COMPUTER-ASSISTED PART PROGRAMMING

Most parts machined on NC systems are considerably more complex. In the more complicated point-to-point jobs and in contouring applications, manual part programming becomes an extremely tedious task and subject to errors. In these instances it is much more appropriate to employ the high-speed digital computer to assist in the part programming process.

The part programmer's job

The part programmer's responsibility in computer-assisted part programming consists of two basic steps:

l. Defining the workpart geometry

2. Specifying the operation sequence and tool path

It is the part programmer's task to enumerate the ements out of which the part is composed. Each geometric element must be identified and the dimensions and location of the element explicitly defined.

After defining the workpart geometry, the programmer must next construct e path that the cutter will follow to machine the part. This tool path specification involves a detailed step-by-step sequence of cutter moves. the programmer must also provide other instructions to operate the machine tool properly.

The computer's job

The computer's job in computer-assisted part programming consists of the following steps:

l. Input translation- The input translation component converts the coded instructions contained in the program into computer-usable form, preparatory to further processing.

2. Arithmetic calculations- The arithmetic calculations unit of the system consists of a comprehensive set of subroutines for solving the mathematics required to generate the part surface.

3. Cutter offset computation-next task of the part programmer is that of constructing the tool path. The purpose of the cutter offset computations is to offset the tool path from the desired part surface by the radius of the cutter.

4. Postprocessor- The postprocessor is a separate computer program that has been written to prepare the punched tape for a specific machine tool. The input to the postprocessor is output from the other three components: a series of cutter locations and other ructions. The output of the postprocessor is the NC tape written in the correct format for the machine on which it is to be used.

Part programming languages - There have probably been over lOO NC part programming languages.

The following list provides a description of some of the important NC languages in current use.

APT (Automatically Programmed Tools)	The APT language was the product of the **MIT** developmental work on NC programming systems. Today it is the most widely used language in the United States. Although first intended as a contouring language, modem versions of APT can be used for both positioning and continues-path programming in up to five axes. Versions of APT for particular processes include APTURN (for lathe operations). APTMIL (for milling and drilling operations), and APTPOINT (for point-to point operations).
ADAPT (Adaptation of APT)	Several part programming languages are based directly on the APT program. One of these ADAPT, which was developed by IBM under Air Force contract. It was indented to provide many of the features of APT but to utilize a smaller computer. The full APT program requires a computing system that would have been considered by the standards of the 1960s. This precluded its use by many small and medium sized firms that did not have access to a large computer. ADAPT is not as powerful as APT, but it can be used to program for both positioning and contouring jobs.
EXAPT (Extended subset of APT)	There are three versions: EXAPT I- designed for positioning (drilling and also straight-cut milling), EXAPT II- designed for turning, and EXAPT III- designed for limited contouring operations. One of the important features of EXAPT is that it attempts to compute optimum feeds and speeds automatically.

UNIAPT	The UNIAPT package represents another attempt to adapt the APT language to use on smaller computers. The name derives from the developer, the United Computing Corp. of Carson, California. Their efforts have provided a limited version of APT to be implemente0 on minicomputers, thus allowing many smaller shops to possess computer-assisted programming capacity.
SPLIT (Sundstrand Processing Language Internally Translated)	This is a proprietary system intended for Sundstrand's machine tools. It can handle up to five-axis positing and possesses contouring capability as well. One of the unusual features of SPLIT is that the postprocessor is built into the program. Each machine tool uses its own SPLIT package, thus obviating the need for a special postprocessor.
COMPACT II	This is a package available from Manufacturing Data Systems, Inc. (MDSI), a firm based in Ann Arbor, Michigan. The NC language is similar to SPLIT in many of its features. MDSI leases the COMPACT II system to its users on a time-sharing basis; the part programmer uses a remote terminal to feed the program into one of the MDSI computers, which in turn produces the NC tape. The COMPACT II language is one of the most widely used programming languages. MDSI has roughly 3000 client companies, which use this system.

PROMPT	This is an interactive part programming language offered by Weber N/C System, Inc., of Milwaukee, Wisconsin. It is designed for use with a variety of machine tools, including lathes, machining centers, flame cutters, and punch presses.
CINTURN II	This is a high-level language developed by Cincinnati Milacron to facilitate programming of turning operations.

The most widely used NC part programming language is APT, including its derivatives (ADAPT, EXAPT, UNIAPT, etc.).

THE APT LANGUAGE –Our objectives are to demonstrate the English-like statements of this NC language and to show how they are used to command the cutting tool through its sequence of machining operations. There are over 4OO words in the APT vocabulary. Only a small fraction will be covered here. There are four types of statements in the APT language:

l. *Geometry statements*. These define the geometric elements that comprise the workpart. They are also sometimes called definition statements.

2. *Motion statements.* These are used to describe the path taken by the cutting tool.

3. *Postprocessor statements.* These apply to the specific machine tool and control system. They are used to specify feeds and speeds and to actuate other features of the machine.

4. *Auxiliary statements.* These are miscellaneous statements used to identify the part, tool, tolerances, and so on.

Geometry statements –

The definition of the workpart elements must precede the motion statements. The general form of APT geometry statements is:

symbol = geometry type/descriptive data

1) An example of such a statements is

P1= POINT/ 5.0, 4.0, 0.0

the statement is interpreted by the APT program to mean a point $x = 5.0$, y = 4.0, and $z = 0.0$.

2) Any symbols used as descriptive data must have been previously defined. For example, in the statement

P2 = POINT/INTOF, Ll, L2

3) A symbol can be used to define only one geometry element. The same symbol cannot

be used to define two different elements. For example, the following sequence would

be incorrect:

P1 = POINT /1.0, 1.0, 1.0

P1 = POINT / 2.0, 3.0, 4.0

4) Only one symbol can be used to define any given element. For example, the following two statements in the same program would render the program incorrect:

P1 = POINT/1.0, 1.0, 1.0

P2 = POINT 1.0, 1.0, 1.0

5) Lines defined in APT are considered to be of infinite length in both directions. Similarly, planes extend indefinitely and circles defined in APT are complete circles.

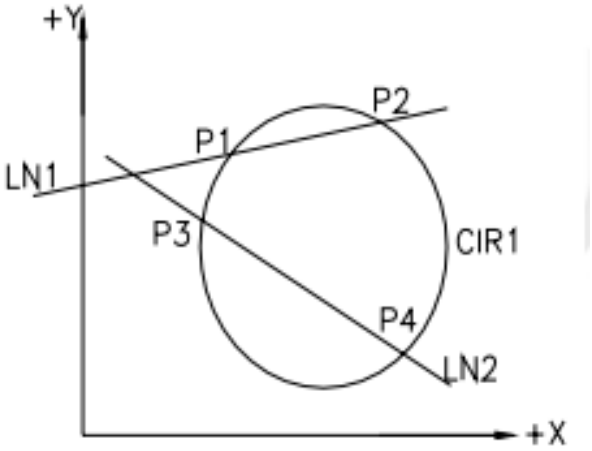
P2 = **POINT/YLARGE, INTOF**, L3, C1

specifies a point at the intersection of line L3 and circle C1 at a Y position above the center point of the circle.

By the Intersection of Line and Circle-

General syntax is = POINT/ $\begin{Bmatrix} XSMALL \\ XLARGE \\ YSMALL \\ YLARGE \end{Bmatrix}$, INTOF, line1, circle1

The modifiers XSMALL, XLARGE, etc. one of which is to be used, signify the point, which has algebraically small or large co-ordinate when projected onto that axis.

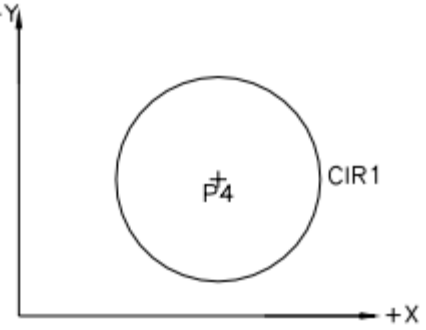


P1=POINT/XSMALL,INTOF,LN1,CIR1
P2=POINT/XLARGE,INTOF,LN1,CIR1
P3=POINT/XSMALL,INTOF,LN2,CIR1
P4=POINT/YSMALL,INTOF,LN2,CIR1

By the Centre of a Circle General syntax is

<SYMBOL>= POINT/ CENTER, circle

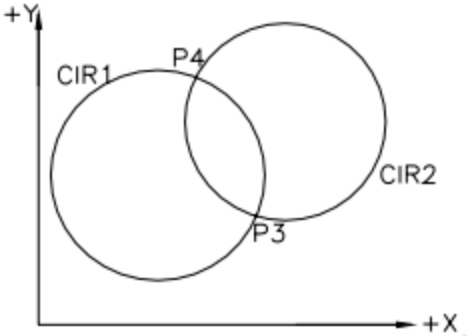
P4 = POINT / CENTER, CIR1



By the Intersection of Two Circles

General syntax is

SYMBOL= POINT/ $\begin{Bmatrix} XSMALL \\ XLARGE \\ YSMALL \\ YLARGE \end{Bmatrix}$, INTOF, circle1, circle2



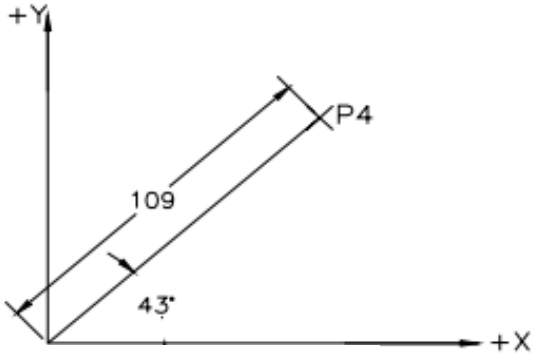
P3=POINT/YSMALL,INTOF,CIR2,CIR1

P4=POINT/YLARGE,INTOF,CIR2,CIR1

Polar Co-ordinates in a Co-ordinate Plane

General syntax

<SYMBOL>= POINT/ RTHETA, $\begin{Bmatrix} XYPLAN \\ YZPLAN \\ ZXPLAN \end{Bmatrix}$, angle, radius

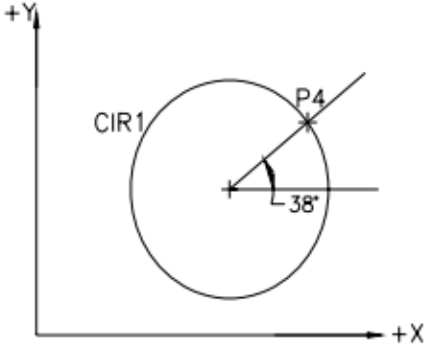


P4=POINT/RTHETA,XYPLAN,109,43

The radius must not be a negative value. The modifiers XYPLAN etc. specify the plane in which the point is lying.

On a Circle at an Angle with X-axis

P4 = POINT / CIR1, ATANGL, 38



To specify a line-

L1 = **LINE**/P0, P1 *specifies a line by two points, previously defined.*

L1 = **LINE**/1.0, 1.2, 1.3, 2.0, 2.1, 2.3

specifies a line by two points, given as explicit coordinates.

L2 = **LINE**/P2, **PARLEL**, L1

specifies a line through point P2 and parallel to line L1.

L3 = **LINE**/P1, **RIGHT**, **TANTO**, C1

specifies a line through point P1 and tangent to circle C1 on the right side of the center point. The point must not be inside the circle.

L4 = **LINE**/P1, **ATANGL**, 45, L1

specifies a line through point P1 at an angle of 45o to line L1.

L5 = **LINE**/P2, **PERTO**, L1

specifies a line through point P2 and perpendicular to line L1.

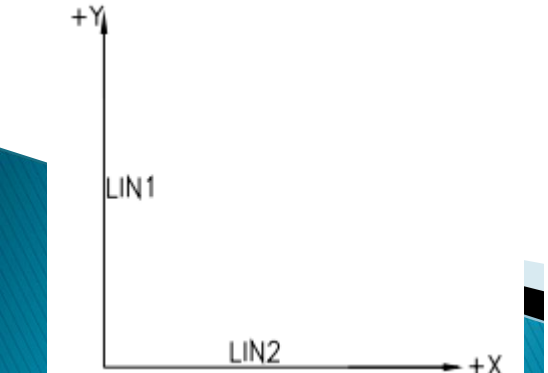
LIN2 = LINE/ PARLEL,LIN1,YLARGE,30

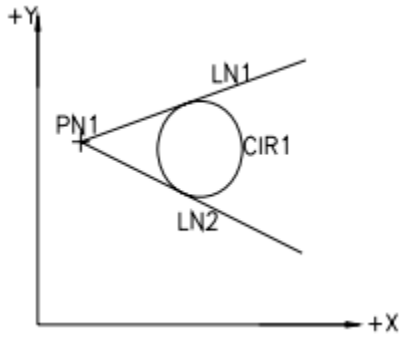
LIN2 = LINE/ PN1, ATANGL, 27, LIN1

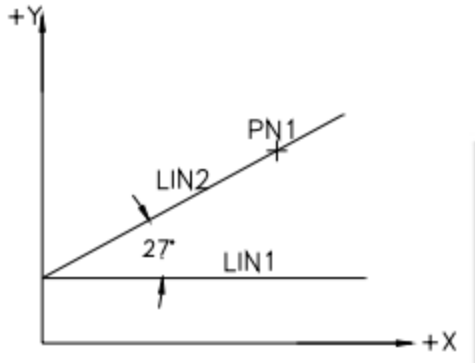
When the line is not specified, XAXIS is assumed.

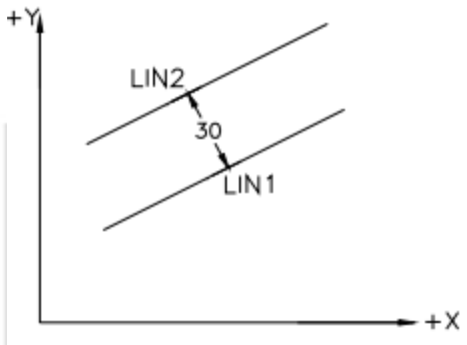
LIN1 = LINE / YAXIS

LIN2 = LINE / XAXIS

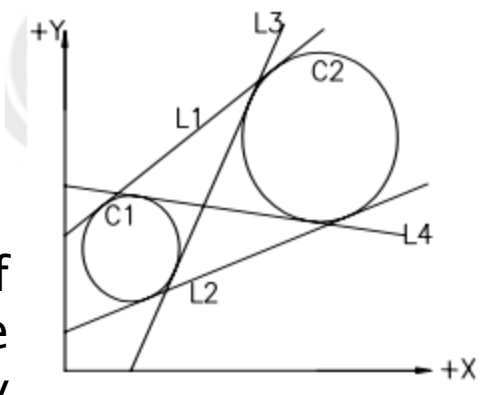








L1 = LINE/LEFT,TANTO,C1,LEFT,TANTO,C2
L2 = LINE/RIGHT,TANTO,C1,RIGHT,TANTO,C2
L4 = LINE/LEFT,TANTO,C1,RIGHT,TANTO,C2
L3 = LINE/RIGHT,TANTO,C1,LEFT,TANTO,C2



RIGHT and LEFT is established looking from the centre of the first circle specified in the definition towards the centre of the other circle. One circle must not be completely inside the other circle.

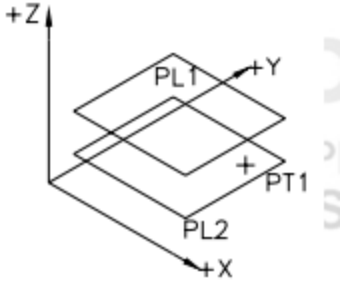
To specify a plane

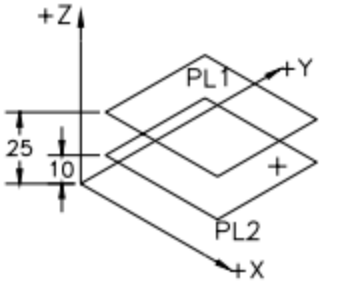
PL0 = **PLANE**/P0, P1, P2 *specifies a plane through three, noncollinear, previously defined points.*

PL2 = **PLANE**/P1, **PARLEL**, PL1 *specifies a plane through a point P1 parallel to a plane PL1.*

By the Coefficient of a Plane Equation a
X + b Y + c Z = d

PL1 = PLANE/ 0, 0, 0, 25
PL2 = PLANE/ 0, 0, 0, 10





To specify a circle

C0 = **CIRCLE/CENTER**, P0, **RADIUS**, 1.0

specifies a circle of radius 1 from a center point of P0.

C2 = CIRCLE/P1,P2,P3

C2 = CIRCLE / CENTER, P1, P2

C2 = CIRCLE / CENTER, P1, TANTO, L1

C2 = CIRCLE/YSMALL,P1,P2,RADIUS,27

C3 = CIRCLE/YLARGE,P1,P2,RADIUS,27

C2 = CIRCLE / CENTER, P1, SMALL, TANTO, CI1

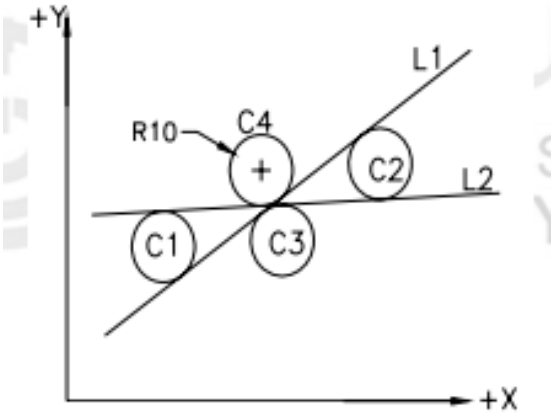
C3 = CIRCLE / CENTER, P1, LARGE, TANTO, CI1

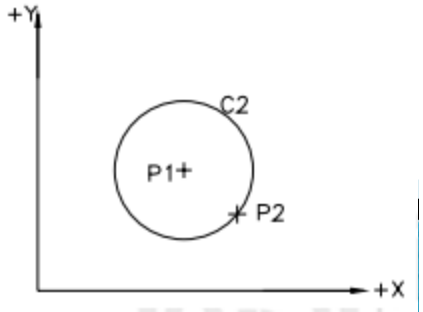
C1 = CIRCLE / YLARGE, L1, YSMALL, L2, RADIUS, 10

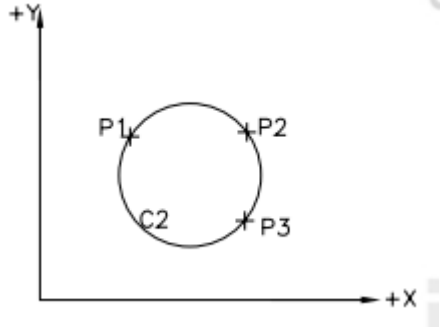
C2 = CIRCLE / YSMALL, L1, YLARGE, L2, RADIUS, 10

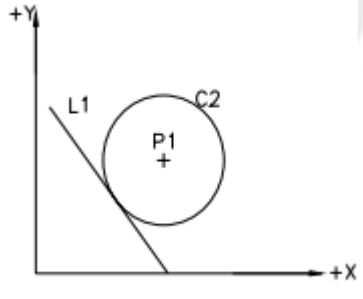
C3 = CIRCLE / YSMALL, L1, YSMALL, L2, RADIUS, 10

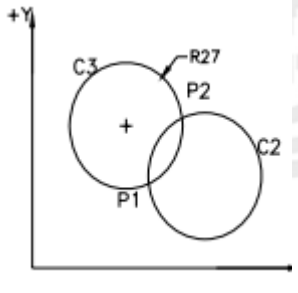
C4 = CIRCLE / YLARGE, L1, YLARGE, L2, RADIUS, 10

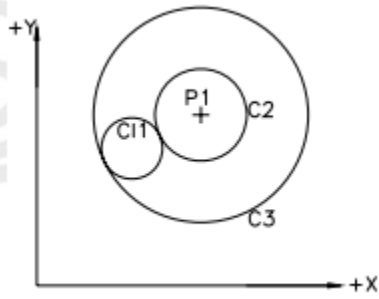






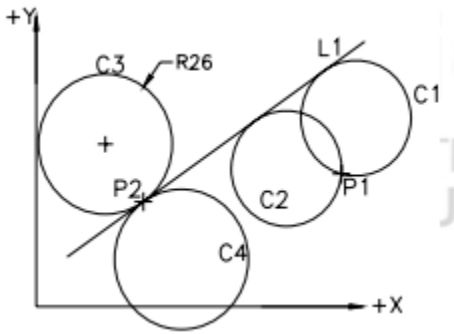






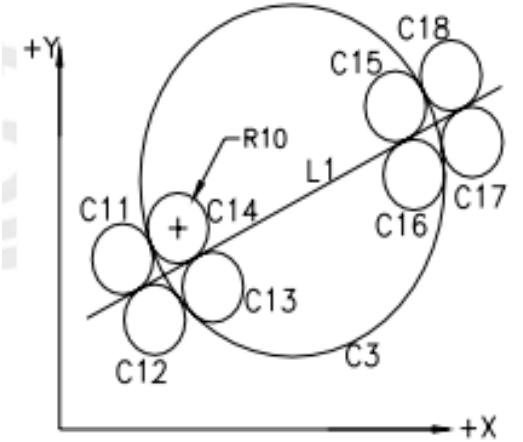
By a Tangential Line, a Point on the Circumference and Radius

C1 = CIRCLE / TANTO, L1, XLARGE, P1, RADIUS, 26
C2 = CIRCLE / TANTO, L1, XSMALL, P1, RADIUS, 26
C3 = CIRCLE / TANTO, L1, YLARGE, P2, RADIUS, 26
C4 = CIRCLE / TANTO, L1, YSMALL, P2, RADIUS, 26



By Two Tangential Circles and Radius

C11 = CIRCLE / YLARGE, OUT, C3, OUT, C4, RADIUS, 10
C12 = CIRCLE / YLARGE, OUT, C3, IN, C4, RADIUS, 10
C13 = CIRCLE / YLARGE, IN, C3, OUT, C4, RADIUS, 10
C14 = CIRCLE / YLARGE, IN, C3, IN, C4, RADIUS, 10
C15 = CIRCLE / YSMALL, OUT, C3, OUT, C4, RADIUS, 10
C16 = CIRCLE / YSMALL, OUT, C3, IN, C4, RADIUS, 10
C17 = CIRCLE / YSMALL, IN, C3, OUT, C4, RADIUS, 10
C18 = CIRCLE / YSMALL, IN, C3, IN, C4, RADIUS, 10



IN and OUT allows the selection of the considered circle by indicating the mode of tangency between the two circles.

MOTION COMMANDS- MOTION COMMAND/descriptive data

FROM/P0 or FROM/ 0.0, 1.0, 2.0

GO/TO, drive surface, TO, part surface, TO, check surface

FROM : From to specify the start point for the cutter

GO : To specify the point of contact between the tool and the work surface

GOBACK : Action verb go back

GODLTA : To move the tool in relative coordinates

GODOWN : Action verb go down

GOFWD : Action verb go forward

GOLFT : Action verb go left

GORGT : Action verb go right

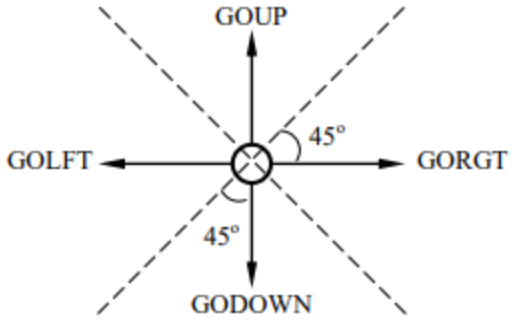
GOTO : To control the flow of program

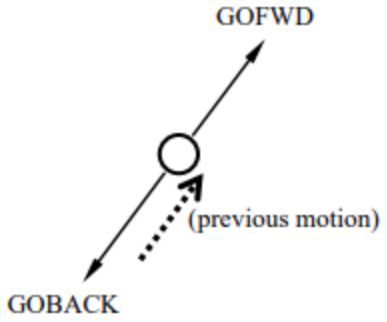
GOUP : Action verb go up

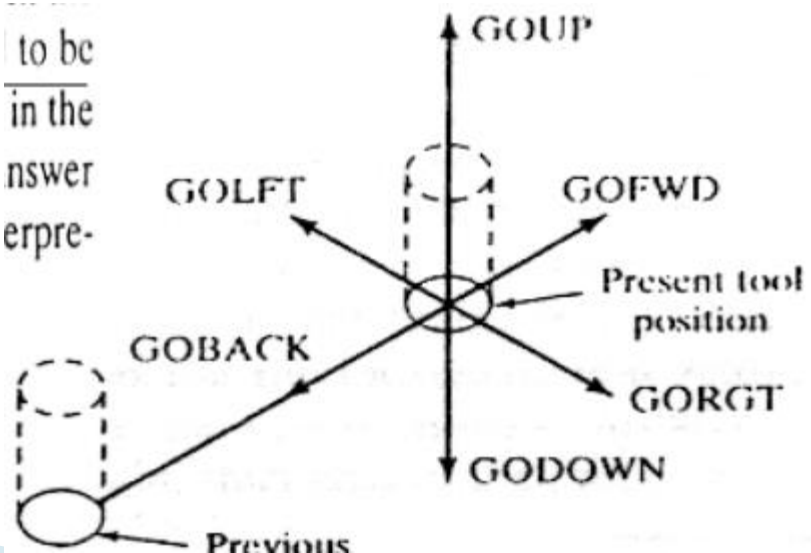
Point to point motion – may be specified as absolute, or as incremental (relative to the last point visited). An example of absolute, point to point motion is:

GOTO/P0

GODLTA/1.0, 2.0, 3.0







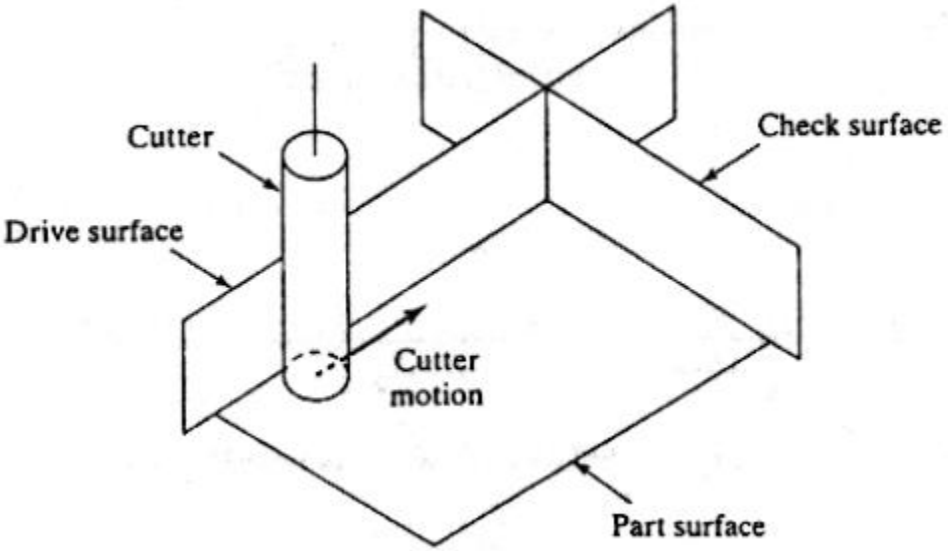
Drive surfaces represent the surface along which the vertical edges of the tool will follow. Part surfaces specify the surfaces the tip of the tool will follow. And check surfaces describe where the tool will come to rest after it has completed the motion of the current step. There are four locations for the tool to stop with respect to a check surface. These four possibilities each have their own modifier words.

The **TO** modifier stops the tool when the first surface of the tool would come into contact with the check surface.

The **ON** modifier stops the tool where the center point of the tool would come into contact with the check surface.

The **PAST** modifier stops the tool where the last surface of the tool would contact the check surface.

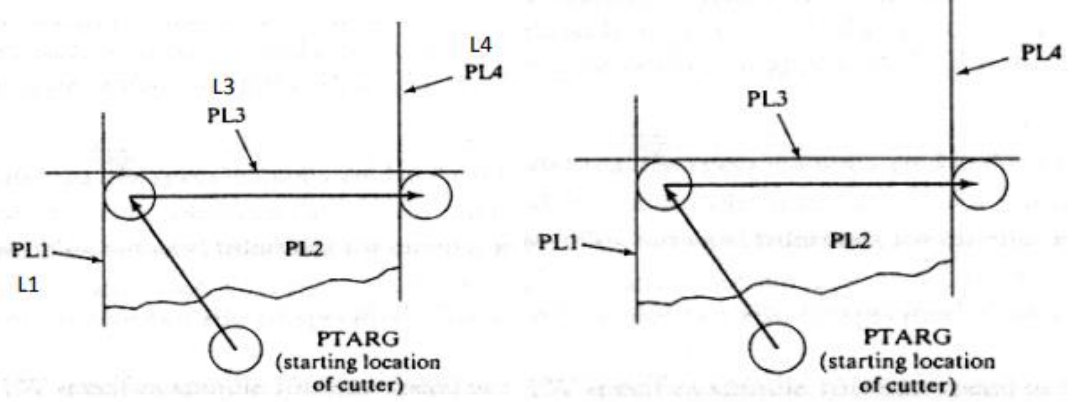
And the **TANTO** modifier stops the tool at the point of circular tangency with the edge of the tool.

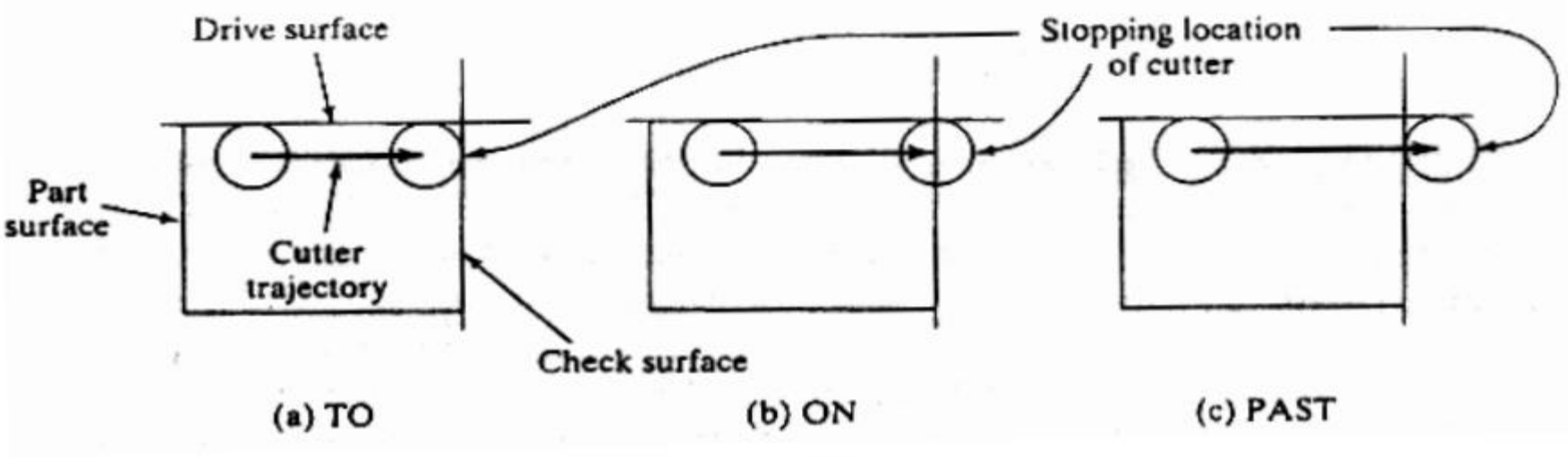


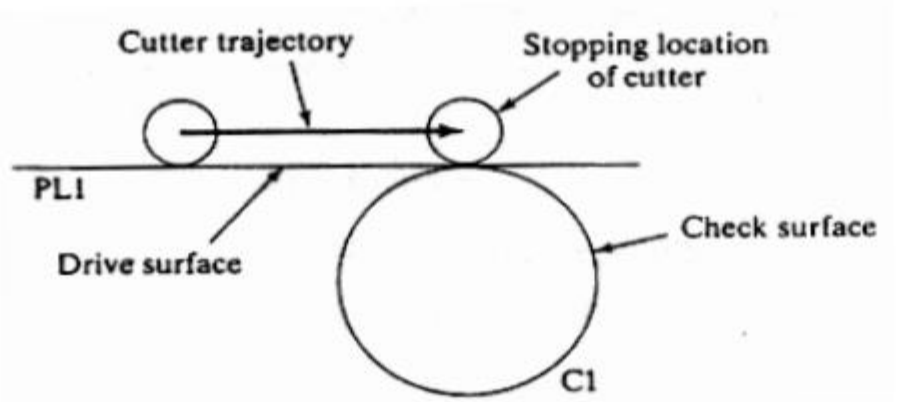
The GO/ command is used to initialize a sequence of contouring motions and may take alternative forms such as **GO/ON, GO/TO, or GO/PAST.**

GO/TO, PL1, TO, PL2, TO, PL3

GORGT/PL3, PAST, PL4







Post-Processor Statements

These statements provide processing parameters to the post-processor program. Typical programs will require parameters for feeds, speed, and other tool/spindle/machine controls. Examples:

SPINDL/600 specifies the spindle to be 600 rpm.

FEDRAT/6.0 specifies a feed rate of 6 inches per minute.

TURRET/T2 specifies loading tool # 2 in the turret.

A final post-processor statement must specify to the post-processor program what type of machine is intended for the final NC code, and the specific controller to generate the code for. An example is:

MACHIN/MILL,2 specifies a mill machine type, and controller type 2

COOLNT/ON

COOLNT/OFF

FINI-End the program

END-Stop the M/C

CLPRNT-cutter location print is used to take print out

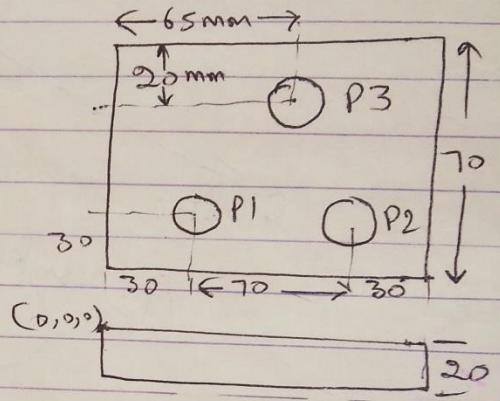
Auxiliary Statements-

These statements complete the APT programming language, and include the FINI statement to mark the end of the program as well as statements to define the width of the tool. An example of the latter is:

CUTTER/0.25 specifies a quarter-inch cutter diameter

The computer would then know to calculate a 0.125 inch offset to accommodate the cutter diameter in computing the center of the tool.

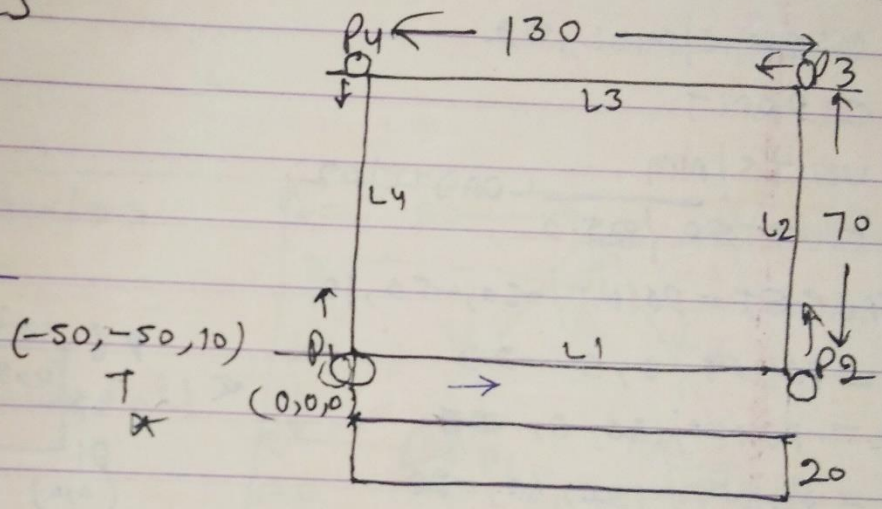
eg → S = 500 rpm & f = 0.05 mm



MACHIN/DRILL, 05 → m/c identify no
CLPRNT → printout
UNITS/MM
TARGET = POINT/0, -50, +50
P1 = POINT/30, 30, 10.0
P2 = POINT/100, 30, 10
P3 = /65, 50, 10 → LOADTL/01
FROM/TARGET → from the initial pt i.e. Target pt
RAPID
GOTO/P1 — for pt to pt motion,
SPINDL/500, CLW
FEDRAT/0.05, MMPR → feedrate is 0.05 mm/rotation
GODLTA/0, 0, -35

GODLTA/0, 0, 35
RAPID
GOTO/P2
SPINDL/500, CLW
FEDRAT/0.05, MMPR
GODLTA/0, 0, -35
GODLTA/0, 0, 35
RAPID
GOTO/P3
SPINDL/500, CLW
FEDRATE/0.05, MMPR
GODLTA/0, 0, -35
GODLTA/0, 0, 35
RAPID
GOTO/TARGET
SPINDL/OFF
FINI
END

2 Milling → cutter dia = 25



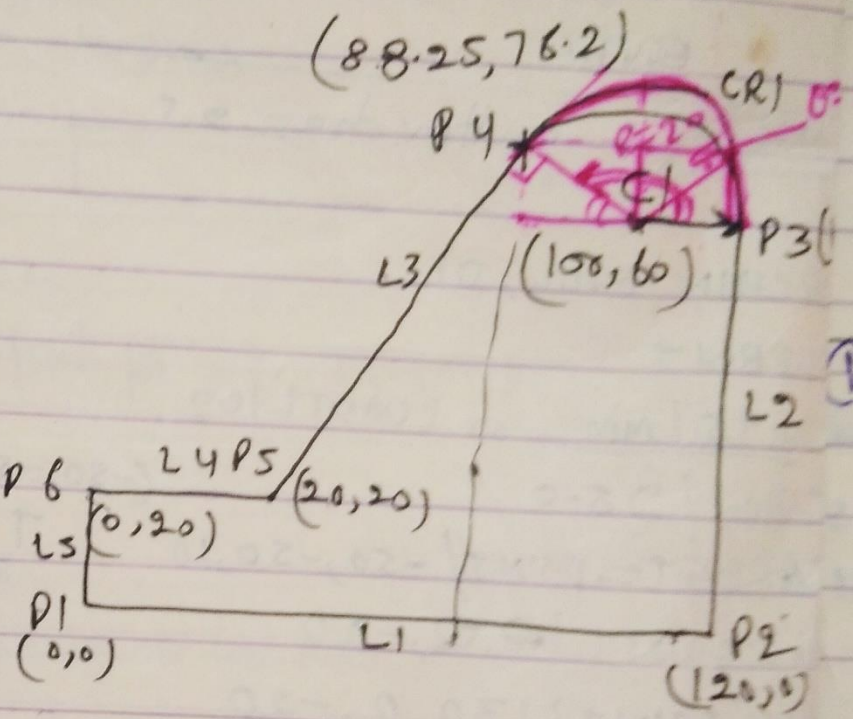
MACHIN/MILL, 01
ELPRNT
UNITS/MM or LOADTL/02
Cutter/25.0
TARGET=POINT/-50,-50,10
P1=POINT/0,0,-20
P2=POINT/130,0,-20
P3=POINT/130,70,-20
P4=POINT/0,70,-20

L1=LINE/P1,P2
L2=LINE/P2,P3
L3=LINE/P3,P4
L4=LINE/P4,P1
PL1=PLANE/P1,P2,P3
FROM/TARGET → LOADTL/02
GO/TO, L1, PL1, ON, L4 (→ drive surf → part surf → check surf)
SPINDL/500, CLW

FEDRATE/20, MMPM
GORGT/L1, PAST, L2
GOLFT/L2, PAST, L3
GOLFT/L3, PAST, L4
GOLFT/L4, PAST, L1
RAPID
GOTO/TARGET
SPINDL/OFF
FINI

(3)

```
MACHIN/MILL,02
CLPRNT
UNITS/MM
CUTTER/25.0
LOADTL/02
TARGET=POINT/-50,-50,10
P1=POINT/0,0,-20
P2=POINT/120,0,-20
P3=POINT/120,60,-20
C1=POINT/100,60,-20
P4=POINT/88.25,76.2,-20
P5=POINT/20,20,-20
P6=P    /0,20,-20
```



L1 = LINE / P1, P2

L2 = LINE / P2, P3

CR1 = CIRCLE / CENTER, C1, TANTO, L2

L3 = LINE / P4, P5

L4 = LINE / P5, P6

L5 = LINE / P6, P1

PL1 = PLANE / P1, P2, P3

FROM / TARGET

GO/TO, L1, PL1, ON, L5 — for contouring mot

SPINDL / 500, CLW

FEDRATE / 50, MMPM

GORGT / L1, PAST, L2 — for intermediate

GOLFT / L2, TANTO, CR1

GOFWD / CR1, PAST, L3

GOFWD / L3, TO, L4

GOFWD / L4, PAST, L5

GOLFT / L5, PAST, L1

RAPID

GOTO / TARGET

SPINDLE / OFF

FINI