen and are activated through the use of the mouse instead of keying in information. Other systems may use electronic pucks, joysticks, or light pens to select commands from the CRT screen or a tablet. Pop-up menus are a method of command input by which text or symbols representing commands are selected, rather than typing in the command itself. When the *middle mouse button* (MMB) (Fig. 10.37) is pressed, pop-up menus appear on the screen near the screen pointer position. Both menus and icons are user-definable and may be modified at any time during a session. By providing both graphic icons and pop-up menus, the user's focus remains on the screen.

The mouse-controlled screen pointer is used to select items from the pop-up menus and graphic icons, and for input of coordinating information. The information is added by pressing the *left mouse button* (LMB) (Figure 10.36).

Single keystrokes by keyboard input can be used to invoke a particular effect, such as inputting a new geometric point at exactly the same location as an existing point near the screen pointer.

As in all application programs that use various forms of user interface, all menu selections and single-keystroke commands are converted into long com-

CIM CAD USER ENVIRONMENT



1. HEADLINE
2. GRAPHIC ICONS
3. SCROLL AREA
4. GRAPHICS AREA
5. POP-UP MENUS

Figure 10.35 *CAD/CAM work station screen display. (Courtesy of CIMLINC Inc.).*

puter commands for execution. It is these long commands that form the basis of the applications program and out of which any new user menus or single-keystroke commands are built.

Cursor and Mouse Concepts

A cursor is a symbol displayed on the CRT screen. Cursors vary in style depending on which mouse (or activation device) button is pressed and the menu being used. The style of the cursor has a specific meaning. It functions as a pointing device and is positioned by moving the mouse or activation device. Sometimes you use it to point to a location where you wish to start a line or specify the next position on a line. Other times you use it to make a selection from a menu. In many cases you use it to indicate which entity is used to preform a specific function, which is related to the entity near the cursor, by a command chosen from one of the many available menus. *The position of the cursor is very important in these conditions.*

Left Mouse Button (LMB)—GRAPHICS (Working) Cursor The graphics cursor (normally a plus sign) is used as a pointing device to create and modify the entities (lines, text, machining paths, etc.), which define the graphics. It has other uses also, which are addressed further on in this section. The style of the graphic cursor may be changed by the user.

Middle Mouse Button (MMB)—Pop-Up Cursor Pop Up cursors are used to make a selection from the Pop Up menus. Pressing the MMB activates and displays the various Pop Up menus, *only for as long as the MMB is held down.* They appear near the graphics cursor position at the time the MMB is pressed. When it is released, the pop up menus disappear.

On the CIMLINC system shown the TITLES are displayed in inverse video (white letters black background).

As the mouse is moved within the menus, the selection the cursor is on turns to inverse video. When the cursor is positioned on a menu that is underneath another, it instantly pops to the top as shown in Fig. 10.37. In cases where the menu is small, position the cursor in the title area to pop it to the top, then on the desired selection of the menu. Releasing the button on a highlighted command will activate the command without typing.

Right Mouse Button (RMB)—Static Menu (ICON) Cursor The static icons are graphic representations of various computer commands, either CAD or CAM, that can be cursor-selected to keep typing to a minimum. By placing the commands on the computer screen instead of a desk tablet device, the operator's eyes do not have to leave the screen.

The icon cursor is an inverse video of a specific icon (see Figure 10.38). It is used to make selections from the static menu.

When the RMB is pressed, the cursor is displayed instantly in the static menu (icon menu), for as long as pressure is applied. When the RMB is released,

Left Mouse Button (LMB) — GRAPHICS (Working) Cursor

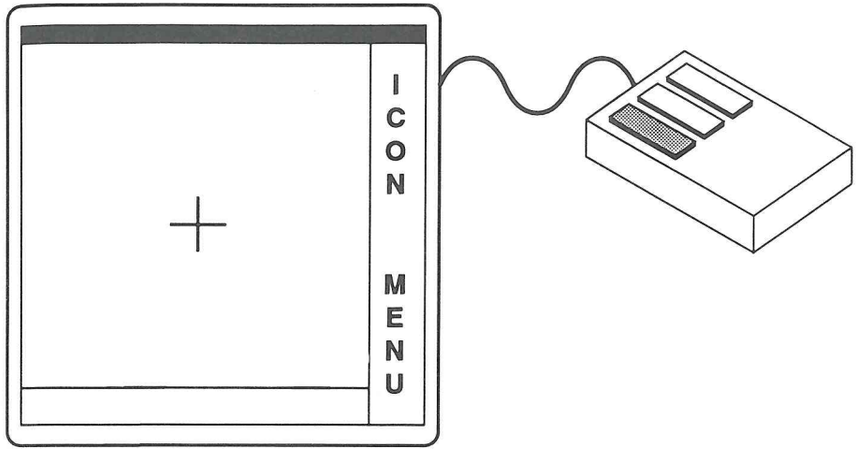


Figure 10.36

Middle Mouse Button (MMB) — Pop-Up cursor

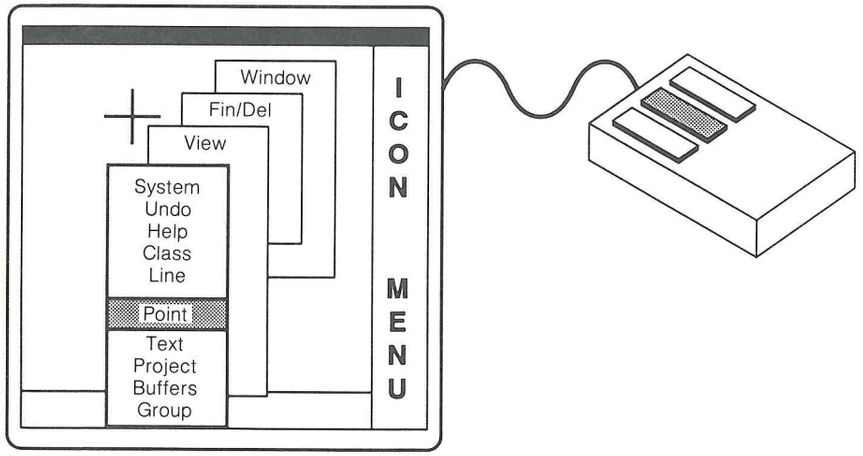


Figure 10.37

Right Mouse Button (RMB) — Static Menu (ICON) Cursor

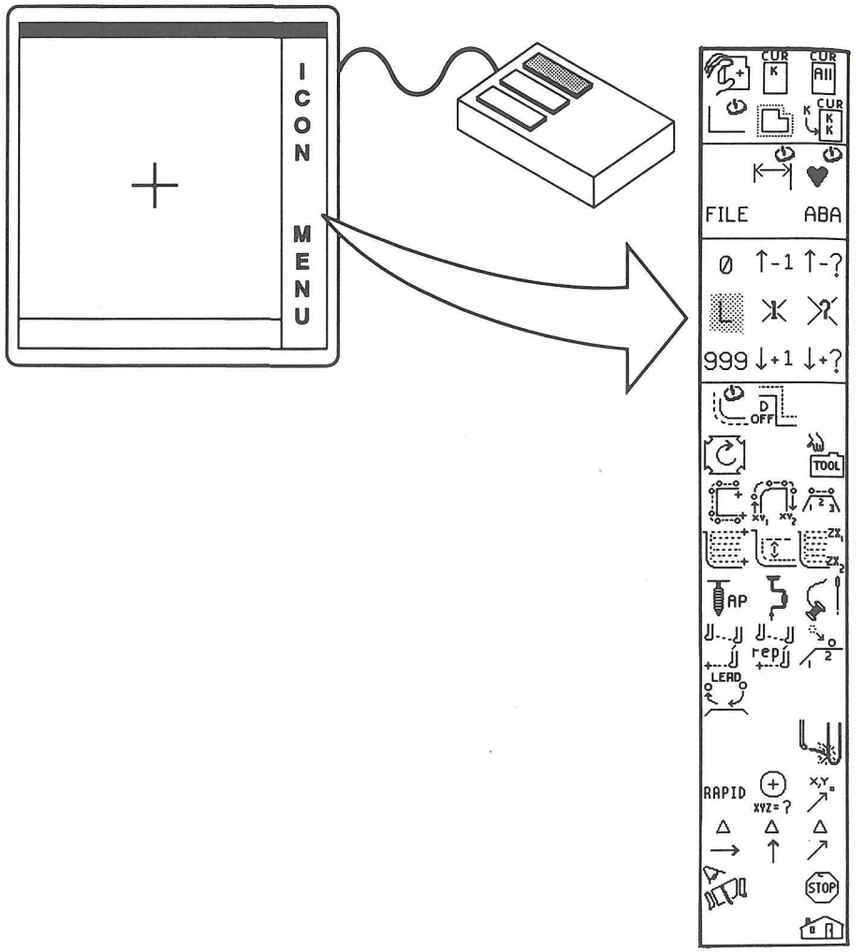


Figure 10.38

the command or action associated to the icon is issued to the CAD/CAM control processor unit and the actual printed command appears at the bottom of the screen.

Now that we have established a general definition of a CAD/CAM system, let us follow the complete process through of generating a program for a part on a CNC lathe. The first step is to generate a CAD design of the part to be manufactured. Figure 10.39 shows a screen display of the screw jack part once it is completed.

After the part creation in the CAD system is complete, the programmer must process the part for its machining operations deciding on the tools to be used. For the screw jack the following tools were decided upon.

- 80° diamond-shaped-insert right-hand turning tool for rough facing and turning
- 0.500 in.-diameter drill for hole
- 0.125-in.-width parting tool for grooves or thread relief
- 30° diamond-shaped-insert right-hand turning tool for finish facing and turning
- Brazed carbide-tipped boring bar for rough and finish boring of tapered bore
- 60° threading tool for 1-24 UNF-2B thread
- 0.125-in.-width parting tool for part cutoff.

Once processing is complete, the programmer will check the tool library generated on the computer system to see if additional tools need to be generated. Figure 10.40 illustrates the necessary geometry and origin point that need to be created in CAD so tool information can be transferred into the tool library in CAM. Having this information stored in the CAM system allows programmers to create and display a variety of tooling packages for use on various part programs.

It has been shown that tools can be drawn up in CAD and its computer definition transferred to a tool library for later use in CAM. It should be noted, though, that tools can also be defined in the CAM system through the use of special icons and pop up menus such as those that appear in Figure 10.41.

Now that the tooling is complete, the programmer will reduce the part drawing to only the machining curves that are required to produce the part program. The programmer reduces the drawing, as shown in Figure 10.42, to its basic profile and then the system will convert them to machining curves with some basic entered commands. Figure 10.42 shows the labeled machining curves that appear as a profile of half the part. Curve *K*1 is the internal bore, *K*2 is the part center, *K*3 is the outside profile, and *K*99 is the original part stock outline.

Figure 10.43 shows us that the machining curves are made up of straight and circular line segments called spans. The CAM system gives us the freedom